UNIVERSITÀ DEGLI STUDI DI MILANO

Computer Science Department

PhD in Computer Science,
XXIX Cycle

Mobile assistive technologies for people with visual impairment: sensing and conveying information to support orientation, mobility and access to images

INF/01

PhD candidate:
Andrea GERINO

Advisor:
Dr. Sergio MASCETTI
Co-Advisor:
Dr. Cristian BERNAREGGI

School Director:
Prof. Paolo BOLDI

Academic Year 2015/16

UNIVERSITÀ DEGLI STUDI DI MILANO

Computer Science Department

PhD in Computer Science,
XXIX Cycle

Abstract

Mobile assistive technologies for people with visual impairment: sensing and conveying information to support orientation, mobility and access to images

Andrea GERINO

Smartphones are accessible to persons with visual impairment or blindness (VIB): screen reader technologies, integrated with mobile operating systems, enable non-visual interaction with the device. Also, features like GPS receivers, inertial sensors and cameras enable the development of Mobile Assistive Technologies (MATs) to support people with VIB. A preliminary analysis, conducted adopting an user-centric approach, highlighted some issues experienced by people with VIB in everyday activities from three main fields: orientation, mobility and access to images.

Traditional approaches to address these issues, based on assistive tools and technologies, have some limitations: in the field of mobility, for example, existing navigation support solutions (e.g. the white cane) cannot be used to perceive some environmental features like crosswalks or the current state of traffic lights; in the field of orientation, tactile maps adopted to develop cognitive maps of the environment are limited in the amount of information that can be represented on a single surface and by the lack of interactivity, two issues experienced also in other fields where access to graphical information is of paramount importance like, for example, didactics of STEM subjects.

This work presents new MATs that deal with these limitations by introducing novel solutions in different fields of Computer Science. Original computer vision techniques, designed to detect the presence of pedestrian crossings and the state of traffic lights, are used to sense information from the environment and support mobility of people with VIB. Novel sonification techniques are introduced to efficiently convey information with three different goals: first, to convey guidance information in urban crossings; second, to enhance the development of cognitive maps by augmenting tactile surfaces; third, to enable quick access to images.

Experience reported in this dissertation shows that the proposed MATs are effective in supporting people with VIB and, in general, that mobile devices are a versatile platform to enable affordable and pervasive access to assistive technologies. Involving target users in the evaluation of MATs emerged as a major challenge in this work. However, it is shown how such challenge can be addressed by adopting large scale evaluation techniques typical of HCI research.

Contents

Chapter 1

Introduction

Today, smartphones are accessible to persons with visual impairment or blindness (VIB). Screen reader technologies, integrated with mobile operating systems, enable non-visual interaction with the device by means of gesture-based interaction paradigms and auditory feedback. People with VIB can now interact with the mobile OS as well as with most applications by third party developers. New opportunities arise thanks to the availability of features like GPS receivers, inertial sensors and cameras, all integrated in the same device. By using these features it is possible to develop novel Mobile Assistive Technologies (MATs) to support people with VIB [54]. The development of MATs requires to address challenges in many fields of Computer Science like accessibility, computer vision, non-visual representation of information and human-computer interaction.

This dissertation describes MATs that have been developed responding to specific needs highlighted by people with VIB, presenting contributions in the fields of orientation and mobility, access to images, didactics and large scale evaluation of assistive technologies. Challenges encountered during the development of each MAT are reported, together with the design decisions and novel solutions introduced.

Solutions presented in this work are validated by a large number of user evaluations, conducted with different methodologies. Each methodology is described in detail, and the results of each evaluation are reported. The analysis of evaluation results identifies interesting insights in MATs and a number of future research directions.

1.1 Problem description

We adopted an user-centric approach to identify everyday activities that may benefit from MATs. By collaborating with people with VIB and associations[1], we analyzed some common problems and identified those that may benefit from MATs.

A first set of problems belongs to the field of orientation and mobility. Orientation and mobility skills are developed since childhood, often with the support of orientation and mobility specialists.

[1] "Istituto dei Ciechi di Milano" (Milan's institute of blind people), "Unione Italiana Ciechi" (Italian Union of Blind and Partially Sighted People) and "Retina Italia Onlus"

However, people with VIB meet a number of challenges while walking independently without a guide, mainly concerning the acquisition of information about the environment [67, 9]. There are many assistive tools that may be used to address this challenge. For example, tactile maps support the development of the cognitive map of an environment [47], and the white cane is used to perceive environmental information, like obstacles, while walking.

A second set of problems pertains to the field of didactics and, in particular, of mathematics and other STEM subjects. Learning these topics poses many challenges to people with VIB because they heavily rely on graphical information that is not accessible (e.g. complex formulas, histograms and charts) [8, 43]. A common approach to enable access to graphical information is to transfer images on tactile drawings that can be haptically perceived. Another approach consists in adopting sonification (i.e. to convey data through the acoustic channel) to represent image features through sound. Solutions from the literature [79] show that, by using image sonification techniques, it is possible to acquire enough visual skills to pass the blindness threshold in the visual acuity tests proposed by the World Health Organization.

Existing solutions, based on assistive tools and technologies, have some limitations. First, access to assistive tools may be difficult. For example, tactile maps of environments must be manually produced either by sighted individuals or by using expensive hardware and, moreover, they may not be available for all locations. Other factors may limit access to some assistive tools. Take for example the "eSight" [2], an assistive augmented reality device built exclusively for people with low vision, and the "OrCam MyEye" [3], a system that runs computer vision algorithms to provide object recognition and OCR to people with VIB. The two devices share two main issues. First, to the best of our knowledge, these systems are not extendible, in the sense that there is no way to develop custom software to address different accessibility needs. Second, the devices price (15,000$ for the eSight and 2500$ for the OrCam) could limit their availability to many end users.

Limitations may also arise when using existing solutions in particular contexts. For example, navigation support tools may be not enough to perceive some types of environmental information: the white cane cannot be used to detect road markings like crosswalks or the current state of pedestrian traffic lights. Tactile drawings are limited in the amount of information that can be represented in a single sheet and students may need to carry many tactile drawings in order to study topics like geometry or function graphs. Also, because the exploration of tactile drawings requires good spatial skills and a lot of concentration, it may discourage young math students from practicing maths and learning STEM subjects. Image sonification techniques enable non-visual access to image information, however a long training is often required in order to become proficient.

Finally, there are issues also from the point of view of researchers in assistive technologies. In order to guide the iterative design process of an assistive technology and evaluate its effectiveness, it is of paramount importance to understand users behavior during interaction. In order to collect valuable feedback to deliver on this objective, a large number of tests must be performed involving many representative users. However, performing these tests requires a large effort that sometimes is not feasible, considering that papers published in recent accessibility literature often do not appropriately include representative users [78]. Also, in many cases, tests are conducted

[2] http://www.esighteyewear.com
[3] http://www.orcam.com

over a short period of time and this represents a severe limitation in the evaluation of certain key aspects of HCI such as, for example, the learning curve of an assistive technology.

1.2 Contributions

In this work we present our contributions in the fields of orientation and mobility, access to images, didactics and large scale evaluation of assistive technologies.

We introduce a novel computer vision technique to detect zebra crossings [3, 55], a particular type of pedestrian crosswalk [4] also called "continental crosswalk" in the United States. The technique, called "Zebra Recognizer", delivers on two main objectives. First, it removes projection distortion from the acquired image, hence improving the accuracy of the recognition and making it possibile to compute the quantified relative position of the crossing with respect to the user. Second, it is efficient and suitable for realtime processing on mobile devices, as it performs the most complex calculations on the device's GPU. The technique has been evaluated on a dataset of about 4200 annotated video frames, obtaining 1.0 precision and 0.89 recall scores.

We propose "TL-Recognizer" [56, 57], a novel computer vision technique to detect the presence and the current state of pedestrian traffic lights (i.e. if the person must wait or is allowed to cross). The technique differs from similar contributions on two main aspects. First, it adopts a robust method to acquire images with proper exposure in different illumination conditions. Second, it implements multi-resolution processing and parallel computation techniques in order to reduce processing times without affecting recognition performances. A dataset of 1252 annotated images is used for performance measurements, which show that the technique obtains 1.0 precision and 0.81 recall scores.

Two auditory guiding modes based on data sonification are introduced to address the challenge of conveying guidance information during road crossings without distracting the user from the surrounding environment [59]. Target users have been involved in the sound design process through informal discussions and a preliminary evaluation. We implemented the two guiding modes based on sonification, along with a less innovative solution relying on speech, in a mobile application which adopts the Zebra Recognizer technique from our previous contributions. Three evaluations involving, in total, 11 blindfolded sighted subjects and 15 blind individuals showed that the mobile application can effectively guide people in crossing the road with any of the three guiding modes. However, most test subjects (75%) preferred one of the two auditory guiding modes.

Another contribution [65] faces the problem of supporting independent orientation of visually impaired people in unknown environments. The intuition is to enable the creation of the cognitive map of an environment prior to journeying. In order to deliver on this intuition we propose an auditory display that reproduces environmental information (audio descriptions and environmental sounds) while the user is exploring the tactile map of an area. A prototypal auditory display, developed as a mobile application, has been evaluated with 5 visually impaired individuals and it obtained a favorable assessment.

[4]Described in "Art. 145, d.p.r. 16/12/1993, n. 495, in materia di "regolamento di esecuzione e di attuazione del nuovo codice della strada" relativo all'art. 40 c.s."

In the field of didactics, we propose "Math Melodies" [28], a tablet application[5] targeted to students in primary school. The application presents accessible exercises in a narrative context that entertain children and stimulates them in practicing math. An interaction paradigm based on "Audio Icons" is adopted to achieve three main objectives. First, to enable access to exercises that heavily rely on images. Second, to reduce the time and mental workload required to grasp the exercises. Third, to entertain both sighted an visually impaired students and stimulate them to keep practicing. We evaluated the final version of the application with 3 blind and 2 sighted students in primary school. All children enjoyed the application and were entertained by "Audio Icons". All blind children experienced difficulties in the early exploration of tables, and needed help by a sighted supervisor. However, after 2 minutes of supervised training, all children got familiar with the application and were able to autonomously solve the exercises.

Also, we present "Audio Functions" [81], a mobile application that enables blind and visually impaired students to explore function diagrams through sound. We introduce three novel exploration modes to support the user in exploring the function shape through sonification, some taking advantage from direct interaction with the touchscreen to enable proprioception. By using the application, students can independently insert the function equation though a specialized keyboard and perceive both the function shape and additional information, including quantitative information (e.g., the pair $\langle x, f(x)\rangle$) and relevant points, like maximum and minimum. We evaluated "Audio Functions" with 7 blind subjects, all with secondary school education in mathematics and familiar with tactile drawings. All subjects were asked to explore three different function diagrams, each one with a different tool. Every user obtained a better understanding of the function graph by using our solution.

We deliver on the need for new solutions to enable non-visual access to image information by introducing five novel "sonification modes" i.e., software modules that combine an image exploration paradigm with an image sonification technique [29]. The novel sonification modes are based on a mono dimensional exploration paradigm along the vertical axis and adopt sound spatialization (employing both interaural level and time differences) and sound equalization filtering to provide spatial information about the missing dimension. In order to evaluate the performances of the novel sonification modes, we implemented them in an iOS application called "The Invisible Puzzle Prototype". The application instructs users on how to use one sonification mode, challenging them to recognize some (invisible) shapes, and measuring their performances. A first evaluation, conducted with 178 sighted subjects and 49 blind ones, showed that one of the techniques shows statistically significant better performance with respect to some of the other techniques.

Another goal of this work is to evaluate MATs with a large number of subjects and for long periods of time. We adopt a methodology typical of HCI research to perform the large scale evaluation of two applications, "iMove"[6] and "The Invisible Puzzle Game"[7] [42, 30]. In particular, iMove usage data collected over a period of four months consists in 771,975 log records across 17,624 unique user pseudo-identifiers. By analyzing this data with both inferential and exploratory methods, including unsupervised learning of user clusters, we pointed out a number of use properties of iMove, including functions that are more commonly used, and clusters of users based on common interaction parameters.

[5]Available on the *AppStore*: `https://itunes.apple.com/us/app/math-melodies/id713705958`
[6]Available on the *AppStore*: `https://itunes.apple.com/us/app/imove/id593874954`
[7]Available on the *AppStore*: `https://itunes.apple.com/us/app/the-invisible-puzzle/id1051337548`

1.3 Common factors between research contributions

Research described in this dissertation addresses challenges in many different areas of Computer Science like, for example, accessibility, computer vision, non-visual representation of information, human computer interaction and software engineering. Despite the heterogeneity of our contributions, there are three different aspects that make all of them part of a coherent research work.

First, there is a logical connection between the contributions. Our first contributions explored the use of computer vision techniques to sense information from the surrounding environment [3, 55, 56, 57] with the goal of supporting orientation and mobility of people with VIB. However, while working on these contributions, we identified a major challenge: to efficiently convey guidance information to the user without distracting her from the surrounding environment. In order to address this challenge we started to explore the field of non-visual representation of information and presented two novel auditory guiding modes based on sonification [59] and an auditory display to support the development of a cognitive map of environments before journeying [65]. The experience made in the field of non-visual representation of information inspired us new ways to use sonification to enable access to images, one of the challenges highlighted in our problem description. We then introduced new interaction modalities and sonifications to support learning of STEM subjects [28, 81] and explored a more general approach to the sonification of binary images [29, 30]. In our contributions we adopt a user centered design approach, involving target users in all development phases of the proposed solutions. However, since our early works, we had to endure a large effort to identify test candidates and involve them in the evaluations. The problem is well known in the literature, so we started to investigate solutions to reduce the effort required to perform tests with a large number of representative users, adopting a methodology to provide training, administer tests and collect usage statistics [29, 30] and extending the approach to evaluate assistive applications published on mobile app stores [42].

A second aspect that links research presented in this dissertation is the profile of target users, i.e. people with VIB. In its “Global data on visual impairment 2010” report, the World Health Organization estimates that 285 million people are visually impaired worldwide. Among those, 39 million are blind. Visual impairment is a matter of great concern in developing countries, where access to health care is still an issue, as well as in developed countries, where elderly population increases and more people will be at risk of visual impairment due to chronic eye diseases and aging processes, making the availability of affordable assistive technologies even more important.

Finally, the last common aspect is that all solutions we introduce are designed to be used on general purpose hardware: mobile touch-screen devices such as smartphones and tablets. This choice brings several advantages. First, people with VIB can adopt a single device to perform multiple tasks, reducing the number of single purpose tools they must carry with them. Second, as these device are mass produced, their cost is often lower if compared with that of devices built specifically for a small population. Third, mobile devices have large on-board processing capabilities, one or more cameras and advanced sensors such as the GPS and the inertial measurement unit (IMU), technologies that enable the development of assistive technologies that source information from the user’s environment. Fourth, assistive technologies are often embedded into mobile operating systems and provide a single interaction paradigm that provides access to all applications, dramatically reducing the amount of training required to use different assistive solutions shipped as mobile applications for the same platform. Finally, touch screen based interaction let users take advantage of proprioception to grasp spatial concepts like, for example, the shape of function graphs.

1.4 Outline

Chapter 2 introduces the state of the art of accessibility research focusing on mobile assistive technologies. In Chapter 3 we present a solution, in the field of orientation, that supports people with VIB in developing the cognitive map of an environment prior to journeying. Two contributions in the field of mobility, with the goal of assisting people with VIB in road crossings, are presented in Chapter 4 and Chapter 5. Chapter 6 highlights our efforts to support didactics of maths and STEM subjects by enabling access to images. We describe how we adopted a large scale evaluation methodology to a MAT available to the general public in Chapter 7. Finally, Chapter 8 concludes this work.

Chapter 2

Related Work

In this chapter we introduce the state of the art of accessibility research in the fields of orientation and mobility, image sonification, didactics of math and STEM subjects and large scale evaluation of assistive technologies.

2.1 Orientation and Mobility

Many solutions have been proposed to support orientation and mobility of people with VIB, broadly divided in two categories: pre-navigation solutions and in-navigation ones. Pre-navigation solutions enable pre-learning of routes and formation of cognitive maps of environments prior to journeying. Tactile maps with Braille labels are used to such extent. In-navigation solutions, instead, support the user by providing guidance while traveling. An issue of the pre-navigation approach based on tactile maps is that the amount of information that can be represented on the map is limited [82], as it is constrained by the size and complexity of the map. To address this issue, Parkes [66] proposes to augment tactile maps by providing auditory feedback about map features while they are being touched by the user. A prototypal system adopting spoken audio, verbal landmarks, environmental audio and auditory icons to convey additional information about the environment is presented by Jacobson [40]. A similar solution proposed by Schneider et al. [76] adopts computer vision techniques to track user interactions on a tactile grid, mapping user touches and objects on the digital map of an environment. However, these solutions have limitations because the adoption of either a touch pad or a touch grid does not convey direct stimulus from tactile relief and the user has to rely solely on proprioception to reconstruct spatial features, an activity that may require a lot of concentration thus becoming cognitively demanding. O'Sullivan et al. [64] propose the "Audio Tactile Maps" (ATMap) system to provide interactive auditory feedback for paper tactile maps. The idea is to use vision-based techniques to track a user's finger while she explores a tactile map and provide auditory feedback about the touched map features. Feedback is provided through both speech audio and recordings of characteristic acoustical features of the environment, such as ambient noise and self produced sounds. Authors perform a preliminary evaluation of the idea in a "Wizard of Oz" experiment where a person with VIB is asked to explore an interactive tactile map, obtaining speech-based feedback from a text-to-speech synthesizer operated by the test supervisor. The results of such evaluation highlighted the need to develop a fully functional prototype of the system and conduct an extended evaluation with people with VIB, a need that is addressed by a contribution presented in Chapter 3.

A solution to support both pre-navigation and in-navigation tasks is the “MOBIC Travel Aid”, introduced by Petrie et al. [70]. The system has two components: a desktop application to support the user while planning the journey by providing access to information about the travel environment (e.g. maps, public transport information) and a mobile unit, consisting in an handheld computer equipped with GPS, that provides guidance while journeying. A limitation of MOBIC is that it relies on dedicated hardware that the user must carry while traveling (along with other tools like the white cane). Similar in-navigation solutions are now available as applications for general purpose devices such as smartphones. An example is “iMove”[1], an iOS application that informs the user about the current address and nearby points of interest. iMove retrieves contextual information by performing geo-spatial queries to remote data sources. A common issue with both MOBIC and iMove is that there is no guarantee that information about the environment (e.g. the position of road crossings) is up-to-date or available at all.

A different approach has been followed by researchers in order to source information directly from the surrounding environment. These solutions aim to support the mobility of persons with VIB by reasoning over data captured by smartphone devices like inertial sensors and the onboard camera. Se et al. [77] propose a technique to detect pedestrian crossings in images by using computer vision. The Hough line segment detector is adopted to identify line segments and their vanishing points. Then, outliers are filtered by a Random Sample Consensus algorithm. The remaining segments are finally validated using cross ratio constraint. This solution has two main limitations. First, it fails to detect a crossing if its pattern is not completely in the camera’s field of view or if it is covered by objects. Second, it is not evaluated with a large set of images. The first issue is addressed by a new technique proposed by Uddin et al [85] that improves the effectiveness of the detection algorithm. However, also this solution has not been extensively evaluated. Another limitation of the aforementioned solutions is that they focus on the detection technique and do not consider the problem of how to provide feedback to the user.

Ivanchenko et al. [37] illustrate a technique to detect pedestrian crossings in images captured in realtime by a smartphone’s camera. The technique is implemented in the “Crosswatch” prototype, a mobile application that provides simple feedback to the user, in the form of an audio tone, when a crossing is detected. In another contribution, Ivanchenko et al. [38] extend Crosswatch with a new computer vision technique to identify standard “two-stripe” crosswalk patterns. The novel solution determines the lateral position of the user with respect to the corridor defined by the two stripes and provides speech feedback to support the user in staying inside the crosswalk. With a similar approach, Ahmetovic et al. [4] propose “ZebraLocalizer”, a technique to detect zebra crossings and compute the approximate position of the user with respect to the crosswalk. ZebraLocalizer uses data from the accelerometers to improve the recognition performance and to provide more accurate information about the relative position of the zebra crossing. Position information is then used to provide guidance, in the form of speech messages, to support the user in correctly aligning to the crossing. This solution has two main issues. First, the position of the user is approximated, as the technique does not take into account the perspective distortion of zebra crossing features and measures metric properties

[1]Available on the *AppStore*: `https://itunes.apple.com/us/app/imove/id593874954`

like distances and angles in perspective images, introducing error. Second, the solution provides feedback in the form of speech messages which may distract the user from paying attention to the environment during the actual crossing. These issues are addressed in an extension of the original work by Ahmetovic et al. presented in Chapter 4.

Another problem that has been addressed adopting computer vision techniques on mobile devices is that of recognizing the state of pedestrian traffic lights. Ivanchenko et al. [39] present a recognition algorithm, targeted at smartphones, to detect the status of traffic lights in the U.S.. The technique is divided in two steps. First, smartphone sensors are used to determine the horizon's position. Second, traffic lights are identified in the area above the horizon by looking for both the circular light and the shape of the pedestrian. Another approach is followed by Roters et al. [73], as they propose an algorithm consisting in three stages: "Identification", "Video analysis" and "Time-based verification". The novelty of this approach is that it performs spatial and temporal reasoning, analyzing sequences of images to validate candidate traffic lights. Experimental results conducted by the authors show that the analysis of image sequences yields better results than the analysis of still images.

The aforementioned techniques share a common issue: images are processed after their acquisition with algorithms designed to be robust under different lighting conditions. However, as highlighted by Diaz-Cabrera et al. [25], obtaining consistent accuracy in different illumination conditions is sometimes unfeasible. For example, particular lighting conditions may produce underexposed and overexposed images from which it is impossible to precisely reconstruct color information [24]. In order to overcome this problem, camera parameters should be dynamically tuned depending on the current illumination condition. This objective is partially met in a technique proposed by Diaz-Cabrera et al. [24] that adapts camera settings (i.e. shutter and gain) depending on values from each image's sky pixel histogram. However, it is unclear how the technique determines camera settings when images do not contain sky pixels. A novel technique to efficiently detect the state of traffic lights on mobile devices, adapting camera parameters to the current lighting condition, is presented in Chapter 5.

A common aspect of many solutions presented in this section is that they are mostly focused on improving the detection quality of environmental features (pedestrian crossings and traffic lights). However, in order to provide solutions that really support people with VIB in orientation and mobility tasks, it is important to find effective ways to convey information about the sensed environmental features. The solution proposed by O'Sullivan et al. [64] shows that non-visual interaction paradigms based on sonification can be used to such extent and we present two novel auditory guiding modes adopting sonification to guide users in road crossings in Chapter 4.

2.2 Access to images and didactics

Access to images is fundamental in learning STEM subjects like maths [8]. For example, images are often used to illustrate concepts like direction, quantity, shape and slope in an holistic manner [84]. Traditionally, access to graphical education material is provided by tactile drawings. However, tactile drawings suffer the same limitations introduced in Section 2.1 about tactile maps: a limited amount of information can be displayed on the map and there may be subjects that do not read Braille and therefore have no access to these labels.

In Section 2.1, we described a solution adopting non-visual interaction paradigms based on sonification to support orientation of people with VIB. Sonification has also been adopted to address the problem of enabling visually impaired persons to understand graphical information. Sanz et al. [74] and Sarkar et al. [75] present comprehensive surveys of sonification systems used to represent visual scenes and bi-dimensional images. Two kinds of sonification techniques can be adopted to support the study of STEM subjects: techniques based on "Auditory Icons", which associate images to metaphorical sounds and those based on a "Parameter Mapping" approach, where image features are mapped to sound attributes like, for example, volume and pitch. Yeo and Berger [93] created a framework for designing image sonification methods, categorizing various aspects of the sonification process. The authors point out the difference between two methods to organize data for auditory display: scanning and probing. In the scanning method, image data is scheduled to be sonified in a predefined order. Differently, in the probing method, the user can interactively change the portion of the image to be sonified.

A technique following the scanning method is proposed by Dallas and Erickson [22]. The technique adopts the parameter mapping approach, generating sound by associating the vertical position of each pixel of an image to frequency, the horizontal position to time and brightness to loudness. This technique is adopted in "the vOICe project" [60] that aims to enable people with VIB to explore frames captured through a camera. A conceptually similar solution proposed by Abboud et al. is called "EyeMusic" [1]. Yoshida [94] adopts the probing method to allow exploration of images on a touchscreen device by using two different sonification modes: local area sonification and distance-to-edge sonification. In the first mode, when the person slides the finger over an edge, a sound representing the line is played. In the second mode, a pulse train signal is used to represent the finger's distance to the closest edge. All of these techniques present some drawbacks. In particular, as highlighted by S. Maidenbaum, a long and arduous training is necessary to become proficient in exploring generic images [53]. However, for certain types of images like binary drawings, simplified techniques may be adopted with the aim of making users proficient even when little or no training is provided. In Section 6.3.6 we propose and evaluate six novel sonification techniques to explore binary drawings that are effective after a few minutes training.

Parameter mapping-based sonification techniques are adopted to represent function diagrams through sound. Gardner et al. [27] introduce "Audio Graphing Calculator", a desktop application that describes a function diagram through a simple sound. Walker et al. [87] propose "Sonification Sandbox", a desktop application that describes a function diagram through a synthetic sound. Sonification enables blind students to understand the trend of the graph and relevant points such as maxima, minima and intersections [19]. Unfortunately, no quantitative information is straightforwardly provided by these programs while a blind student is exploring the curve. Moreover, the sound feedback is not enough to convey additional information like, for example, the asymptotic behavior of a function (e.g. to find out an horizontal asymptote) and concavity in a given interval. We address these issues in Section 6.2 by introducing "Audio Functions", a mobile application that adopts a probing approach to sonify function graphs and their properties.

2.3 Large Scale Evaluation

Understanding user behavior during interactions with a software application is of paramount importance for evaluating the application's effectiveness, for guiding the iterative design process, and for informing the design of similar applications. However, there are inherent challenges in conducting behavioral studies both over long periods and with large samples of participants with disability. Indeed, as shown by Petrie et al. [69], it is difficult to find sufficient numbers of target users and bring them to a facility to take part in usability evaluations. Also, it may be cumbersome to recruit a sufficiently diverse sample of test subjects to account for different abilities. We experienced such issues in some user evaluations presented in this dissertation like, for example, the one described in Subsection 4.6.2 where finding and involving 12 test subjects required a very large effort.

In the broader field of human computer interaction, instrumented remote evaluations [33] are adopted to address these issues by collecting usage data while users are interacting with a software application in the users' normal environment (e.g. at home or at office) and analyzing them offline. We adopted this methodology to evaluate six novel sonification techniques (see Subsection 6.3.6). However, even if the adoption of an instrumented remote evaluation simplified the conduction of our tests, a large effort was still required in order to identify and reach appropriate test subjects. A solution to this issue could be to recruit test subjects and administer instrumented remote evaluations using online labor markets like Amazon Mechanical Turk [46]. Accessibility researchers have already shown that online labor markets can be used to provide human-powered assistive solutions [12]. An example is "VizWiz", a mobile assistive application to crowdsource answers to visual questions from sighted workers [11]. However, while it is known that people with disabilities participate in online labor markets as workers [96], it is not known, to the best of our knowledge, how many test subjects with VIB can be recruited on these platforms.

A common practice in human computer interaction research is to collect and analyze real-world usage data in order to evaluate software applications. However, not many contributions in the field of assistive technologies adopt methodologies involving the analysis of collected real-world usage data. To name a few, "Webinsitu", a contribution by Bigham et al. [10] where user actions automatically collected during web browsing are used to assess the accessibility of web pages by visually impaired users. Usage log analysis is also performed by Nakajima et al. [62] to evaluate the localization error of a navigation assistance tool using Video Light Communication (VLC) for guiding people with visual impairments. Hurst et al. [35] used log data from real-world tasks over a long period to build predictive models in distinguishing users by pointing performance. In a work by Riboni et al. [72], behavior anomalies perceived during user interaction with a sensor-enabled smart home environment act as a diagnostic tool for detecting mild cognitive impairments in senior patients. In Chapter 7 we describe the methodology we adopted to collect and analyze real-world usage data from "iMove", a mobile application that supports orientation and mobility of people with VIB. The analysis allowed to pinpoint a number of usage properties of the application, cluster users depending on their usage patterns and identify improvements to be introduced in future versions of the application.

Chapter 3

Supporting orientation with Audio-Tactile Maps

It is common for people with VIB to explore a new location (such as a work environment or a university campus) when there is little other traffic present, in an effort to develop a mental map of the location; the spatial model developed from gathering this experiential knowledge is a cognitive map [47]. Developing a cognitive map is a challenging task, as it requires to pay attention to a variety of proprioceptive, tactile and acoustic cues. It has been shown [44, 64] that the nature of sound reflections in a space can give valuable acoustic cues for orientation and that people with VIB use self-produced sounds such as finger-snaps and footsteps to learn about their surroundings by listening to the reverberant feedback from the space. Acoustic cues are also provided by environmental sounds like, for example, noise coming from vent shafts or from a nearby street. Tactile maps can be adopted to understand the spatial features of an environment before journeying but, due to the lack of acoustic cues, cannot substitute "in-situ" exploration. The "Audio Tactile Map" (ATMap) project aims to address this type of scenario by providing a virtual reality tool which can help a visually impaired individual to form a cognitive map of a location remotely, before visiting the physical site. This is achieved by supplementing a paper tactile map with an interactive auditory display of the target environment, featuring verbalized information and simulations of characteristic acoustical features. In the next section we introduce the ATMap prototype, a system that builds on the previous experience by O'Sullivan et al. [64], refining the proposed auditory display and defining two novel user interfaces. Finally, we describe the preliminary evaluation of the prototype involving 5 subjects with VIB.

3.1 The ATMap prototype

ATMap is a prototype software system that provides an auditory display for users interacting with a paper tactile map. Two forms of the system interface have been created as desktop and mobile versions. The first of these is designed as a larger-format information point; the movement of a user's hands are tracked over the tactile paper map with selections being made through large push-buttons. The use of comparatively inexpensive camera technology has kept

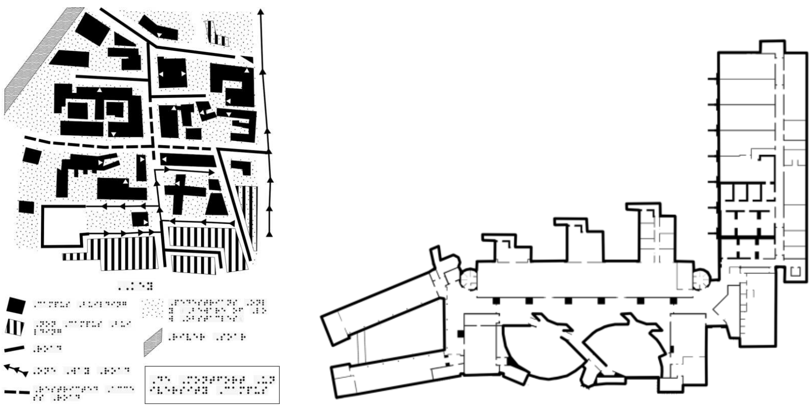

(a) DMU Campus map. (b) Queen's Building map.

FIGURE 3.1: Tactile maps used with the ATMap prototype system.

the cost of this system low while allowing maps to be used up to A3 paper size. In addition, this technology has the potential for tracking interaction with objects other than low-relief paper tactile maps. The mobile interface uses a tactile paper overlay on a multi-touch tablet to sense user interaction. The ubiquity and quality of such devices makes their use in the deployment of the ATMap worthy of investigation, despite the smaller sizes of maps that can be produced at reasonable cost. It must be stressed that although touted as a mobile interface, this latter version is still designed to assist in the cognitive map training rather than as an in-navigational tool to be carried and used while in transit. Both interfaces communicate with the system software running on a computer, but a self-contained application which runs on the iOS operating system for Apple mobile devices has also recently been prototyped.

The first map used with the system is a portion of the campus of De Montfort University (DMU) in Leicester, including buildings, streets and the nearby river. As shown in Figure 3.1(a), various textures indicate specific features and a key written in Braille is provided at the bottom of the map. A second map of a building within this campus has also been used. The map shown in Figure 3.1(b) shows the first floor of the Queen's Building, with portions of the ground floor and main ground floor entrance also visible. Dark blocks are used to indicate interior spaces without any additional use of a texture key. These digital map images are printed on swell paper using a regular office printer and are then passed through the fuser oven. Areas of dark print absorb heat and become raised, generating a textured surface as shown. The fuser has a heat setting which determines the height of the raised elements.

3.1.1 Software overview

To facilitate rapid-prototyping while maintaining flexibility, the ATMap software was written in the Java programming language. A digitised map and data files are loaded and the map is analyzed using an image processing module to automatically extract regions of interest of arbitrary shape. In the DMU campus map example, each of the buildings is segmented and labelled. Labels are combined with metadata, such as the building name, and any associated audio files for that map location. At runtime, a software zone is specified for each building; tracking data is projected onto the map coordinates and a zone returns its information when selected. The system has a graphical user interface (GUI) that displays information for sighted individuals and provides administrator access to system settings and metadata. ATMap was developed to allow two modes of use by tracking the interactions of the user with the paper tactile map. The first mode, called "exploration", presents the user with information about specific landmarks and can be used to find out about buildings, roads or other key features of the map. The second mode, "navigation" presents the user an interactive, sequential route between two chosen landmarks. Intermediate waypoints are provided which describe nearby landmarks and features, helping to create a cognitive map of the route.

3.1.2 Auditory Display

The ATMap system delivers spatial audio over headphones (but with mono compatibility for loudspeaker playback) and uses audio in two ways; text-to-speech synthesis of map metadata and sounds of characteristic acoustical features of spaces. Interaction with the map produces audio feedback and the nature of sounds produced is context sensitive. When a user selects a map feature, the information contained in the associated map zone is rendered using a basic text-to-speech engine. For example, when a building is selected on the ATMap of the DMU campus, its name is synthetically spoken first, followed by any additional stored information. Audio is also provided in the form of environmental and self-produced sounds. Examples of the former are the background sounds positioned in the DMU campus map; the sound of the river can be heard at outdoor locations towards the top left of the map, for example. Some interior spaces on the Queen's building map have recordings of self-produced sounds, such as hand-claps, finger-clicks and footsteps, embedded at various locations. All sounds were recorded using binaural microphones at the associated physical locations, as this allows for the reproduction of realistic 3D sound-fields using a pair of headphones [32]. This type of auditory display preserves the acoustic cues found to be useful for navigation in physical spaces by people with VIB.

3.1.3 User Interfaces

Desktop Interface

The mounted paper tactile map is shown in Figure 3.2. A "Leap Motion" device is positioned above the map and is used to track the movements of hands on the tactile paper. The Leap Motion is an inexpensive consumer device and is finding popular use in interactive applications as a free air gesture controller. It comes with a suitable Application Programming Interface (API) to allow rapid prototyping and code integration. The device has a pair of cameras and illumination technology contained within a small form factor enclosure, making it unobtrusive and well-suited to the current application. The coordinates and orientations of hands, fingers and tools in view of the device are sent to a computer over a Universal Serial Bus (USB) cable connection. Preliminary testing of the interface with a visually impaired user participating in the early stages of the system development has previously been described [64]. The observations made were generally positive and encouraged further development of the system, including implementation of new functionality and/or modifications to existing features. For example, the default low quality TTS engine was replaced and audio files were instead pre-rendered using

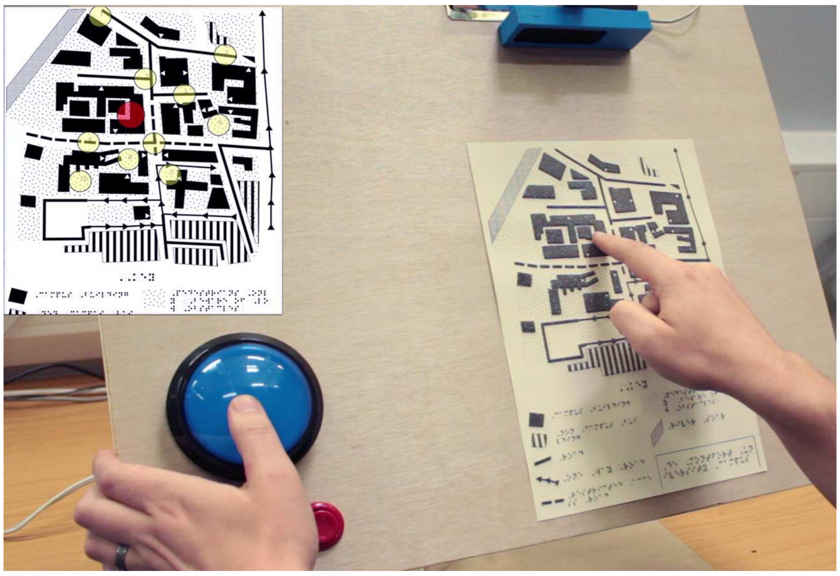
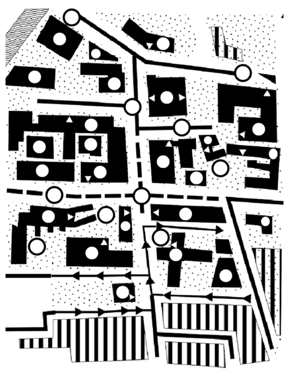


Figure 3.2: The hardware setup for the desktop interface of ATMap, consisting in the leap motion device, the tactile map and three push buttons mounted on a wooden board.

the AT&T Natural Voices® Text-to-Speech before being loaded into the system. This improved the discernibility of delivered information and also made the ATMap more pleasant to use. Large electronic push-buttons (visible in Figure 3.2) were another addition, providing robust and reliable selection events. These are interfaced with the host computer using an Arduino microcontroller over a USB serial connection. However, in testing, it was found that the Leap Motion did not always provide robust tracking in an uncontrolled environment and required repeated set-up and calibration. Using the device to track hands which were in contact with a surface was problematic. In lieu of this, an alternative approach was undertaken to allow the use of a mobile tablet interface and further test the system software. This also allowed investigating the feasibility of deploying the system on mobile platforms and widening the potential user base.

Mobile Interface

The mobile interface is designed to use a tactile paper overlay on a touchscreen device. Affordable mobile tablets do not have screen sizes that would allow the production of larger tactile maps (i.e. A4 paper size and larger), but such devices are now so commonplace that it was deemed necessary to investigate their use as the sensing component of an ATMap. An iOS application was therefore developed that sends control information (currently from a single point of touch) to the system software. The messaging format used Open Sound Control (OSC), which is an effective means of sending control information via User Datagram Protocol (UDP) over a network [91]. Adoption of this popular communications approach allows easy integration with other systems, as many OSC clients exist for common development environments as well as code libraries for standard programming languages (e.g. Java, C++). A module for receiving and parsing OSC messages was therefore integrated into the existing ATMap software. In light of the experiences of Brock et al. [16], a mono-touch interaction mechanism using single-and double-tap selection gestures was implemented. The mobile version of the interface described above has been used to evaluate the usability of the ATMap, the results of which are presented in the following sections. Development on the project has now progressed to include the full ATMap software system in a


(a) Redesigned DMU Campus map.

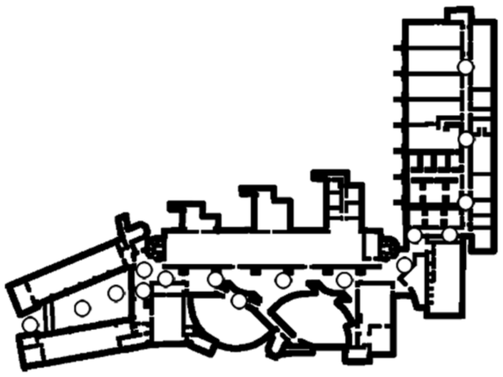

(b) Redesigned Queen's Building map.

Figure 3.3: Tactile maps redesigned for the evaluation of the mobile interface. Circles represent the location of auditory icons.

mobile application for the iOS platform. However, this integrated solution has not been formally evaluated.

Adapting maps for the Mobile Interface

The tactile paper was found to be suitable for use as an overlay for an Apple iPad touchscreen only with some modifications to the designs of the maps. Using higher settings of the fuser oven results in a more raised surface of the tactile paper, preventing the screen from detecting a touch through its capacitive sensing mechanism. To circumvent this issue, the maps were redesigned to contain voided areas within tactile features e.g. as buildings were represented as dark- printed raised blocks, a light area placed within would be rendered as a flat region. This had the advantage of specifying a discernible area which acted as the selection point for a feature of interest. In Figure 3.3(a), the modified version of the DMU campus map shows the circular voids (in white) which were designed to be easily identifiable among the rectangular buildings. Touching at these areas produced reliable touch coordinate and gesture information. As this map is of buildings from an exterior view, it was also possible to use circles to indicate the locations of outdoor auditory icons containing environment sounds. The Queen's Building map shown in Figure 3.3(b) necessitated a different design as it depicts interior spaces. The use of voided circles for both room information and auditory icons made the interface confusing for users, so circles were only employed as icons containing both environmental and self-produced sounds. The selection gesture made at these locations determined the sound played back: a single tap played environment sounds and a double tap played the self-produced sounds. The rooms themselves are bounded by thick outlines; making a section gesture anywhere within a space returned the associated information.

The ATMPad Application

The original ATMap prototype is comprised of different components: a tactile paper map, a Leap Motion camera and electronic circuit, with a laptop running the system software. As previously discussed, use of the Leap Motion was found to require some set-up and that the

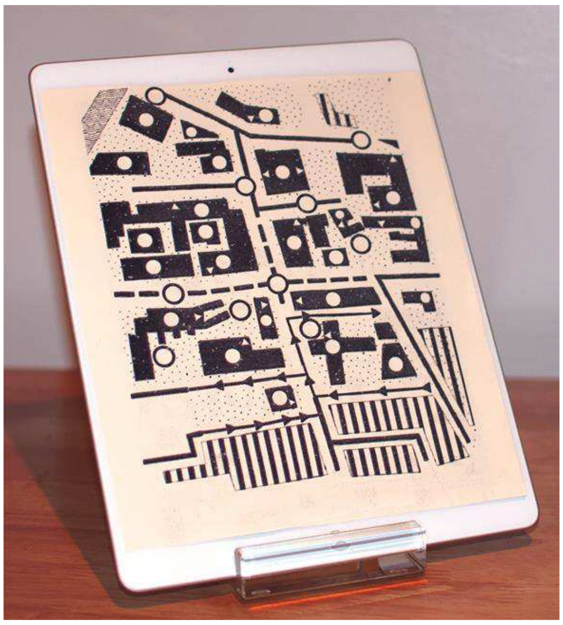


FIGURE 3.4: Apple iPad with tactile map overlay.

sensor be calibrated occasionally. The mobile version of the interface communicates with the system software over a network. In order to explore the feasibility of a simpler architecture, a second prototype called ATMPad has been developed as a tablet application for the iOS platform. The application displays a map of the current environment and detects touches on areas of interest. Auditory feedback is then provided through the internal speaker or via wired or Bluetooth headphones. Different environment representations can be loaded by pointing the tablet camera towards a QR code printed on a tactile map, which may then be placed over the tablet screen as shown in Figure 3.4.

3.2 Evaluation of mobile interface

From the beginning, the ATMap project was concerned with the impact on the blind and visually impaired community and with the usability of the developed tool. For this reason, the involvement of blind and visually impaired individuals was necessary as early as possible both in the design and implementation of the ATMap system, and in the preliminary evaluation of the prototype [64]. The testing stage presented in the following sections was therefore important not only to technically ensure that the system was working adequately, but also to gain feedback on the usability of the application and on its suitability for the use by people with VIB. These first tests are focused on the exploration mode of the ATMap.

3.2.1 Experimental methodology

A total of 5 visually impaired individuals (3 completely blind, and 2 with residual vision) carried out the test, 2 of whom were females. Testing was carried out in November/December 2014 on one open environment map (the DMU Campus map) and one closed environment map (the

Environment	Selection Gesture	Auditory Display
DMU Campus Centre	Single-tap on circle within building	Mono text-to-speech with location information
	Single-tap on circle on the roads/outdoors	Binaural location recording
Queen's Building	Single-tap on any location inside building	Mono text-to-speech with location information
	Single-tap on circle inside/outside building	Binaural location recording (environmental/background sound)
	Double-tap on circle inside/outside building	Binaural location recording of finger snapping sound

Table 3.1: Selection gestures and associated audio feedback used in the evaluation of the ATMap mobile interface.

Queens Building map). The touch gestures used to interact with the maps were as shown in Table 3.1.

The evaluation was structured in the following way:

- Brief introduction to the ATMap project and system (5 minutes).
- Exploration of the open environment map (15 minutes).
- Exploration of the closed environment map (15 minutes).
- System Usability Scale questionnaire (5 minutes).
- Computer System Usability questionnaire (5 minutes).
- Interview (5 minutes).

To navigate the map the individuals used an Apple iPad over which the swell paper map was positioned and which communicated with the ATMap software over a wireless network, as previously described. The individuals wore a pair of Beyerdynamic DT770 headphones for the delivery of both mono and binaural signals.

3.2.2 Questionnaires

Two questionnaires were presented to the individuals after the exploration stage. These were the System Usability Scale (SUS) questionnaire [17] and the IBM Computer Usability Satisfaction questionnaire [50]. Users were asked to rate their levels of agreement with a series of statements regarding use of the system. Note that statements have been shortened in the figures presented below to aid illustration, but these were presented to subjects with strict adherence to recommended wordings.

For the SUS questionnaire, the possible answers to each question could go from 1 ("Strongly Disagree") to 5 ("Strongly Agree"). For the odd-numbered statements, a high number corresponded to a positive feedback on the system, while for the even-numbered statements it was the inverse. For this reason, the statements have been re-ordered in Figure 3.5(a) (with statement order, moving downwards; 1-3-5-7-9-2-4-6-8-10). Considering the odd-numbered statements, the mean response value was 4.04 (±1.27 standard deviation), while for the even-numbered statements the mean response value was 1.6 (±1.04 standard deviation).

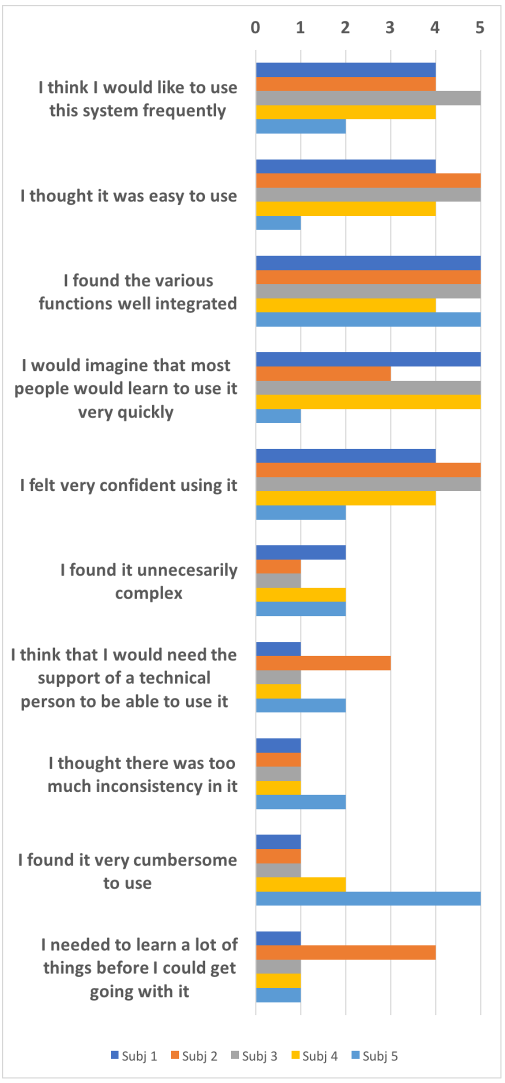
For the Computer Usability Questionnaire the possible responses to each statement could range from 1 ("Strongly Disagree") to 7 ("Strongly Agree"), with the possibility of selecting the value 0 for answers which did not apply to the currently evaluated system ("n/a"). In this case, for all statements a higher response value corresponded to a more positive feedback on the system. Seven statements from the original questionnaire did not seem to apply to the ATMap system (at least 60% of the subjects answered "n/a"), and were therefore removed from the analysis. The responses of each subject to each of the 12 statements are reported in Figure 3.5(b). The mean response value across all subjects was 6.39 (±0.86 standard deviation). The IBM questionnaire also gave the opportunity for each subject to report up to three positive notes regarding the system, and three negative ones.

Positive comments

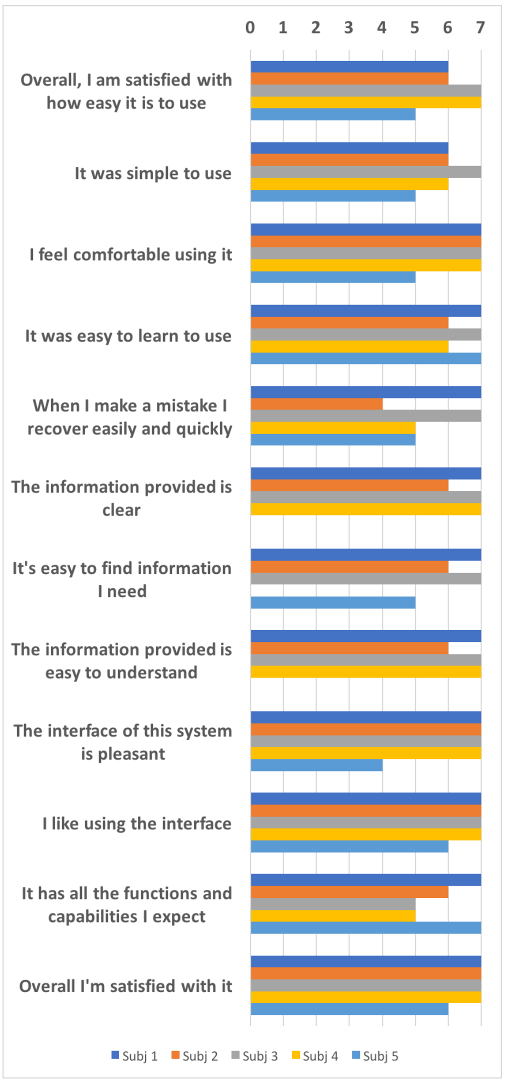
- *It's a very good system, especially if someone was coming new to the place and had very little or no vision.*
- *Very interesting system, which could potentially be very useful. I would like now to try to go to the actual location of the map and see how much I've learnt.*
- *The audio (speech and sounds) was pleasant and informative.*
- *Very informative and easy to use.*
- *I feel that I really know the environments.*
- *Interesting approach to learning maps before going in the place.*
- *Potentially useful also when you are in the place, maybe with guidance while walking.*
- *The maps are clear, but mainly using my residual vision. The tactile design is clear though.*
- *The circular features works quite well when feeling them, even though apparently they are too small for the touch action.*

Negative comments

- *Some of the circles were very insensitive, while some were hypersensitive...the level of sensitivity needs to be more equal.*
- *It might work better if the whole circles were raised (like button press).*
- *Some difficulties in starting the audio playback.*
- *Audio playback locations were limited in number.*
- *Start play of the click noise is difficult to activate.*
- *Takes some time to get used to navigating the map.*



(a) Responses to the SUS questionnaire.



(b) Responses to the IBM questionnaire.

FIGURE 3.5: Subject responses for the two administered questionnaires (0 stands for “n/a”).

- *The environmental sound couldn’t be stopped, once when you knew what they were, and they were overlaying the other sounds when playing.*
- *It would be nice to know what the streets were...names of the streets.*
- *The responsiveness is an issue...either related with the iPad or the touch response, but it is sluggish.*

3.2.3 Interview

A brief interview was carried out after the two questionnaires. The subjects were all asked four questions, which are reported here followed by a brief summary of the responses.

Did you find the non-speech audio informative? All subjects did, except for Subject 5, who reported that for him it was not particularly informative, but that they could potentially be informative for visually impaired individuals with no residual vision (it is important to note that Subject 5 was one of the two subjects with residual vision who did this evaluation).

Did you find both finger snap and background noise informative? Again, except for Subject 5 all subject found the two types of audio feedback informative. Three subjects clearly preferred the background noise, and one the click noise.

What features would you like to see in future versions of the system? Here follows a list of the answers given to this question:

- *Ability to program two locations and give guidance description on how to move from one to the other.*
- *Use buttons instead of circles.*
- *Considering the answer of buttons, the possibility of using an extra button (not on the map) with the other hand for triggering sounds could be interesting. Another option could be to change the textures in the circles (maybe little dots).*
- *Simpler selection of the sound to play back, and possibly more recordings of noise and clicks in different locations. Maybe an error report for when the playback is not triggered properly.*
- *More audio playback areas. Maybe also interactive audio playback, which changes as I move in the environment.*
- *More interactivity, maybe sound which is changed continuously while moving in the environment.*
- *Surely simpler trigger for the click noise.*
- *Audio playback stop, and better audio quality.*
- *Where the crossing is in Mill Lane, for example, it'd be useful to know that that is a crossing.*
- *Add more tactile and audio description features regarding, for example, roads, etc.*

As can be observed, some of these responses refer to features which have been developed for the ATMap system, but which were not evaluated in this stage (e.g. the use of push buttons to select features). Other responses are related to potential improvements that could be implemented in a further version of the ATMap system.

Do you think the system is useful in forming a mental map of the associated area prior to visiting it? Two of the five subjects had a good knowledge of both the DMU Campus and the Queen's Building configuration, but reported that the ATMap system could potentially be "extremely useful" for learning the configuration of environments they did not already know. Two of the subjects who were not familiar with the real environments reported that they felt they had a good mental representation of the environments after navigating them through the ATMap system. One of the subjects reported that he was not sure about this, but that he would have liked to have the possibility to navigate the real environments and verify if he could remember something from the ATMap navigation.

3.3 Discussion

In general, considering the results of both questionnaires and of the interviews, the system was very well received. Relevant feedback was gathered regarding technical issues, problems and potential improvements for the ATMap system. As reported in the previous sections, Subject 5 gave lower scores overall and had more critical comments on the usability of the system. The feedback of all subjects was nevertheless constructive and often addressed real technical problems that are being resolved before the deployment of future prototypes and evaluation stages.

Tactile surfaces may be produced using inexpensive heat-reactive paper, providing touchable information display for a variety of applications. The addition of inexpensive sensing technology can add interactivity through a computerized system with potential for multimodal information delivery. The delivery of information through audio and tactile channels is of benefit to people with VIB and this first application is designed to help in the formation of a cognitive spatial map of a location prior to journeying. The auditory display provides both verbalized information and environmental sounds for map locations. Previous research has shown that these can be beneficial for the formation of cognitive maps in visually impaired individuals and can aid in navigation. In addition, advanced auditory display can present audio recordings that preserve some of the acoustic cues used in psychoacoustic and spatial cognition.

3.4 Summary and future directions

In this chapter we present the ATMap prototype system, which extends research presented in a previous work [64] highlighting three main contributions. First, a novel auditory display to augment tactile maps is introduced. The solution adopts verbalized information and simulations of characteristic acoustical features to convey additional information about the environment. Indeed, the adoption of an auditory display makes it possible to represent a larger amount of information on the map if compared with approaches based on Braille labels, addressing a known issue highlighted by previous research [82]. Also, by using binaural recordings of environmental noise and self produced sounds, the system provides valuable acoustic cues to support the development of cognitive maps of environments. A second contribution consists in the fact that the ATMap system tracks interaction over traditional tactile maps, letting users perceive direct stimulus from tactile relief, which makes it possible to better understand the represented environment, if compared with previous contributions adopting touch pads and touch grids [66, 40, 76]. Finally, a preliminary experimental evaluation conducted with 5 subjects with VIB shows that the system is well received by users and highlights some issues of the current approach.

Future research directions include proposals for system improvements and new evaluations. The auditory display should be improved in order to introduce the real-time acoustical rendering of sounds during map exploration. Such functionality should adopt head tracking facilities to provide realistic binaural rendering and implement an approach based on HRTF to provide accurate sound spatialization. Also, the usability of the ATMap system should be improved in order to make its desktop interface more intuitive. For what concerns the mobile interface, two possible improvements may be introduced: first, accelerometer input may be used to improve the robustness of selection gestures in the absence of pressure sensing multi-touch devices; second, magnetometer input may be used to determine the direction pointed by the user and adapt the acoustical rendering.

For what concern evaluations, we should perform a quantitative study comparing how people with VIB orient themselves in the environment after learning its structure using standard tactile maps or tactile maps augmented by ATMap. Performance may be measured by asking subjects to rebuild the shape of the environment with plastic bricks or by asking them to perform some navigation tasks directly in the environment.

Chapter 4

Recognizing zebra crossings

In Chapter 3 we introduced a solution to support orientation by enabling the development of a cognitive map of an environment before journeying. However, knowing the characteristics of an environment and how to navigate it is not enough to provide safe journeying. There are many difficulties like, for example, to avoid obstacles along the way (e.g., people on the sidewalk, trash bins, poles, etc.), to find a target (e.g., stairs, doors, intersections, etc.) and to get information reported on pedestrian signs (e.g., crossing a road over a zebra crossing when the traffic light is green, etc.).

In this chapter we describe our efforts in developing a mobile application called *ZebraX* that addresses the problem of supporting a person with VIB while crossing the street. ZebraX detects zebra crossings, a common type of pedestrian crossings (See Figure 4.2 for an example) in images captured with the mobile device camera and provides guidance to perform the actual crossing.

There are a number of challenges involved with the identification of pedestrian crossings. First, given the hazards inherently connected with road crossing, it is crucial to have no false positives, i.e., to erroneously recognize a crossing in an image that actually contains none. At the same time, in order to guarantee an effective solution, most pedestrian crossings should be properly identified. Second, it is necessary to precisely compute the relative position between the user and the pedestrian crossing. Third, since the application should be responsive, the identification process should have a low execution time. Fourth, guidance information should be provided without overloading the auditory channel, thus allowing users to focus on important environmental sounds like the one of an approaching car.

4.1 The ZebraX application

The ZebraX application is divided into three main modules, as depicted in Figure 4.1. The *Recognizer* module implements the "ZebraRecognizer" algorithm [3] which identifies zebra crossings in an image and computes the relative distance between the user and the zebra crossing. Starting from the positioning data computed by the Recognizer module, the *Logic* module computes the messages that are to be conveyed to the user. The Logic module is also in charge of keeping distance quantities updated by using gyroscope readings. Finally, the *Navigator* module is in

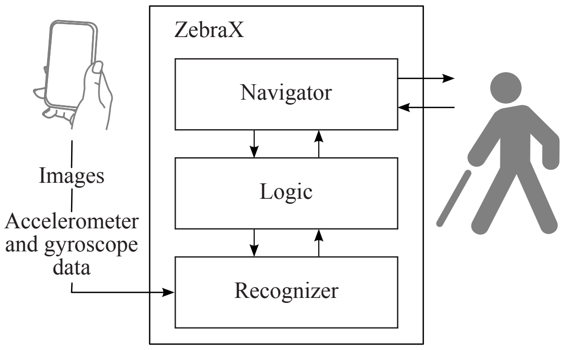
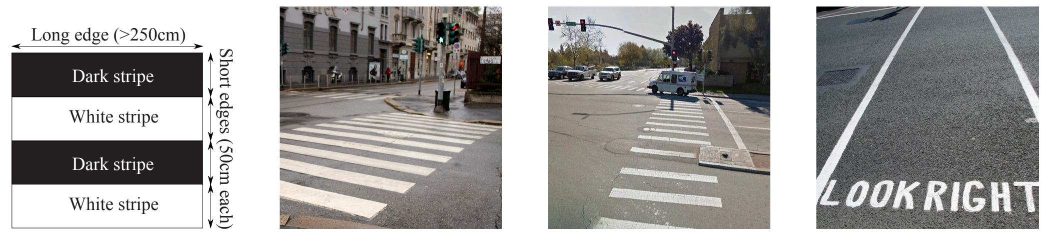


FIGURE 4.1: Modules of the ZebraX application.



(a) Zebra crossing pattern in Italy. (b) Pedestrian crossing in Italy. (c) Pedestrian crossing in USA. (d) "Two lines crossing" in the UK.

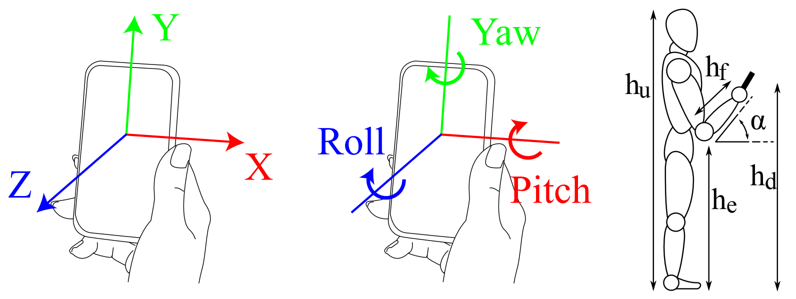
FIGURE 4.2: Examples of zebra crossings.

charge of conveying audio instructions to guide the user towards and along the zebra crossing. The main challenge in Navigator is that the user needs to be continuously informed about his/her position with respect to the zebra crossing. However a person with visual impairment or blindness should not be overwhelmed with too many audio messages, because they can divert the attention from the surrounding audio scenario, which is essential to acquire indispensable information (e.g., an approaching car, a person walking by, etc.).

4.2 The Recognizer module

The Recognizer module implements a computer vision technique tuned for detecting zebra crossings as defined by Italian traffic regulations (see Figure 4.2(a)), but it can be easily adapted to most definitions used worldwide. For example, given the similarity between Italian zebra crossings (Figure 4.2(b)) and the US version (see Figure 4.2(c)), it is possible to reconfigure Recognizer to recognize the US zebra crossings with very limited effort, by setting and re-tuning the detection parameters. Clearly, the solution does not directly apply to other types of pedestrian crossings, like the "two lines crossings" (see Figure 4.2(d)). Still, the adopted methodology can be used to design similar solutions for other pedestrian crossings or other kinds of geometrically well-known horizontal traffic signs.

A zebra crossing is described as a horizontal traffic sign consisting of an alternating pattern of dark and light stripes (see Figure 4.2(a)). It is composed of at least 2 light stripes and 1 dark stripe. The stripes are commonly rectangular and, less frequently, in case of diagonal crossings, parallelograms. They are 50cm thick and have a width of at least 250cm. The dark stripes are



(a) Device axes. (b) Rotations. (c) Height.

FIGURE 4.3: Rotation and position of the mobile device while using ZebraX.

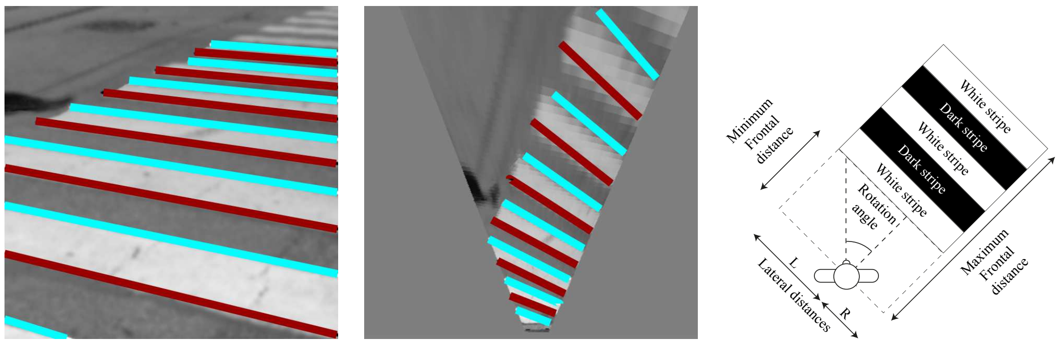
of the same color of the underlying road while the light stripes may be white or, when the road is undergoing repairs, yellow.

The recognition process is entirely computed locally on the mobile device because the responsiveness requirements of ZebraX make it impractical to have a remote computation due to network latency. For the detection of zebra crossings, Recognizer relies on data sources available on off-the-shelf smartphones: video camera, accelerometer and gyroscope. The first captures video frames that can then be analyzed with computer vision techniques in order to detect zebra crossings. Accelerometer and gyroscope, instead, can be used to extract the orientation of the device with respect to the ground plane and the detected crossings.

Technically, the input of Recognizer consists of the user's height h_u, an image i with height i_h and width i_w and the gravity acceleration data represented as a three dimensional unit vector $a = \langle a_x, a_y, a_z \rangle$. Its elements a_x, a_y and a_z are measured in $g = 9.80665$ m/s^2, take values in $[-1, 1]$ and represent, respectively, the portion of the gravity that is applied on the device x, y and z axes (see Figure 4.3(a)).

The output of the algorithm is the most suitable detected zebra crossing, if any. It is characterized by a list of stripes, each one defined by its top and bottom line segments and its color (i.e., black or white). We represent the position of each line segment both in the source image (e.g., Figure 4.4(a)) and on the rectified ground plane (e.g., Figure 4.4(b)). The result also includes five compact and easy-to-use distance measurements (see Figure 4.4(c)). "Rotation angle" is the angular distance between the user's heading and the line perpendicular to the stripes. "Minimum frontal distance" ("maximum frontal distance", respectively) is the distance between the user and the closest (farthest, respectively) stripe. Finally, "lateral distance left" ("lateral distance right", respectively) is the distance between the user and the left (right, respectively) border of the crosswalk.

The Recognizer module is internally divided into 6 steps, as shown in Figure 4.5. The first three steps (rectification matrix computation, image pre-processing and line segments detection) are all aimed at extracting the line segments that represent the stripes. In the last three steps (line segments grouping, zebra crossing validation and final result computation) the line segments are processed and the user's relative position with respect to the crossing is computed.



(a) Line segments in the source image. (b) Line segments in the rectified ground plane. (c) Relative distances between user and zebra crossing.

FIGURE 4.4: Zebra crossing identification and relative distances.

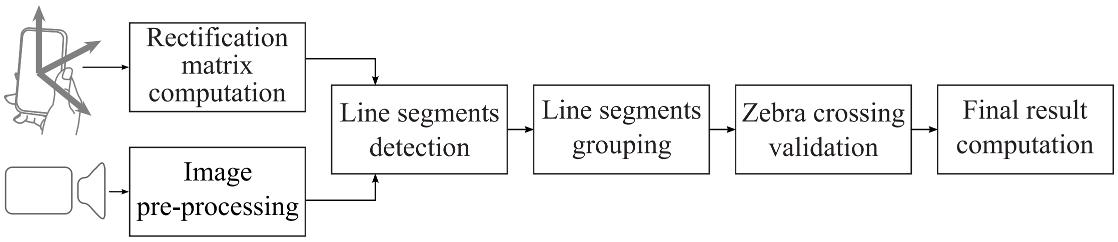


FIGURE 4.5: ZebraRecognizer flowchart.

4.2.1 Geometric operations

Horizon computation

Horizon computation is a prerequisite for many steps of the Recognizer module like, for example, the Rectification Matrix Computation step we are about to introduce. ZebraX uses accelerometer and gyroscope data to compute the equation of the horizon line in the image reference system. The computation is based on Property 1 (proofs of formal results are reported in our previous work [55]).

Property 1. Let ρ and θ be the device pitch and roll angles respectively, $C = \langle C_x, C_y \rangle$ is the center of the image and f is the focal distance of the camera (in pixels). Then, the equation of the horizon line h inside the acquired image is:

$$\sin(\theta)x + \cos(\theta)y - \sin(\theta)(C_x + \tan(\rho)\sin(\theta)f) - \cos(\theta)(C_y + \tan(\rho)\cos(\theta)f) = 0 \qquad (4.1)$$

Rectification Matrix Computation

Planar rectification is a homography, represented by a 3×3 rectification matrix, that removes the projective distortions from the image of a planar surface and returns a view of the same plane in which the camera's axis is perpendicular to the plane. For the ground plane, the rectified image is a view from directly above it, as seen in Figure 4.6. Once the rectification matrix is known, it can be selectively applied to some elements (i.e., line segment end points) instead of the whole image, thus reducing the execution time. In our previous work [3], the rectification matrix is obtained multiplying the affine rectification matrix, determined using the approach proposed by

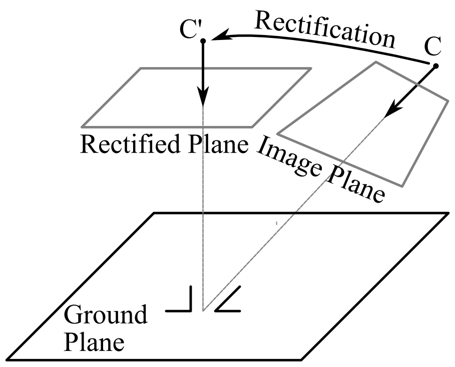


FIGURE 4.6: Rectification homografy

Liebowitz and Zisserman [52], with an approximated metric rectification matrix computed using a custom technique. The custom technique has been adopted because the original solution from Liebowitz and Zisserman requires prior knowledge of image properties, which is not available in our context. A more recent approach by Lefler et al. [48] shows how to compute the rectification matrix using the plane's vanishing line and the vertical vanishing point, which can be both computed using the gravity vector and the equation of the horizon line. We experimentally observed that the approach by Lefler et al. [48] yields better performance (mainly in terms of recall) and hence we decided to adopt it in Recognizer.

Note that the rectification matrix is computed for each new frame using the last available gravity data. Since on the system used for the experiments (iPhone 5S) gravity acceleration data is updated about 100 times per second, each time a new frame is received the rectification matrix is computed with values no older than 10ms.

The application of the rectification matrix to the image yields a "rectified plane" in which the distances are proportional to those on the ground plane. More specifically, the distance between any two points on the ground plane is equal to the distance of the corresponding points on the rectified plane multiplied by a *zoom factor*. To compute the zoom factor it is necessary to know the distance between any two points in the rectified plane as well as the distance between the corresponding two points in the ground plane.

In our case, we consider two artificial points on the rectified plane that are crafted in such a way that we can derive the distance of the two corresponding points in the ground plane thanks to the knowledge of the camera position in space and camera parameters. Property 2 shows how to derive the zoom factor (proof is reported in our previous work [55]).

Property 2. Let ρ be the device pitch angle, h_d the device's height, C the center of the image and f the focal distance of the camera (in pixels). R is the rectification matrix computed previously while A_i and B_i are arbitrary points below the horizon and that lie on line vl that is perpendicular to the horizon and that passes through the image principal point C.

Points $A_r = R \cdot A_i$ and $B_r = R \cdot B_i$ are rectified points corresponding to A_i and B_i respectively.

Then, the zoom factor z is:

$$z = \frac{h_d \cdot \left[tan\left(\pi - \rho - atan\left(\frac{CB_i}{f}\right)\right) - tan\left(\pi - \rho - atan\left(\frac{CA_i}{f}\right)\right)\right]}{A_r B_r} \tag{4.2}$$

Note that Property 2 assumes that the height h_d of the camera with respect to the ground is known. To estimate this value, Recognizer assumes that the user is holding the device in a position like the one depicted in Figure 4.3(c) in which the elbow is close to the hip and the forearm has an inclination of about $\pi/6$ with respect to the ground plane. By considering the proportions of the human body [36], the device height can be derived from the user's height h_u (either estimated or asked to the user). Indeed, on average, the height at elbow is $0.615 \cdot h_u$ and the forearm length is $0.205 \cdot h_u$. Consequently, the device height from the ground is estimated as:

$$h_d = 0.615 \cdot h_u + sin(\pi/6) \cdot 0.205 \cdot h_u \tag{4.3}$$

Clearly the above computation is subject to some approximation. However, the error does not practically affect navigation. For example, considering a 175cm tall person, the technique estimates that the device is held at 125cm from the ground. Even in the extreme case in which the device is actually kept at the height of the shoulders[1] (about 145cm from ground), a zebra crossing at a distance of 2m is computed as being 2.33m from the user i.e., the error is less than an average step length.

4.2.2 Image Pre-Processing

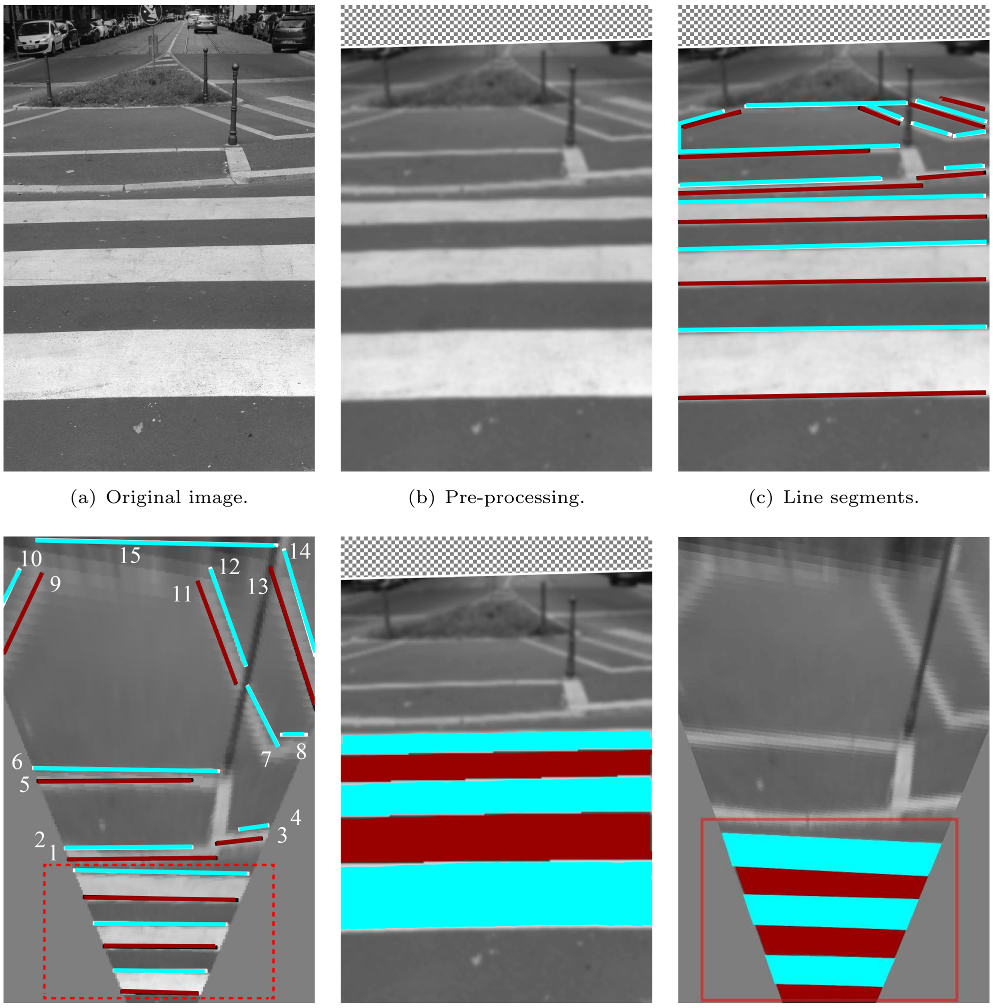
As observed, zebra crossings can be painted with different colors. Hence we are only interested in the light and dark components of the image. For this reason, we acquire grayscale images. Clearly, the use of single-channel images also helps improving the computation performance and reduces the memory footprint.

The acquired images contain many small details we are not interested in, such as cracks, paint imperfections, leaves and dirt. These imperfections may actually impair detection, hence we use resampling and blurring to filter them out. The first method rescales the image until small details become undetectable. Also, it reduces the image size and thus diminishes the execution time of per-pixel operations that follow. However, the size still has to be sufficient for a correct detection. As highlighted in our experiments (see Section 4.5), the best recognition results can be obtained at a relatively low resolution (i.e., 180×320). ZebraX acquires images at this resolution. Vice versa, the images in the test-sets were recorded at the resolution of 1080×1920 and resized, with a linear interpolation filter, before running each test so that Recognizer can be evaluated with images at different resolutions.

Finally, a Gaussian blur filter is applied to the image. Similarly to the resampling, the aim is to filter out imperfections in the image and ease the line segments detection. Since this step reduces the number of recognized line segments, it also indirectly affects the computation performances because fewer line segments need to be processed in the following steps.

Figure 4.7(b) shows an example of the pre-processing step applied to Figure 4.7(a) (the portion of the image above the horizon is ignored). Henceforth with "image" we intend the result of the pre-processing step.

[1]This is an unnatural position that we never observed during experiments.



(a) Original image. (b) Pre-processing. (c) Line segments.

(d) Line segment groups. (e) Crossing validation. (f) Detected crossing.

FIGURE 4.7: Main steps of Recognizer.

4.2.3 Line Segments Detection Algorithm

The line segments detection step is a modified version of the EDLines algorithm [6]. The input is composed by the pre-processed image, the horizon line and the rectification matrix. The output is a set of detected segments in the rectified coordinate system. There are four main differences with respect to the original algorithm.

First, our technique ignores the portion of the image above the horizon since no zebra crossings will ever be found there. This approach significantly reduces the computation time for two different reasons: it speeds up the line segments detection process itself and it reduces the number of detected segments, hence reducing the computation time of successive processing steps. This solution also helps improving the recognition accuracy as it prevents false positives (i.e., a false crossing recognized above the horizon).

The second difference with respect to the original EDLines algorithm is that our solution computes additional information about the detected line segments. First, in addition to gradient direction, our solution also computes the gradient orientation of the detected segments, so, in

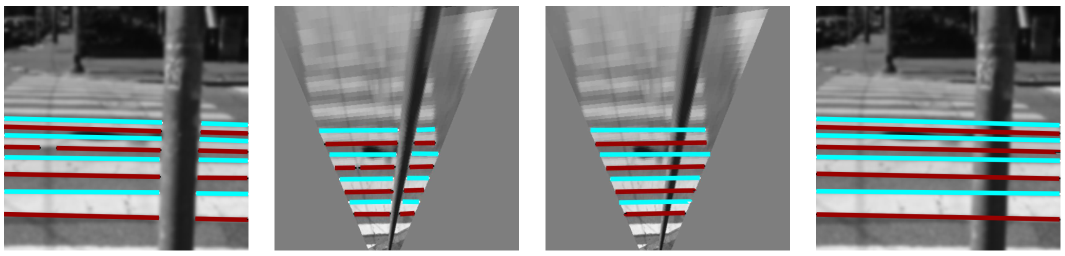

(a) Split line segments (projected). (b) Split line segments (rectified). (c) Merged line segments (rectified). (d) Merged line segments (projected).

FIGURE 4.8: Example of line segments merging.

practice, we compute the angle of the gradient in $[0, 2\pi)$ rather than in $[0, \pi)$. This information is useful in the following steps since the direction and the orientation of the gradient can differentiate between segments on the top and those on the bottom of each stripe. The second additional information computed by our version of EDLines is whether each end point of each line segment lies on the image boundary. This is useful, in the following computation, to distinguish between stripes that terminate in the end point position and those that, instead, can potentially continue but are not visible in the image.

The third difference is that our technique also merges close segments. Two segments having both slope distance and spatial distance lower than specified thresholds are merged. This is useful, for example, when two or more portions of a line segment have been recognized as different line segments due to minor imperfections in the image, noise, flawed coloration of the stripes or objects between the observer and stripes (Figure 4.8 shows an example). The line segment s resulting from the merging of two line segments $s1$ and $s2$ is computed as follows: first, the lines $l1$ and $l2$ on which the two line segments lay are calculated. Then, a new line l (equation in general form: $ax + by + c = 0$) is computed with parameters a, b and c being weighted averages (based on the two segments' lengths) of the corresponding parameters of lines $l1$ and $l2$. Finally, the segment s is computed as the union of the two line segments' projections on l.

In our previous solution [3], this merging operation was computed using line segments in their representation on the image, hence subject to projection distortion. Vice versa, in our current solution, line segments are rectified before being merged.

The fourth difference is that, during line segment computation, we use orthogonal regression instead of least squares line fitting for the purpose of determining the equation of the line on which each line segment lays. Orthogonal regression computes the orthogonal distance between each point and the candidate line, differently from the line fitting algorithm that computes the vertical distance. Orthogonal regression is needed in our case since we are also interested in vertical line segments.

As a final step, after merging lines segments, we prune the segments that are too short to possibly represent a stripe edge. Figure 4.7(c) shows an example of application of our personalized version of EDLines.

4.2.4 GPU Computation of Line Segments Detection

While our implementation of EDLines has been highly optimized, it is still the most expensive operation of Recognizer and it takes about 45% of the entire computation time. The reason is that three operations required by EDLines have a time complexity linear in the number of pixels in the image. These three operations consist in the computation of gradient magnitude, gradient direction and the so called "anchors", i.e. pixels where the gradient operator produces maximal values and that are likely to be edge elements. Since the aim of these three operations is to extract the anchors, we globally refer to them as "anchors extraction".

To reduce the computation time of "anchors extraction", we implemented it through two fragment shaders, so that the computation can be run by the GPU highly parallel architecture. Indeed, while general purpose GPU computation frameworks like CUDA and OPENCL are still not available on mobile devices, it is possible to use programmable fragment and vertex shaders that are actually available in mobile GPUs. The core idea behind a fragment shader is that it defines how to compute each pixel of an output image. To achieve a highly parallel computation, each pixel in the output image must be computed independently from all the others in the sense that it is not possible to use, in the computation of a pixel, the result of the computation of a different one.

The proposed solution adopts a single fragment shader to compute both gradient magnitude and direction. These two operations can be computed in a single fragment shader as both depend on the input image only. Vice versa, anchors computation depends on the result of the other two operations, hence it is implemented in a separate fragment shader. The result of each operation is stored in a different channel of an RGB image.

Our experimental results, run on an iPhone 5s with the methodology presented in Section 4.5, show that, on average, anchors extraction is more than 4 times faster when run on GPU. In absolute terms, the average time required to compute these operations on a single frame is about 8.5ms when computed on the CPU and less than 2ms when computed on the GPU.

4.2.5 Line Segments Grouping

The aim of the line segments grouping phase is to partition the set of line segments into blocks, each one representing a different candidate crossing. Each candidate crossing is characterized by a set of stripes, that, in turn, are composed by a pair of line segments each. During line segments grouping, rectified line segments are processed, so that it is possible, for example, to straightforwardly check geometrical properties (e.g., parallelism) and to compute quantified measurements (e.g., the width of each stripe).

Line segments detected using our custom implementation of EDLines are grouped adopting a custom hierarchical agglomerative clustering technique. Starting with singleton clusters composed of exactly one segment, clusters are grouped according to three criteria: "slope", "horizontal overlapping" and "vertical distance".

The idea behind the slope criterion is that the line segments in the same crossing are mutually parallel. For example, in Figure 4.7(d), line segment 7 is not grouped with the line segments in the dashed box due to the ‘slope’ criterion. The same holds for line segments 9, 10, 11, 12, 13 and 14.

In addition to being parallel, line segments composing a zebra crossing should also be reciprocally ‘aligned’. Technically, consider the projections of the line segments on a line parallel to them; it should hold that the large part of each line segment projection overlaps with the projections of the other line segments. The horizontal criterion captures this property. The evaluation of this criterion should take into account that in some cases a line segment can actually have a small overlap due to the fact that it is partially outside the field of view. It is possible to distinguish these cases because it is known, for each end-point of each line segment, if it lies on the image boundary (see Section 4.2.3). Consider the example of Figure 4.7(d). The line segments in the dashed box are all grouped together, even if the line segments closer to the user have a smaller overlapping: this is due to the fact that part of the line segment is outside the field of view. Vice versa, line segments 3, 4 and 8 are not grouped together with the line segments in the dashed box because their overlap with the other line segments is too small.

Finally, the vertical distance criterion guarantees that, in each group, two consecutive line segments must have opposite gradient directions and a distance of about 50cm (this is specific for Italian regulation). For example, in Figure 4.7(d) line segments 1 and 2 are too close to the line segments in the dashed box and hence are not grouped with these line segments. Analogously, line segments 5, 6 and 15 are too far away and, again, are not grouped together with the line segments in the dashed box.

The first criterion (“slope”) is applied to the entire set of line segments (considered as a single set) and results in a set of blocks, each one used as input for the iterative application of the other two criteria.

4.2.6 Zebra Crossing Validation

After the line segments grouping step, each resulting block is validated according to two criteria: “grayscale consistency” and “number of edges”.

The Grayscale consistency criterion captures the fact that each light (or dark) stripe has a grayscale level that is lighter (darker, respectively) than the average grayscale level of the candidate crossing. Clearly the expected grayscale level (light or dark) of a stripe is known due to the fact that the gradient of its two edges is defined. The minimum required difference between the stripe grayscale level and the crossing average grayscale level is specified by the “grayscale consistency magnitude threshold” parameter. Thanks to this criterion, structures that are geometrically similar to stripes but without consistent dark/light alternating grayscale level are discarded. An example of application of the grayscale consistency criterion is shown in Figures 4.9(a) and 4.9(b). After the grouping phase, some line segments are grouped in a single block and hence are marked as a candidate crossing (Figure 4.9(a)). However, as can be observed in Figure 4.9(b), there is a too small difference in the grayscale intensity of the identified stripes. By enforcing the grayscale consistency criterion the candidate crossing is discarded.

The second validation criterion, is “number of edges”. It defines that a valid zebra crossing should be composed of a minimum number of edges. In most of our experiments, this value is

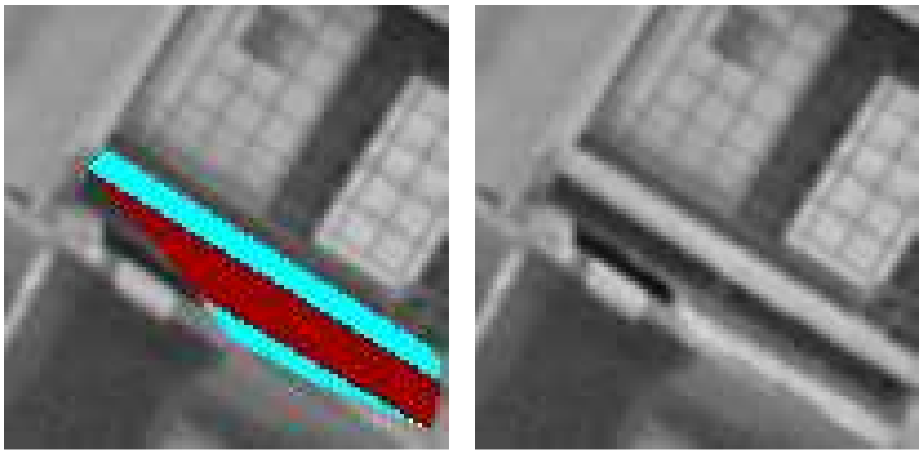

(a) A false positive candidate crossing after grouping. (b) False positive is discarded by "grayscale consistency" criterion.

FIGURE 4.9: Application of the "grayscale consistency" criterion.

set to 5, hence guaranteeing that each crossing contains at least two white stripes, as required by Italian regulation. Consequently, blocks that contain a smaller number of line segments are pruned.

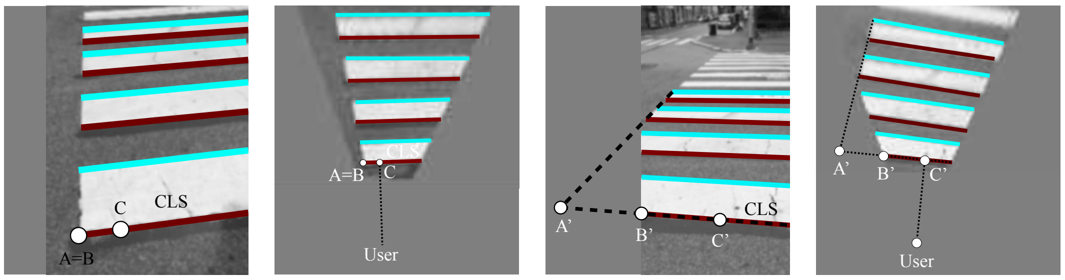
In theory, the number of edges criterion could only be checked as the last step of the recognition procedure (i.e., after enforcement of grayscale consistency). However, checking the number of edges criterion requires a negligible time (i.e., it takes constant time in our implementation). For this reason this criterion is evaluated after each step of grouping and validation in order to reduce the number of line segments to process, hence improving the overall computation time of Recognizer.

A candidate crossing that meets the grayscale consistency and the number of edges criteria is marked as a 'validated crossing'.

4.2.7 Final Result Computation

In many cases either none or a single validated crossing is returned by the validation phase. However, it is possible that two or more crossings are returned. This happens, for example, at crossroads or when there are two consecutive zebra crossings separated by a traffic island. To decide which one is the "most relevant" crossing for the user, we adopted the following methodology. We identified, in a set of sample images (see Section 4.5) the cases in which two or more validated crossings are identified. By observing them, we empirically defined this procedure: the most relevant crossing is the closest to the user among those having roughly the same direction as the user. Consequently Recognizer first checks if any detected crossing has an orientation angle within a threshold from the user's orientation. If favorable crossings are available, all other crossings are discarded. Among the remaining ones, the closest one to the user is selected as the most relevant.

Once the most relevant crossing has been selected, its position with respect to the user is computed for the purpose of guiding the user during the crossing. In particular, the distance is



(a) Left edge is visible. (b) Figure 4.10(a) rectified. (c) Left edge is not visible. (d) Figure 4.10(c) rectified.

FIGURE 4.10: Computation of lateral distance.

computed as a set of five distance measurements, represented in Figure 4.4(c). “Frontal distance” is defined as the distance between the user and the closest line segment (CLS in the following). “Rotation angle” is the (oriented) angular distance between the user’s heading and the crossing. In the figures shown in this chapter we represent the user pointing upwards, so the rotation angle corresponds to the stripes angle. In theory, since the line segments should be mutually parallel, the angle is the same for all line segments. However, in practice, there can be some approximation and hence the rotation angle is computed as the average angle of all line segments. The third and fourth distance measurements are “lateral distance left” and “lateral distance right”. We will describe the former, the latter is analogous. “Lateral distance left” intuitively represents the distance between the user and the left border of the crossing measured on CLS. More formally, it is the (directed) distance between the left border of CLS and the projection of the user’s position on CLS.

There is an issue arising in the computation of “lateral distance left” (the same holds for “lateral distance right”). Indeed, it is possible that the edge of the first detected stripe is not entirely contained in the image. In this case the left end-point of CLS does not necessarily represent the left border of the closest stripe. Let’s consider two examples. In Figure 4.10(a) the left end-point of CLS (point B) actually represents the left end of the stripe (point A). Figure 4.10(b) shows the rectified view. Differently, in Figures 4.10(c) and 4.10(d) the first stripe is not fully contained in the image and the left end-point of CLS (i.e., point B') is not the left end of the stripe (i.e., point A'). In the first case (Figures 4.10(a) and 4.10(b)) it is clear that the user is close to the left border and hence he/she should be instructed to strife right before crossing. Should the same instruction be provided in the second case? The answer is negative. Indeed, by observing the stripes that are farther from the user, it is possible to infer that the first stripe extends on the left of the user hence, intuitively, it is safe to start crossing in the current position. To capture this intuitive reasoning, we take into account the left end-points that are marked as not-being on the image boundary (see Section 4.2.3). If there are too few of these points, the “lateral distance left” is marked as *not quantifiable*. Vice versa, we use an orthogonal regression algorithm to find the stripe “border” i.e., the line that passes through these points. We then compute the intersection A' of this line with the line where CLS lies. The “lateral distance left” is then computed as the length of $\overline{A'C'}$.

4.3 The Logic module

The Logic module is responsible for computing instructions to guide the user in finding a zebra crossing, aligning with it and performing the actual cross. To deliver on this objective, Logic keeps a representation of the position of the user with respect to the crossing and updates it whenever new positioning data is provided by Recognizer. A moving average filter is adopted in order to be tolerant to wrong detections that may be introduced because of sudden movements of the phone or obstacles in the camera field of view. Also, readings from the device's gyroscope are used in order to keep distance information updated between two different runs of Recognizer.

Depending on the current position information, the logic module computes instructions to guide the user in completing the crossing. In order to keep the user focused on the crossing and avoid creating confusion, Logic computes only one instruction at a time, every time new position data is available. It is up to Navigator to chose when and how to convey each instruction to the user. The possible instructions are the following:

- Rise/lower the phone
- Rotate left/right
- Step left/right
- Crosswalk not found
- Crosswalk ahead
- Cross

The next instruction to be conveyed to the user is determined according to the following technique. First, Logic checks if the device's pitch resides in a specific range, defined to let the phone's camera aim at the right height. If the pitch must be adjusted, a "rise/lower the phone" instruction is conveyed. Otherwise, the current instruction is determined according to the presence of a crossing and the user's alignment with it. If no crossing is found, the "crosswalk not found" instruction is conveyed. If a crossing is detected but the user is not aligned in front of it, the "rotate left/right" or "step left/right" instructions are conveyed according to the current position of the user. If a crossing is found but it is too far away from the user, the "crosswalk ahead" instruction is conveyed. Finally, if the user is correctly aligned to the crosswalk, a "cross" message is conveyed. Along with the current instruction, Logic conveys quantified information relative to the instruction. For example, if a "rotate left" message is delivered, the distance from the target angle is conveyed too. Guiding modes based on sonification convey such information to the user.

It is important to note that the "cross" instruction is conveyed by Logic when the user is correctly aligned in front of the crosswalk, but to decide whether it is safe to cross or not is left to the user and her understanding of the environment. In the future, other solutions based on computer vision may be adopted to further support the user by, for example, determining the current state of traffic lights.

4.4 The Navigator module

The Navigator module implements two auditory guiding modes based on data sonification, together with a benchmark guiding mode based on speech messages. The two sonification-based guiding modes are similar, with the main difference being that one produces mono sound (i.e., one single sound signal) and the other produces stereo sound (i.e., two different sound signals, one for the left and one for the right ear). We conducted the sound design process employing a user-centric approach, frequently considering end users feedback and carrying out a preliminary evaluation session. ZebraX was then used to conduct three sets of evaluations aimed at assessing the effectiveness of the guiding modes. The audio files of the sonifications and examples of their application during road crossing are available on-line[2].

4.4.1 Speech guiding mode

Referring to the instructions computed by the Logic module (see Section 4.3), the Navigator module delivers to the user a set of messages generated by the iOS on-board text-to-speech synthesizer. Since the subjects who participated to the evaluation were all Italian mother-tongue, the messages were delivered in Italian (an English translation is available between brackets).

- Abbassa/alza il dispositivo (Rise/lower the phone)
- Ruota a sinistra/destra (Rotate left/right)
- Passo a sinistra/destra (Step left/right)
- Non trovato (Crosswalk not found)
- Strisce davanti (Crosswalk ahead)
- Attraversa (Cross)

Each message is automatically reproduced once, as soon as the Logic module computes an instruction different from the previous one. This choice of timing has been made to prevent the guiding mode to be too verbose and distract the user from paying attention to the environment. However, users can request to reproduce again the last instruction by tapping on any part of the screen. This choice has been made to avoid using buttons that may have been difficult to find during single handed operation of the phone. As a future work, different ways to request the reproduction of the last instruction should be investigated, maybe using voice recognition techniques or wearable devices.

4.4.2 Guiding modes based on sonification

One of the main problems with the speech guiding mode is that it does not convey quantified information about the relative position between the user and the crosswalk. For example, if the user is instructed to rotate right, he/she does not know how much rotation is required in order

[2]http://webmind.di.unimi.it/zebraexamples/

to be aligned with the crosswalk. In theory, it could be possible to design a speech guiding mode in which the quantity is reported (e.g., "rotate right - 20 degrees"). However, this guiding mode would be much more verbose and, most importantly, it would be clearly impractical to update the quantity associated to the message (i.e., the rotation angle in the above example) while the user is moving.

To overcome this problem, the guiding modes based on sonification must inform the user about the *quantity* associated with the instruction. For this reason we base our technique on *parameter mapping sonification* [34], which is based on the creation of a link between the data to be rendered and the parameters of a synthesizer (or of any other device which generates or plays back sound).

The process of user-centric analysis of the system raised another important requirement that has a direct impact on the sound design. Most people with VIB are not willing to wear headphones, as this prevents the acquisition of audio information from the environment (e.g., an approaching car). This problem can be partially solved by using bone-conducting headphones[3]. However, some users declared to find bone-conducting headphones rather uncomfortable, due to the mentally-demanding task to distinguish the sounds produced by the headphones from the environment sound.

Two solutions have therefore been designed: the mono sonification delivers one monaural audio signal, which is suitable to be played by the device speaker. Vice-versa, the stereo sonification employs sound spatialization in order to allow the user to clearly perceive certain sounds as coming from the left or from the right, therefore conveying information using an additional cue. This sonification requires the user to wear a pair of headphones, and employs, for a determined set of messages, a binaural spatialization approach [32]. Considering the low resolution of bone-conducting headphones in terms of high frequencies (above 10 kHz), and the complexity of the individual-related features of a full Head Related Transfer Function (HRTF) simulation, the stereo technique was not implemented performing a full spatialization. A simpler approach was taken, modifying the differences in level and time of arrival of the sound at the two ears (i.e., Interaural Level Differences - ILD and Interaural Time Differences - ITD).

Two further requirements emerged during sound design:

- Since for certain types of messages the understanding of the pitch of the sound is essential, the fundamental frequency of the stimulus had to be easily perceived.
- For a precise spatialization, the sound had to feature a large and dense spectrum.

For these reasons, a custom set of impulsive sounds of short duration was designed and implemented. The test sound was produced by additive synthesis of 5 to 20 harmonic or inharmonic partials (depending on the type of message to be sonified), each implemented by an exponentially damped oscillator. Attack times of all partials was set to 1 ms. The relative amplitude of the partials followed a roll-off of -3 to -6 dB/octave, whereas decay times differed depending on both the partial and the sonified message type (a similar approach was employed by Katz et al. [45]). Different repetition and envelope patterns were also used in order to allow a clear distinction between the sonification of different instructions.

[3]Bone-conducting headphones do not occlude the ear canal and, therefore, do not impede the perception of sounds from the surrounding environment.

Mono sonification

In order to deliver left-right-type messages without relying on sound spatialization, low pitch sounds were associated to a rotation/step towards the left, and high pitch sounds towards the right. This choice can be intuitively explained considering the keyboard of the piano from the point of view of the player (high-pitch notes on the right).

Considering the list of speech messages in Section 4.4.1, the following mono sonifications have been designed and implemented:

- *Rise/lower the phone.* Impulsive sound with fast transients and harmonic spectrum (similar to a short beep). Two quick repetitions with no pause. High pitch (800 Hz) for the 'rise' message and low pitch (200 Hz) for the 'lower' message. The signal is repeated increasing linearly the rate (from 1 Hz to 2.5 Hz) the closer the user gets to the right inclination.
- *Rotate left/right.* Impulsive sound with fast transients and in-harmonic spectrum (similar to a percussive sound on metal). The left-right information is delivered modifying the frequency of the stimulus; 300 Hz for the left rotation and 1200 Hz for the right rotation. The repetition rate of the sound is modified linearly from 1.6 Hz (large rotation) to 3.3 Hz (small rotation), varying continuously until the user reaches the target angle.
- *Step left/right.* Impulsive sound with fast transients and in-harmonic spectrum (similar to a percussive sound on wood). Two fast (200 ms) repetitions. The left-right information is delivered modifying the frequency of the stimulus; 300 Hz for the left step, and 1200 Hz for the right step.
- *Not found.* Low frequency (200 Hz) in-harmonic sound, slow transients, two repetitions (300 ms the first and 500 ms the second).
- *Crossing ahead.* Pure-tone (single frequency with no harmonic components) impulsive sound. A rising scale of 6 notes (between 800 and 1700 Hz, one each 100 ms) for a required 10 m advance, 5 notes for 8 m, 4 notes for 6 m, 3 notes for 4 m and 2 notes for 2 m. The scale is repeated every 1000 ms, modifying the message as the person gets closer to the target.
- *Cross.* Impulsive sound with fast transients and in-harmonic spectrum (similar to a percussive sound on wood). A group of three notes (one note every 150 ms) with fundamentals at 500-800-1000 Hz is repeated every 1200 ms. If the user is required to proceed towards the right, the frequency of the fundamentals is multiplied by 0.33 (lower pitch), while if towards the right is multiplied by 2 (higher pitch). The level of the sound is rather low, but it becomes louder (up to +20 dB) the more the user needs to modify the path towards the left or the right. When the user is at less than 4 meters from the target, the delay between repetitions is decreased linearly (down to 700 ms).

Stereo sonification

In the stereo sonfication mode the audio signal is delivered differently to the two ears. The user is therefore able to clearly localise a sound in any position between left, center and right. As outlined earlier, the spatialization was performed employing ILD (from 0 to 10 dB) and ITD (from 0 to 0.5 ms)

The following stereo sonifications have been designed and implemented:

- *Rise/lower the phone.* Same as mono mode.
- *Rotate left/right.* Same sound as mono mode, frequency 500 Hz. The impulse is continuously repeated every 400 ms, and is spatialized on the left if the user needs to turn left, and vice-versa if the user needs to turn right. The repetition continues until the user can center the sound on the front (therefore when reaching the target angle).
- *Step left/right.* Same sound as mono mode, frequency 500 Hz. Sound spatialized on the left or on the right (depending on the required direction)
- *Not found.* Same as mono mode.
- *Crossing ahead.* Same as mono mode.
- *Cross.* Same sound as mono mode, with frequencies 500-800-1000 Hz. The left-right direction is given by gradually spatializing the sound on the left or on the right, so that the task of the user is to rotate in order to keep the sound central.

4.4.3 Preliminary evaluation

During the design of the auditory guiding modes several test subjects were asked to use the application and provide feedback. In addition to these informal evaluations, a preliminary evaluation was carried out in order to allow for the fine tuning of the whole application, and in particular of the auditory guiding modes. This section describes the evaluation methodology, its results and how the guiding modes were changed according to this evaluation.

Evaluation methodology

The evaluation was conducted at the Milan Institute for Blind People (Istituto dei Ciechi di Milano[4]), which offered support in terms of location and test subjects.

The evaluation was conducted with five congenitally blind test subjects in a controlled environment, namely a large corridor (20m long, 6m wide approximately), where a real-size zebra crossing was represented on a large plastic sheet. The choice of conducting the evaluation in an indoor space was driven by the fact that we wanted test subjects to focus on the sonified audio without being distracted from environmental noise,even if such space provided acoustic cues that could ease orientation. The auditory guidance information was delivered using a pair of wired bone conducting headphones[5], connected with an iPhone 5. Each test subject was asked to perform five tasks in random order, one task for each one of the instructions listed in Section 4.4.1 (except for *Not found*). The goal of each task was to reach a target position (e.g., by rotating, by moving forward, etc.) starting from a random position. Each task was repeated

[4]`http://www.istciechimilano.it/`

[5]Headphones model is *Goldendance Audio Bone Aqua*

three times, once for each auditory guiding mode (again, in a random order), and was preceded by a five minutes training.

The following data was measured for each task and auditory modality: time to perform the task, average error (distance from the target, in degrees or metres), and tolerance (number of times each person entered and exited a small area around the target).

At the end of the evaluation, every test subject was asked to give feedback about the application, in particular about the three auditory guiding modes.

Evaluation Results

Considering the low number of test subjects, statistical significance was not calculated. Based on simple descriptive statistics we observed that, in tasks concerning rotation (i.e., rotate left/right and raise/lower the phone), the two sonification guiding modes were more effective than the speech guiding mode. Regarding the other instructions, no notable difference was observed among the three audio guiding modes.

Regarding the test subjects' feedback on the application, it is worth noting that all of them reported to be unable to judge the effectiveness of speech and sonification guiding modes in the real world (i.e., with traffic noise). To address this problem, successive evaluations (see Section 4.6) were conducted in outdoor spaces, with audible traffic noise. Furthermore, the following comments were made by more than two subjects:

1. The sound spatialization was not evident. It was often not possible to clearly distinguish when a sound was coming from the left, center or right.
2. The repetition rate changes, which for certain sonified messages indicated the proximity to the target, were not clearly identifiable.
3. Both sonifications required longer training if compared with the speech messages.

In addition to these comments, we observed that in some cases the headphones wire entered the camera field of view, hence preventing the computer vision technique to work properly.

4.4.4 Updated auditory guiding modes

Certain features and parameters of the auditory guiding modes were modified in order to reflect the results of the preliminary evaluation.

To address the first comment, a simple evaluation was carried out in order to establish the minimum detection thresholds for ILD and ITD using bone conducting headphones. Using a simple up-down 1 dB step adaptive procedure [49], the discrimination threshold was measured for seven test subjects. The mean discrimination threshold (i.e., the smallest inter aural difference which allowed a test subject to position a sound source on the left or on the right) for the ILD was 1.15 dB, and for the ITD 0.13 ms. Considering that these mean discrimination thresholds are sensibly larger to the ones obtainable with a standard pair of headphones, the spatialization ranges were changed. The ILD was increased to a maximum of 20 dB (before it was 10 dB), and the ITD to a maximum of 1 ms (before it was 0.5 ms).

To address the second comment, the following minor modifications have been applied:

- *Rise or lower the mobile phone* - the repetition rate has been increased to a maximum of 3.3 Hz (before it was 2.5 Hz).
- *Step left or right* - the repetition rate has been linked to the required displacement (before, the sonification was of boolean type, therefore no information was delivered about the amount of required displacement). The stimulus is repeated every 800 ms if the required displacement is relatively large (2 m), increasing linearly the repetition rate (up to one repetition each 400 ms) for smaller displacements (50 cm).

Finally, considering the third comment, an additional functionality was added to ZebraX. In all auditory guiding modes, the user can tap on the screen of the device to listen the current instruction through a speech message. In practice, with the speech guiding mode, upon tapping on the screen ZebraX repeats the last message that was played. Vice versa, with mono and stereo, upon tapping ZebraX provides a speech explanation (using the same messages defined for the speech guiding mode) of the instruction being sonified. The addition of an optional touch-activated speech message within the sonification guiding modes represents a major change in the design of the guiding modes, which is discussed in Section 4.7.

4.5 Evaluation of the Recognizer module

4.5.1 Experimental methodology

When Recognizer is run in ZebraX, the input data are taken directly from the device's camera and sensors, and this makes it impossible to run the recognition procedure twice with the same input. Clearly this is a problem when debugging the application, tuning parameters and measuring performance. To overcome this issue we developed two applications to first collect images and then process them off-line. *zRecorder* is a mobile application that records the stream of images and motion sensors data (i.e., accelerometer and gyroscope). The other application, *zSimulator*, reads the data stored by zRecorder and uses it as an input to run Recognizer so that its performance can be measured. zSimulator can be run both on traditional devices (i.e., desktops and laptops) and on mobile ones. This approach significantly eases the debugging process and enables regression tests, parameters tuning and reproducible experimental tests.

We used zRecorder to create four sets of images (with corresponding motion data) at 1080×1920 resolution. All sets are publicly available[6]. The first set, called *Testset1*, consists of 40 videos and 4015 frames captured in different illumination conditions (sunny, cloudy and night). All frames have been manually annotated to distinguish those containing a zebra crossing (1877) from the remaining ones (2138). The second set, called *Testset2*, includes 6 videos with 206 frames. In this case, for each frame we also annotated the relative position of the crossing. To

[6]http://webmind.di.unimi.it/ZebraRecognizerTestSet/

minimize the approximation while collecting this information, we recorded the videos by using a tripod positioned at a given frontal and left/right distance from the crossing. Since the tripod is stationary, the frontal and lateral distances are fixed for each video, while the rotation angle varies. To measure the rotation, before starting the recording, we calibrate the device so that it is perfectly perpendicular with the stripes and then, for each frame, we measured the rotation angle by using gyroscope readings. We empirically observed that the error introduced by the gyroscope is negligible, also considering that the duration of the recording is of few seconds and that the device is not subject to sudden movements (since it is on a tripod). The third set, called *Testset3*, is a subset of Testset1 that contains only heavily blurred images with zebra crossings (manually selected from Testset1). Testset3 contains 613 images, mostly taken in conditions of low ambient light. Finally, the fourth set, called *Testset4* contains 265 images of zebra crossings that are partially covered by external objects, for example a pole (like in Figure 4.8).

We used a desktop pc for computationally intensive evaluations (e.g., parameters tuning) and an iPhone 5s smartphone for evaluating the execution time of the final application.

We take four indicators into consideration: precision, recall, execution time and positioning accuracy. Precision, calculated as the ratio between the correctly detected crossings and all the detected crossings, measures the amount of false positives. A precision score of 1.0 means that each detection corresponds to a crossing in the examined image, conversely a lower ratio implies that some crossings were detected where none was present. The recall metric is computed as the ratio between the detected crossings and all the correct crossings in the dataset. While a score of 1.0 means that all the crossings were correctly detected, lower values indicate that some of the crossings were not. Given the safety concerns for the navigation of users with visual impairment or blindness in a dangerous environment, we notice how anything less than a perfect precision score is unacceptable, while a high recall score, although important, is less critical. Henceforth, unless differently stated, we report our results in which the precision is always equal to one. We point out, however, that precision may be less than one in other test sets and in real world usage scenarios.

The execution time defines the average time needed to run Recognizer. It does not take into account the time required to load the image from the hard drive nor the time required to resize the input image. Indeed, when Recognizer is used in ZebraX, the input image is already acquired at the necessary resolution and no resizing is needed. Clearly, lower execution time allows higher frame rates, increasing the detection responsiveness with respect to the user's movements. Also, it means that the procedure is less computationally intensive, with a lower power consumption.

Finally, the positioning accuracy indicates to which extent the relative position returned by Recognizer is precise. The positioning accuracy in a given frame is characterized by four values, one for each distance measurements. Each value is the difference between the distance computed by Recognizer and the expected (actual) value. Clearly, positioning accuracy can only be computed if the expected relative distance is known and hence only using *Testset2*.

It is worth noting that, in the following, different versions of Recognizer are compared. As Recognizer is a completely analytical computer vision technique, it does not adapts to any training or test set. Instead, each version of Recognizer is different because of the line segment detection approach adopted, of algorithmic improvements aimed at increasing recall and other

Parameter	Min	Chosen	Max
Resolution	90×160	180×320	720×1280
Grouping angle	1.5	3	7.5
Grayscale consistency	1	5	9
Blur kernel size	3	9	13
Blur standard deviation	0.7	0.9	1.4

TABLE 4.1: Most influential parameters and their values.

optimizations introduced to reduce the computation time. The parameters of each version are always tuned using *Testset1*.

4.5.2 Parameters tuning

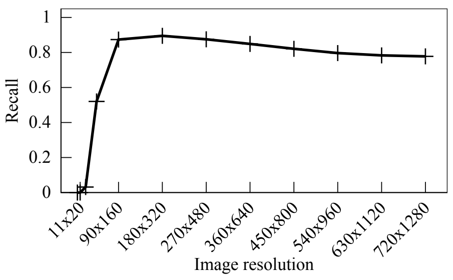
This section reports the results of the study conducted to tune five representative parameters that highly influence recognition performances: “resolution”, “grouping angle”, “grayscale consistency”, “blur kernel size” and “blur standard deviation” .

The “resolution” parameter specifies the size of the image on which the detection is run. The “grouping angle” parameter defines the maximum angular distance between two line segments that are grouped together (see Section 4.2.5). The “grayscale consistency” parameter defines the minimum difference in grayscale level (value range between 0 and 255) between a stripe and the average grayscale level of the crossing (see Section 4.2.6). The last two parameters refer to the strength of the blur filter applied during image pre-processing (see Section 4.2.2). These parameters are listed in Table 4.1 together with their minimum and maximum values used during parameters’ tuning, and their default chosen values.

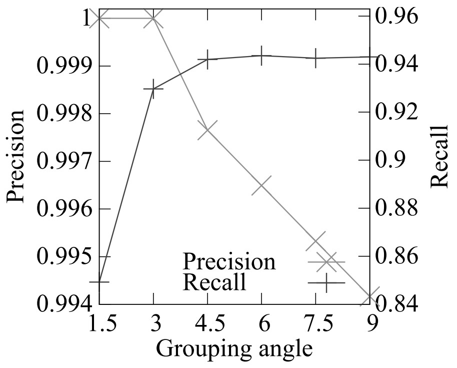
Figure 4.11(a) shows that with a very low resolution (below 90×160) recall diminishes drastically. This is due to the fact that in these cases the features are hard to detect. For high resolutions (above 180×320) there is also a reduction in recall due to the fact that noise and imperfections are more visible and impair drastically the segment detection stage. While this behavior can be offset by using stronger blurring during the preprocessing step (see Section 4.2.2), higher resolutions do not improve the detection accuracy. Thus, the default resolution used for the detection is 180×320 pixels.

For the “grouping angle” parameter we observe (see Figure 4.11(b)) that, for larger values of this parameter, recall is higher due to the fact that larger blocks of line segments are generated with the application of the “slope” criterion hence it is less likely that they are pruned by the “number of edges” criterion. However, for values larger than 3°, some false positives can be introduced and hence precision diminishes, although very slowly. For this reason, the default value is 3. The analysis for the “grayscale consistency” parameter is similar (see Figure 4.12(a)): for smaller values of this parameter the “grayscale consistency” criterion is easier to satisfy, hence there is a higher recall. However, for values smaller than 5 precision is less than 1. Hence, we choose 5 as the default value.

Figure 4.12(b) shows that, for what concerns the “blur standard deviation” parameter, there is a peak in both precision and recall for the value 0.9. Thus, we chose this value as the parameter default. For the “blur kernel size” parameter, we can observe in Figure 4.12(c) that, for values

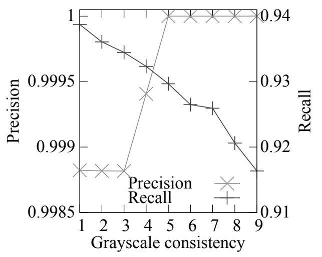


(a) Image Resolution parameter. Recall significantly drops at resolutions below 90x160.

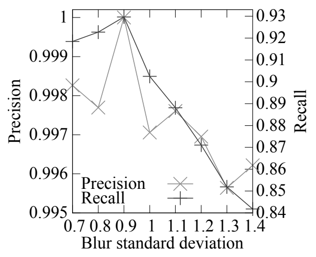


(b) Grouping angle parameter. For grouping angles wider than 3 precision is no longer 1, while recall slightly improves.

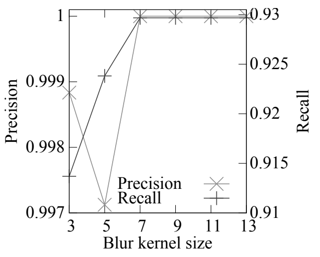
FIGURE 4.11: Results of tuning the image resolution and grouping angle parameters.



(a) Grayscale consistency. With values smaller than 5 precision is no longer 1.



(b) Blur standard deviation. There is a peak in both precision and recall for the value 0.9.



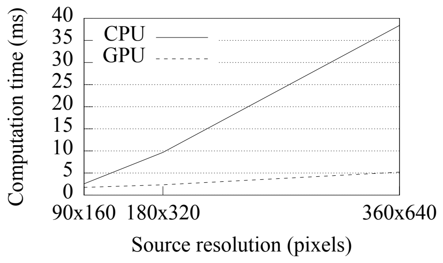
(c) Blur kernel size. Kernel sizes above 7 have no impact on precision and recall.

FIGURE 4.12: Results of tuning the grayscale consistency, blur standard deviation and blur kernel size parameters.

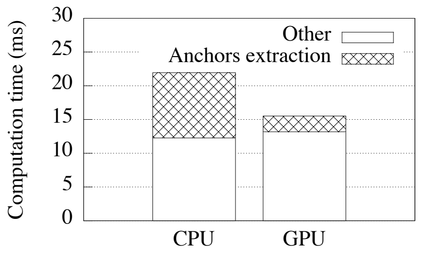
smaller than 7, there are some false positives (i.e., precision is less than 1). Vice versa, when this parameter is set to 7 or higher, precision is 1. For values larger than 7, both precision and recall are not influenced, but the computation costs are higher. Hence, we chose the value of 7 for this parameter as default.

4.5.3 Impact of GPU computation

One set of experiments is aimed at assessing the improvements of the GPU implementation of anchors extraction (see Section 4.2.4). Figure 4.13(a) shows the comparison between the CPU and the GPU implementations for different values of the "resolution" parameter. As expected, this parameter significantly influences the execution time of anchors computation as this is an operation with time complexity linear in the number of pixels. Indeed, the computation time of the CPU implementation is 2.5ms for images with resolution 90×160, while it is almost exactly four times larger (i.e., 9.67ms) for images with four times the number of pixels (i.e., 180×320). The same increase can be observed for images with resolution 360×640. Differently, with the GPU implementation, the total computation time is composed by a constant-time overhead (we estimate its cost to be about 1.5ms) and the actual computation, whose cost is indeed linear in the number of pixels and about 10 times faster than with the CPU implementation. So, overall, while the computation on the GPU leads to an improvement of about 30% for 90×160 images,



(a) Computation time of anchors extraction as a function of the source resolution. Even if computation times increase almost linearly for both implementations, the rise of the CPU version is more steep.



(b) Impact of anchors computation time in Recognizer. At the default resolution, anchors extraction implemented on the GPU is about 4 times faster than on the CPU.

Figure 4.13: Comparing the computation time of the CPU and GPU implementations of Recognizer.

the improvement is much larger with 180 × 320 images (the default resolution value) where the GPU implementation is more than 4 times faster. In our experiments we also observed that for larger images the benefits are even higher (e.g., for 360 × 640 images the GPU implementation is about 8 times faster).

One question is how the GPU implementation of anchors extraction impacts on the overall computation time of Recognizer. Figure 4.13(b) helps us provide an answer by showing, at the default resolution, how the entire computation time of Recognizer is divided between anchors extraction and all other operations. When anchors extraction is computed in CPU, it requires almost the same time as all the other operations (precisely, anchors extraction takes 44% of the entire computation time). Vice versa, with the GPU implementation, anchors extraction requires 15% of the entire computation time. Since the GPU implementation is about 4 times faster, it improves the overall Recognizer computation time by about 30%.

4.5.4 Robustness

We used Testset3 and Testset4 to evaluate the robustness of the proposed solution when the zebra crossing is heavily blurred or partially covered. In this analysis, since the two testsets contain true positives only (all images contain a zebra crossing), we only evaluated recall. Note that Recognizer has not been specifically tuned for these two testsets of images: the values of all parameters are the same as defined in the tuning phase, conducted with Testset1 (see Section 4.5.2).

Figure 4.14 shows a comparison of the recall obtained in Testset1, Testset3 and Testset4 when two different techniques are used for line segments detection. In this section we consider the default technique only (EDLines); we discuss the results with the other technique (LSD) in Section 4.5.5.

We can observe in Figure 4.14 that, when Recognizer is run with heavily blurred images, recall slightly improves (from 0.93 with Testset1 to 0.96 with Testset3). This is due to the fact that images in Testset3 are mainly captured in conditions of low ambient light (when it is easier to have heavily blurred images). In this light condition, there is a higher contrast between light stripes and the dark background, which makes it easier to detect crossings.

For what concerns Testset4, we can observe that, considering only images in which the stripes are partially covered, the decrease in recall is very small: from 0.93 with Testset1 to 0.88 with Testset4. This supports the fact that, in the great majority of the cases, the line segment detection algorithm is able to reconstruct the entire stripe edge, even when it is partially occluded.

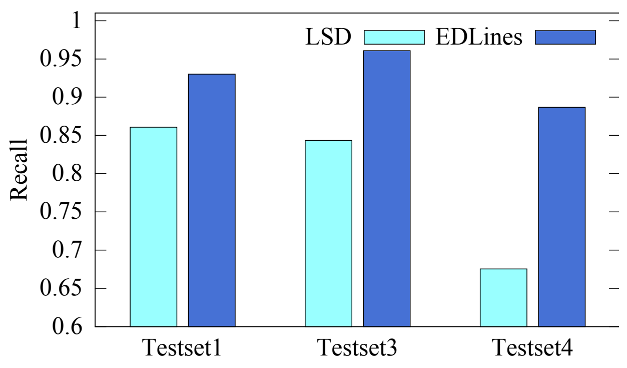


FIGURE 4.14: Recall with Testset1, Testset3 and Testset4 using EDLines and LSD.

4.5.5 Comparison with previous solutions

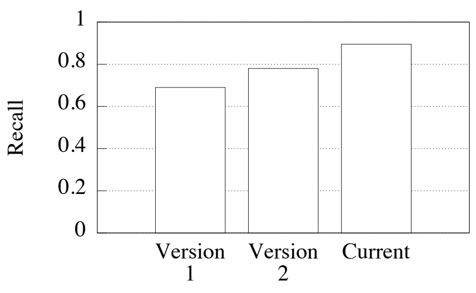
In this section we first compare the impact on Recognizer of two different algorithms for line segment detection and then we compare the solution presented in this chapter with our previous ones.

Since line segment detection is a crucial part of our technique, we investigated the impact of two different approaches: a customized version of EDLines (described in Section 4.2.3) and a customized version of Line Segment Detector (LSD) [86].

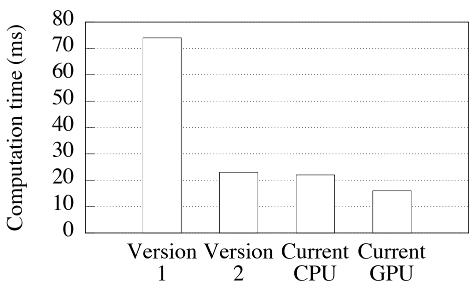
We implemented a version of Recognizer adopting LSD and we tuned it with the same methodology described in Section 4.5.2 for the “standard” Recognizer version that uses EDLines. In practice, we tuned the parameters to obtain no false positives (i.e., to have precision 1) and to have the highest possible recall. Figure 4.14 shows that the algorithm performs consistently better, in terms of recall, when EDLines is adopted. Indeed, in Testset1, the recall is 0.93 and 0.86 for EDLines and LSD, respectively. A similar result is obtained for Testset3. For Testset4, the Recognizer version using EDlines yields a much higher recall score. This suggests that the solution based on LSD is less efficient in reconstructing the line segments if they are partially occluded. Also, when LSD is adopted, Recognizer has a computation time that is about 3 times higher than with EDLines. For the above reasons, we can conclude that EDLines outperforms LSD for this specific application.

We now compare the solution presented in this chapter with previous solutions [4, 3], that are henceforth called “Version 1” and “Version 2”, respectively. A direct comparison with other solutions is unfeasible because the implementation and the data used for their evaluation are not public. Vice versa, as we explain in Section 4.5.1, the data used for our tests is public, so that a direct comparison of future works with our solution is possible. The three solutions are compared according to two metrics: recall and computation time.

For each version we use the corresponding default system parameters, which, as previously stated, are tuned to yield a precision equal to 1.



(a) Recall comparison.



(b) Computation time comparison.

FIGURE 4.15: Performance comparison between different versions of Recognizer (precision is 1 in all versions).

For what concerns recall, Figure 4.15(a) shows that it improved from .69 in Version 1 to .78 in Version 2 up to .93 in the current version of Recognizer. The improvement from Version 1 to Version 2 is mainly due to the fact that in Version 2 the geometrical properties are checked on the rectified image. The improvements from Version 2 to the current version is due to the number of improvements described in previous Sections.

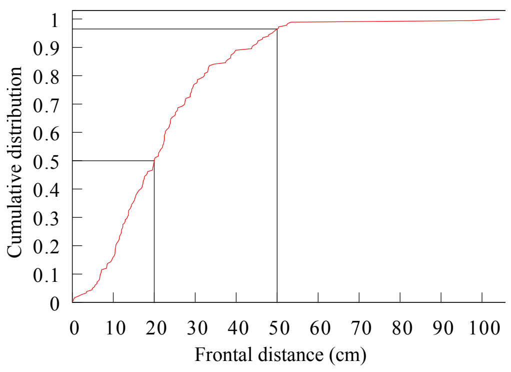
For what concerns the computation time, in Version 1 the average time to process each frame is 74ms, while in Version 2 it is 23ms. In the current version of Recognizer the average time is 22ms with the CPU implementation of anchors extraction while it is 16ms with the GPU implementation. Considering also the image acquisition time, ZebraX can process about 25 frames per second.

The small improvement between Version 2 and the current CPU implementation is due to two contrasting factors: on one side, we engineered and optimized the code, hence improving the computation time by about 20%. On the other side, we fixed a bug in the line segments merging algorithm (see Section 4.2.3). The effect of the bug was to erroneously terminate before merging was complete, hence resulting in a partially incorrect result but faster computation. After fixing this bug, all line segments are now correctly merged, but the improvement in computation time from Version 2 to the current version is negligible. Still, the GPU implementation of anchors extraction guarantees an improvement of about 30%.

4.5.6 Positioning accuracy

A set of experiments is aimed at asserting the approximation introduced when computing the four distance measurements (see Section 4.2.7). In the following we indicate as “error” the absolute value of the difference between the distance (frontal, angular or left/right shift) returned by Recognizer and the expected (correct) distance. It is worth noting that, in some cases the curvature of the road surface may introduce a small error in distance measurements. The analysis of such issue is left as a future work. However, our experiments have been conducted on a flat road surface and results presented in the following are not affected by this issue.

For what concerns the frontal distance, the average error is 0.22m. Figure 4.16(a) shows the cumulative distribution function (CDF) chart of the error occurring in the computation of frontal


(a) Frontal distances cumulative distribution.

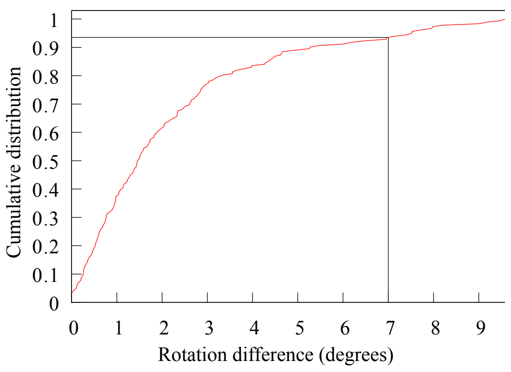

(b) Rotation distances cumulative distribution.

FIGURE 4.16: Accuracy of frontal and rotation distances.

distance. It can be observed that in 50% of the cases the error is less than 20cm, while in 96% of the cases the error is less than 50cm, which corresponds to approximately one step.

In a few cases the error is about 1m: this is due to the fact that the first white stripe is not recognized. Fortunately this problem occurs in few frames (less than 3%) that are generally non-consecutive (the longest sequence we measured is composed by two consecutive frames). This makes it possible to identify the occurrence of this problem in the Logic module by checking for sudden changes in the frontal distance. Indeed, since the temporal distance between two consecutive frames is less than 0.05s (frequency is about 25 frames per second), a change in the frontal distance larger than 0.5m clearly indicates that the closer stripe has not been recognized. The results of the frontal distance error also show that the mean error is larger when the observer is far from the crossing. For example, when the observer is 4m far from the crossing, the mean error is 0.24m, while for a distance of 2m the mean error is less than 0.2m.

For what concerns the rotation angle, the average error is about 2.2°. Figure 5.3 shows the CDF chart: it can be observed that the error is up to 9.5° and that in 93% of the cases the error is less than 7°. In this case it is not possible to check for sudden changes, as a user can possibly rotate very quickly. Nevertheless values can be smoothed by using a moving average in the Logic module. For example, with a moving average of length 3, the average error in the rotation angle is 1.0° and the maximum error observed in the experiments is 3.0°.

During our experiments we observed that the computation of the lateral distance is subject to a non-negligible approximation caused by two factors: first, EDLines frequently does not recognize the entire stripe edge, but just a portion of it and consequently the computation of the border is not always precise. See Figure 4.17 for an example. Second, the projection of the user's position on CLS (the line segment closest to the user) can be imprecise due to approximations in the computation of CLS angle. To address the former issue, our solution excludes, from the border computation, the points that introduce an error above a given threshold as, for example, point A in Figure 4.17. This is useful, for example, when there are few line segments that are much shorter than the actual stripe edge. To address the latter issue, when computing the projection on CLS, instead of using the angle of CLS, we use the average angle computed among all the stripes. The resulting technique always correctly identifies a lateral distance as *not quantifiable* (i.e., the border is out of the field of view, see Section 4.2.7). In some rare cases it happens that, even if the border is visible, it is still identified as *not quantifiable*. This is often due to the fact

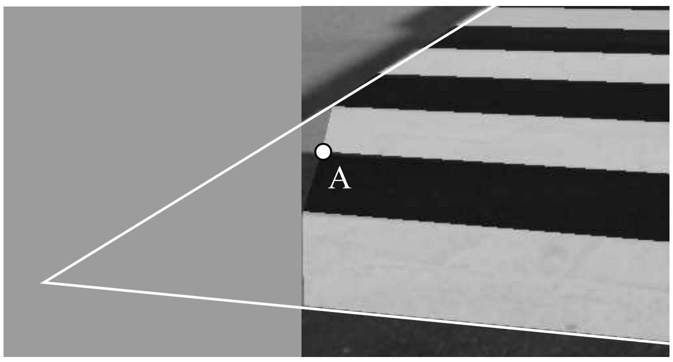


FIGURE 4.17: The stripe's edge is not completely recognized.

that the border is only visible in stripes that are far away from the user and these stripes are not recognized. In any case, in 87% of the cases, if a border is visible then it is identified by our technique and, in these cases the average error is 0.25m.

4.6 Evaluation of the auditory guiding modes

Considering the difficulties in recruiting test subjects with VIB, we decided to carry out the evaluations also with sighted individuals. We conducted three sets of empirical evaluations: a quantitative evaluation with 11 blindfolded sighted test subjects (Section 4.6.1), a qualitative evaluation with 12 blind test subjects (Section 4.6.2) and, finally, a quantitative and qualitative evaluation conducted with 3 test subjects with VIB (Section 4.6.3). In Section 4.7 we report a discussion of the empirical results.

The evaluations were conducted with an iPhone 5s, and all test subjects wore wireless bone-conducting headphones[7]. During the evaluations, subjects with VIB who were used to walk with a white cane kept it in one hand, while the smartphone was held in the other hand, pointing in front of them.

4.6.1 Quantitative Evaluation with Sighted Test Subjects

The quantitative evaluation was conducted with 11 blindfolded sighted test subjects. In the following sections the evaluation settings and methodology are described first, followed by the presentation of the results.

Evaluation Setting and Methodology

The evaluation was conducted in an outdoor environment where a real-size zebra crossing was represented on a large plastic sheet. The zebra crossing used during the evaluation is compliant with Italian traffic regulations; it is composed by five light stripes over a dark background, and each stripe is 2.5m large and 0.5m wide[8] (see Figure 4.18).

[7]Headphones model is *Aftershoks bluez 2*

[8]Italian regulation defines zebra crossings that are similar to those used in most countries worldwide

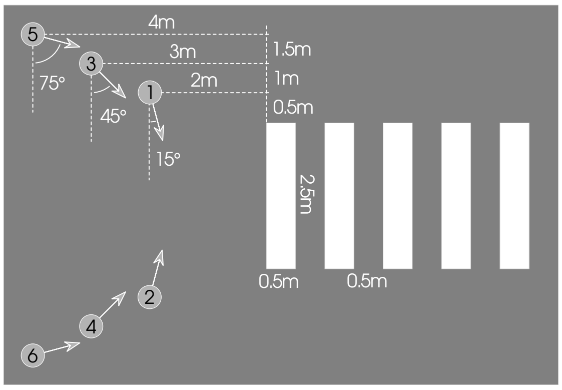


FIGURE 4.18: Layout of the plastic sheet on which the evaluations were conducted. Numbers and arrows represent starting points and starting directions, respectively.

The outdoor environment was chosen in order to give a more realistic setting to the tests. In order to reduce the test subjects' ability to orientate using environmental sounds, and to minimize hazards, it was decided to carry out the evaluation in a large courtyard. Sound of traffic and other environmental noises were audible, but particularly diffuse in the environment and generally not usable for orientation purposes. For the same reason, the plastic sheet was moved or rotated after each test, so that it was impossible for the test subjects to predict the position of the zebra crossing based on previous tests. Furthermore, in order to avoid that tactile and/or audio feedback coming from the ground surface could give clues to help orientation, the whole testing area was covered by a very large plastic sheet

Each evaluation was organized into three phases: learning, practice and measurements. During the learning phase each test subject had access to a document describing the evaluation structure, introducing ZebraX and the three different auditory guiding modes. The document was presented in the form of an HTML page, so that test subjects could listen to sonification examples[9].

During the practice phase, each test subject could try ZebraX with the three auditory guiding modes. No time constraints were enforced; each test subject could freely decide how long to practice with each guiding mode, until he/she felt comfortable with it. On average, test subjects tested the speech guiding mode for about 1 minute, and the other two guiding modes for about 2 minutes each.

During the measurement phase each test subject was asked to autonomously align with the zebra crossing and to actually cross it. These two operations were repeated for two "rounds" of tests. During each round, three tests were conducted, one for each guiding mode, in order: speech, mono, and stereo. For each test, the subject started from a different point, in a different starting direction. The choice of the starting points was determined by the idea that the time and effort required to find the crossing, align and cross should be almost the same for all starting points. After some informal evaluations, the 6 starting points depicted in Figure 4.18 were chosen.

During the measurement phase, the ZebraX app recorded a number of parameters related with the completion of the task. These included: the time to align (i.e., to reach the first stripe), the time to cross (i.e., from the first stripe to the end of the crosswalk), the complete list of messages and the number of taps on the screen to repeat/clarify the message.

[9]The document was presented in Italian. Its English translation is available here: http://webmind.di.unimi.it/zebraexplanation/

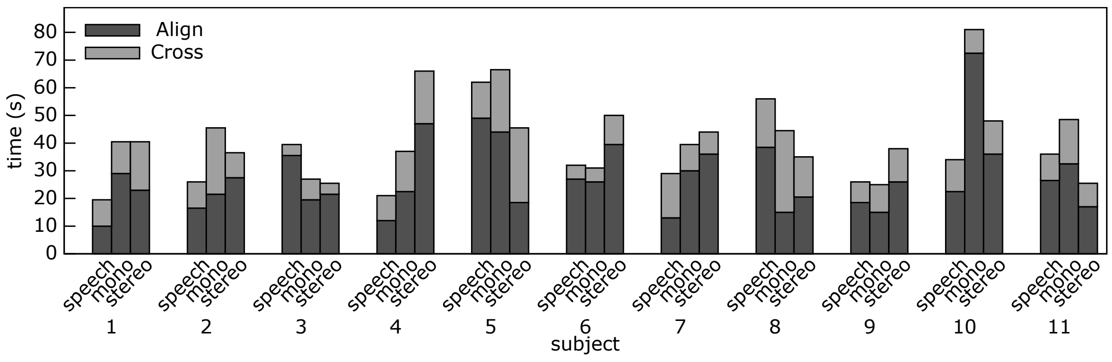


FIGURE 4.19: Average alignment and crossing time in the two rounds of the quantitative evaluation conducted with sighted subjects.

Evaluation Results

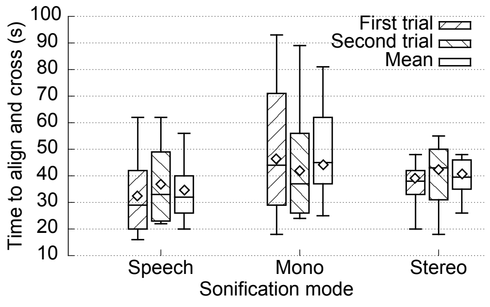
During the measurement phase all test subjects were able to successfully complete all crossings. The only exception was the test subject 6 who, during the test with the mono guiding mode - second round, misinterpreted a "rotate left" message and walked straight. Since the subject was going to hit a parked car, the supervisor had to stop the test.

Figure 4.19 shows, for each test subject and each guiding mode, the average time required in the two rounds to align and cross. We can observe that 5 test subjects have been able to align and cross faster with speech guiding mode, 2 test subjects with mono and 4 with stereo. Mean alignment time is 24s, 29s and 28s with speech, mono and stereo modes respectively, while mean crossing time is 10s, 14s and 12s respectively. Overall, the mean time to align and cross is 34s, 44s and 41s.

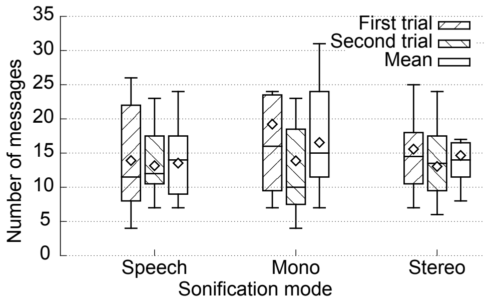
The above results seem to suggest that there is not a clear difference in crossing time for the three guiding modes. These results can be also graphically observed in the boxplot shown in Figure 4.20(a). This chart also seems to highlight that, differently from what expected, there is no learning effect between the first and second round. Indeed, on average, the crossing time in the second round is slightly lower for the mono guiding mode compared with the other two guiding modes.

Another metric that can help understand the performance of the three guiding modes is the total number of changes in the message to be conveyed during the task (this metric will be referred to as "number of messages"). Clearly, a smaller value indicates higher performance. In this case it emerges that speech and stereo guiding modes yield very similar results, while mono sonification requires a slightly larger number of instructions, on average (see box plot in Figure 4.20(b)).

Inferential statistics have been performed to identify whether the differences between guiding mode groups are statistically significant. Considering the time to align and cross, the data sets are normally distributed, therefore a one-way ANOVA was conducted. The results show that there are no statistically significant differences between the three groups ($F(2, 63) = 1.178$, $p = 0.314$). Similarly, no statistical difference was found between the first and second round performances, and between the starting points (for all guiding modes).



(a) Time to align and cross.



(b) Number of messages.

FIGURE 4.20: Boxplot representation of results from the quantitative evaluation conducted with sighted subjects (◇ symbol represents mean).

Considering the number of messages, the data sets are not normally distributed, therefore a Kruskal-Wallis test was conducted. No statistical difference was found between the three groups ($\chi^2 = 0.164, p = 0.921$).

4.6.2 Qualitative Evaluation with Blind Subjects

The qualitative evaluation was conducted in an indoor environment during an exhibition of assistive technologies[10]. The evaluation was conducted with 12 blind subjects.

The evaluation was divided into three phases, similarly to the quantitative evaluation. The learning and practice phases were conducted with the same methodology presented in Section 4.6.1. However, due to the particular context of the exhibition, the measurement phase could not be performed. A questionnaire has been administered instead.

The questionnaire is organized in two sets of Likert-scale items; the first one is derived from the System Usability Scale [17] and is composed of 7 statements related to the ease of use of the three auditory guiding modes (see Figure 4.21). The second one, composed of 8 statements, is derived from IBM Computer Usability Satisfaction Questionnaire (CSUQ) [50] and is aimed at evaluating the satisfaction with the preferred guiding mode, which is specified by the subjects with an answer to a multiple choice question.

There are some topics on which most of the test subjects seem to agree, and others in which there is no consensus. The test subjects agree on the fact that instructions provided with the speech guiding mode are simple to follow (consider item 1 in the first set), and they all seem to have an overall positive view of ZebraX (consider in particular items 1, 2, 7 and 8 in the second set).

There is generally a lower consensus on the items in the first set. For example, test subjects have very different feelings about the ease of following instructions with the mono guiding mode. 8 test subjects state that they are easy to follow (with a rate of 4 or 5) while 4 test subjects do not agree with that statement. Very similar result are obtained for the stereo guiding mode. 8 test subjects state that instructions provided with the stereo guiding mode are easy to follow. Interestingly, only one test subject found the instructions provided with both mono and stereo

[10]HANDImatica 2014, held in Bologna, Italy.

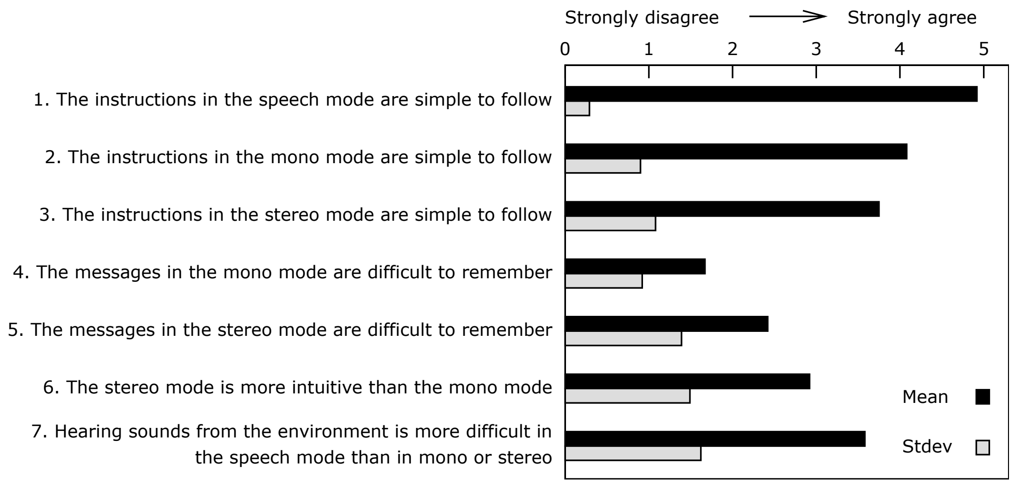


Figure 4.21: Results of the questionnaire administered at the end of the qualitative evaluation, first part.

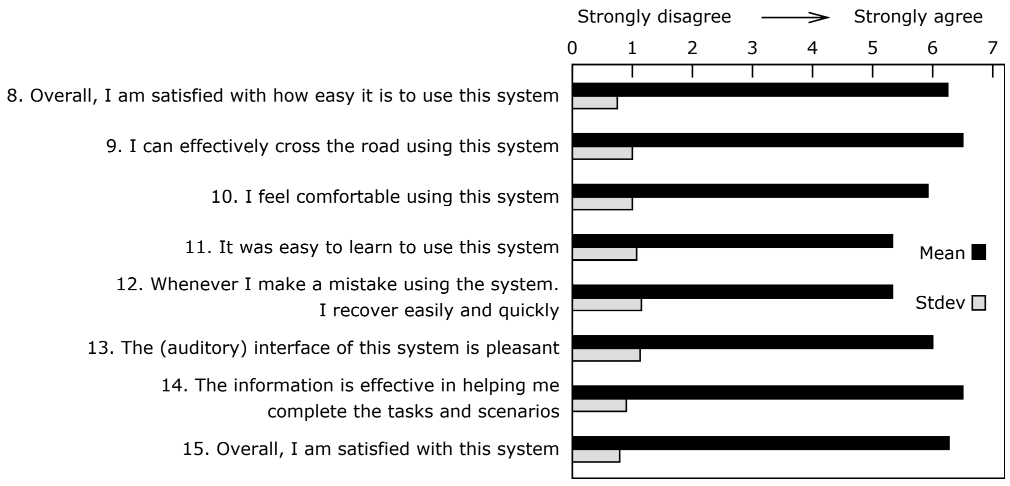


Figure 4.22: Results of the questionnaire administered at the end of the qualitative evaluation, second part.

guiding modes hard to follow. Instead, 6 test subjects found that one of the two guiding modes based on sonification is hard to follow, while the other one is not. This suggests that test subjects have clear and contrasting preferences. To confirm this, 50% of the test subjects state that the mono guiding mode is more intuitive than stereo, while 50% state the opposite.

Three test subjects prefers the speech guiding mode, 4 prefers mono and 5 prefers stereo. Despite this, in the second set of items test subjects converge towards a positive view of ZebraX (see Figure 4.22). Indeed, subjects argue to be satisfied by the ease of use of the application and that they have been able to complete the crossing using ZebraX. However, sometimes users reported to be a little disoriented by contrasting instructions provided by ZebraX. By observing users, we noticed that the issue occurred when users accidentally moved the hand, aiming the camera away from the crossing. Three possibile solutions may be considered as future works to address this issue. First, to warn the user when excessive movement is detected by the IMU, helping her aim back in the right direction. Second, the Logic module may adopt temporal reasoning to filter out wrong detections. Third, wearable devices such as smart glasses and wearable cameras may be used to obtain a steady aim in the correct direction.

4.6.3 Qualitative and Quantitative Evaluation with Test Subjects with VIB

The third evaluation consisted in a quantitative and qualitative evaluation conducted with three test subjects with severe visual impairments.

Evaluation Methodology

The evaluation was conducted with three test subjects: one of them was blind, the other two were partially sighted, and not able to recognize zebra crossing through their residual sight[11].

The evaluation consisted in five phases. The first three phases (learning, practice and measurements) were similar to the quantitative evaluation described in Section 4.6.1.

The fourth phase was conducted in a urban crossroad, and consisted in a set of about 10 crossing attempts. A supervisor was constantly supporting the test subjects, in the attempt to avoid any hazard. At each crossing attempt the supervisor guided the test subject to the crosswalk vicinity, and then asked him/her to align with the crosswalk. Once aligned, the test subject had to wait for the traffic light to turn green (this information was provided by the supervisor) and was then asked to cross. In case the crossing was not complete before the traffic light turned yellow, the supervisor was instructed to guide the test subject towards the sidewalk. No formal measurements were collected during this phase. The goal was simply to allow test subjects to use ZebraX in a real environment.

The fifth phase consisted in the qualitative evaluation described in Section 4.6.2 with an additional set of open questions.

Evaluation Results

During phase three (measurements), all test subjects have been able to successfully complete the crossing in all the attempts. Figure 4.23 shows the time to align and cross. For what concerns the comparison among the three guiding modes, results are not dissimilar to those presented in Figure 4.19. One difference is that, in the case of test subjects with VIB, the average crossing time is about 27s with the three guiding modes. This is more than 10s faster if compared with the performances of blindfolded sighted users. The number of messages is also similar; mean values are 20, 11 and 14 for the three guiding modes respectively. In this regard, we have to underline that test subject 12 (the blind subject) had some problems, at the beginning, finding the correct inclination of the device. This caused a large number of ‘raise’ and ‘lower’ messages in the two runs with the speech guiding mode.

In phase four, all test subjects completed the crossing before the traffic light turned yellow. The test subjects conducted at least one test with each guiding mode, but they were left free to choose how to conduct the majority of tests. All of them choose to use their preferred guiding mode (listed below).

[11]The two partially sighted subjects were blindfolded during the test.

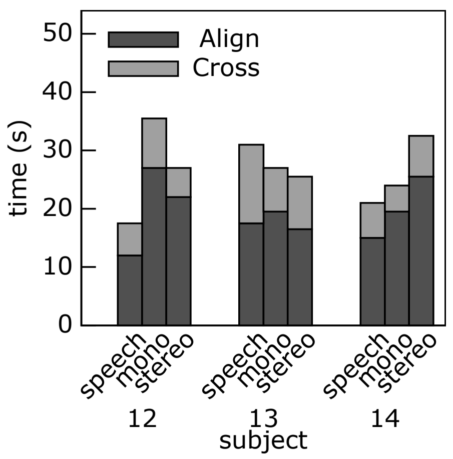


FIGURE 4.23: Crossing time in the 6 tests conducted by each of the 3 test subjects with VIB.

In phase five, it emerged that the three test subjects agreed on the fact that the instructions provided in the speech and the mono guiding modes were easy to follow (for both items, two test subjects rated 7 and the other rated 6). A slightly different score was given to the stereo guiding mode (two test subjects rated 4 and the other rated 3). Vice versa, there is no consensus about how hard it is to remember the sonifications; two test subjects reported that they are hard to remember, while test subject 14 reported the opposite.

Each one of the three test subjects preferred a different guiding mode. Test subject 11 preferred stereo guiding mode, justifying the choice by saying that the stereo guiding mode "provides both the spatial references and the clearness of the speech messages that can be activated by tapping"[12]. Test subject 12 declared to prefer the speech guiding mode because it was less cognitive demanding. This test subject comments that "you need to get used to this app, because when you are crossing you need to pay attention to the surrounding. With the stereo [and mono] guiding mode[s], you need to concentrate to remember the sounds [i.e., the association between the sounds and the instruction], and this may distract you". Finally, test subject 13 preferred the mono guiding mode, reporting these motivations: "I like the other two [guiding modes] as well. Still, stereo [guiding mode] requires me to concentrate, while speech messages can get confused with other sounds in the environment".

Finally, the last questions about the overall satisfaction denoted high satisfaction by all three test subjects.

4.7 Discussion

In this chapter we introduce the ZebraX application and describe the Recognizer, Logic and Navigator modules. The Recognizer module is in charge of recognizing pedestrian crossings from the images captured by the smartphone's camera. The extensive experimental evaluation highlights three major contributions with respect to the state of the art. First, Recognizer removes projection distortion from the features identified in the input image, thus improving the recognition quality both in terms of precision and recall. Second, the module computes the

[12]The interview was conducted in Italian, and only the english translation is reported.

relative distance between the user and the crossing with quantified and precise measures. Third, Recognizer is engineered to adopt GPU parallel computation techniques to deliver realtime recognition on mobile devices.

Regarding the Navigator module, that is responsible for guiding the user in the actual crossing through auditory feedback, a number of discussion points emerge from the analysis of the experimental results and from the experience derived by the observation of the different evaluation stages. It is quite clear that there is no guiding mode which is best suited for all test subjects. While on average the speech guiding mode allowed the test subjects to align and cross more quickly, the majority of test subjects (6 out of 11) were faster to align and cross with one or both the sonification guiding modes. More importantly, test subjects distribute their preferences for the best guiding mode almost uniformly among the three solutions (4 prefers speech, 5 mono and 6 stereo guiding mode).

An important fact to be considered is that, following the results and feedback of the preliminary evaluation stage (Section 4.4.3), the guiding modes have been integrated with touch-activated speech messages. While this functionality clearly facilitates the usability of the application, its implementation essentially changed the nature of the evaluation, which in practice became a comparison between a guiding mode based on speech only and two guiding modes based on the combination of sonification and speech. We expected the test subjects to rely on the tap gesture mainly during the training phase, and then to gradually get used to the sonifications. Nevertheless, we did not observe a statistical significant decrease in the number of tap gestures between the first and second round tasks.

During the tests with the two sonifications, some test subjects frequently tapped on the screen, requesting the speech cue. We believe that these test subjects did not get well acquainted with the sonification technique, and therefore required constant speech feedback in addition to the sonification. For example, during the second round with the stereo guiding mode, test subject 4 tapped on the device almost three times for each new message received (67 taps and 24 messages). Differently, other test subjects used the tap gesture only sporadically. For example, test subject 5 tapped only 2 times in the second round with the mono guiding mode, during which he received 15 messages in total. This indicates that the test subject was confident to have correctly interpreted the great majority of messages.

Interestingly, sighted test subjects frequently used the tap gesture also with the speech guiding mode (more than half of the sighted test subjects used the tap gesture more than once every four messages). The tap gesture seemed to provide a form of confirmation or reminder of the last message read. It was not the same for the three test subjects with VIB, who did not use this functionality with the speech guiding mode.

To judge the applicability of the two sonifications we should also consider that, while they are considered less intuitive (all test subjects believed that at least one of the two sonifications was harder to understand than the speech guiding mode), test subjects still expressed their appreciation for them even after a short practice (11 out of 15 test subjects preferred the mono or stereo guiding mode). This is due to the fact that, according to some of the test subjects, the speech messages prevented hearing of environment sounds. Also, as reported by two test subjects as answers to the open questions, the guiding modes based on sonification conveyed the “quantity” of the expected user movement. This additional information, once appropriately grasped, could further facilitate the alignment and crossing phases.

A further consideration should be made regarding the fact that during the evaluations no learning effect emerged. None of the metrics defined to estimate the time and effort indicated a statistically significant improvement between the two rounds. This could be due to the short duration of the tests. Furthermore, a ‘tiring’ effect could have appeared, considering that the test subjects were required to keep high levels of concentration during the whole evaluation (approximately 20 minutes). Using the speech guiding mode, 6 of the 11 blindfolded test subjects required a longer time to align and cross during the second round if compared with the first one. Similarly, with both mono and stereo guiding modes 5 test subjects required longer time in the second run. Since the sonifications appeared to be less immediate, we initially guessed that they should have taken larger benefit from the learning effect derived by frequent use of ZebraX.

Another aspect to be considered is that the evaluations conducted on the plastic sheet were more challenging than those in the real environment. It is in fact true that when testing the app on the plastic sheet, no specific haptic or audio cue is available. Vice-versa, when crossing on the road there are a number of hints that can help a person with VIB orientate during the crossing, including, for example, the sidewalk and traffic noise, and the feeling of different types of terrains under the feet.

One final remark is related to the unexpected high dispersion of the quantitative results with respect to the mean. The relative standard deviation is 41%, 47% and 40% for speech, mono and stereo guiding modes, respectively. Combining these data with the experience derived from the observation of the experiments, we can highlight two important facts. First, some test subjects are more confident and hence move faster (e.g., test subject 9), while others are more cautious (e.g., test subject 5) and tend to move and rotate more slowly. Second, there are some human errors that can lead one test subject to have different results in two tests with the same guiding mode. For example, test subject 4 completed the two tests with stereo guiding mode in 28s and 104s respectively. In the second round, the test subject misinterpreted a message, believing that the crosswalk was on his right, while actually it was on his left. This caused the align process to take much longer (78s in total) than in the previous round.

4.8 Summary and future directions

In this chapter, we introduce the ZebraX application, a novel solution to guide people with VIB in road crossings. ZebraX is composed of three different modules, Recognizer, Logic and Navigator.

For what concerns the Recognizer module, research presented here extends a previous contribution by Ahmetovic et al. [3] in many directions. First, perspective projection distortion is removed from detected features, improving the overall recognition quality; Second, the relative distance between the user and the crossing is computed, providing quantified and precise measurements that are used by other ZebraX modules to guide the user in the actual crossing; Third, the technique is optimized and engineered in order to improve accuracy and reduce computational complexity, offloading computationally expensive operations from the mobile device’s

CPU to its GPU. The experimental evaluation shows that, in our data sets, the updated Recognizer module outperforms our previous solutions in terms of recall and computation time, while precision is still equal to 1. Also, Recognizer detects crossings even if they are partially out of the camera's field of view, addressing a limitation of a previous approach [77]. Another issue of previous research [77, 85] is that the proposed solutions have not been evaluated on large data sets of zebra crossing images. Recognizer, instead, has been evaluated on a large dataset of about 4000 images captured under different illumination conditions.

A research question left unanswered by previous contributions [77, 85] is to find an effective way to give feedback to the user, providing guidance while crossing without distracting him/her from paying attention to the surrounding environment. The Logic and Navigator modules are introduced to this extent. The Logic module computes instructions to guide the user in the actual crossing, while the Navigator module implements three auditory guiding modes to convey instructions to the user. The first guiding mode adopts speech synthesis to convey instructions, similarly to what is done in previous research [4, 38]. The remaining two modes are novel contributions that adopt sonification and sound spatialization to convey, at the same time, instructions and quantitative information like, for example, the distance of a recognized crossing.

Three evaluations have been conducted with people with VIB and blindfolded subjects in order to evaluate the three auditory guiding modes. All subjects managed to successfully complete all crossings. Interestingly, no best auditory guiding mode emerged from the quantitative evaluation: while on average the speech based guiding mode allowed to cross more quickly, the majority of test subjects were faster with non-speech guiding modes. For what concern subjects preferences, people with VIB declared to prefer the two non-speech guiding modes even if they are less intuitive.

Research presented in this chapter highlights several ideas for future works. First, Recognizer could be extended in order to detect other types of pedestrian crossings. Second, the Logic module should be extended in order to adopt some form of spatial-temporal reasoning to track zebra crossings between consecutive frames, hence mitigating the impact of wrong detections. Also, more evaluations in real world usage scenarios should be performed. A possible way to reach this objective, is to integrate ZebraX in publicly available mobile applications to support navigation of people with VIB[13]. Such integration would enable the collection of real world usage data to further validate the technique and to perform longitudinal studies that may show how training impacts on the performance of the auditory guiding modes. Finally, from a usability perspective, we observed that keeping the phone steady was difficult and sometimes tiresome for test subjects. The adoption of wearable devices such as smart glasses or wearable cameras should be considered in order to provide a better way to aim the camera without putting too much strain on the user.

Results highlighted in this chapter show that MATs based on computer vision can be effectively used to support navigation of people with VIB. The same approach can be used to address other challenges encountered while navigating unknown environments. An example is the detection of the state of traffic lights, a challenge that we approach in the next chapter.

[13]An example of such application is *iMove*, a commercial application that supports orientation of people with VIB developed by EveryWare Technologies.

Chapter 5

Recognizing traffic lights

The ZebraX application described in Chapter 4 proved to be effective in supporting people with VIB when crossing roads: it supports users in finding the crossing, in aligning to it and in staying inside its boundaries while crossing. However, the application has a limitation, as ZebraX has no way to detect when it is safe to cross. A solution to this problem consists in crossing at intersections equipped with acoustic traffic lights. There are many different models of acoustic traffic lights. For example, in Italy, there are acoustic traffic lights that produce sound on demand by pushing a button placed on the pole. The sound signals to the person with VIB when the light is green. In Germany, there are models that always produce a sound when the light is green (no button has to be pushed) and they adapt the intensity of the sound according to the background noise.

Nonetheless, as reported by many associations for blind and visually impaired persons, in most industrial countries (e.g., Italy, Austria, France, Germany, etc.), acoustic traffic lights are not ubiquitous; they are present in some urban areas but may be absent in small towns. Furthermore, acoustic traffic lights are not always working properly because damages often take a long time to be reported and fixed. The situation can be even worse in developing countries.

A number of solutions have been proposed in the scientific literature to recognize traffic lights, however all of them share a common problem: they use images acquired through the device camera with automatic exposure. With this approach, in conditions of low ambient light (e.g., at night) traffic lights result overexposed (see Figure 5.1) while in conditions of high ambient light (e.g., direct sunlight) traffic lights are underexposed (see Figure 5.2).

In this chapter we present *TL-detector*, a traffic light recognition system that solves the above problem with a robust image acquisition method, designed to enhance the subsequent recognition process.

5.1 The target of the detection

In this work we consider traffic lights currently used in Italy, which adhere to European Standard 12368 [83]. This standard specifies a number of physical properties of the traffic lights, including,

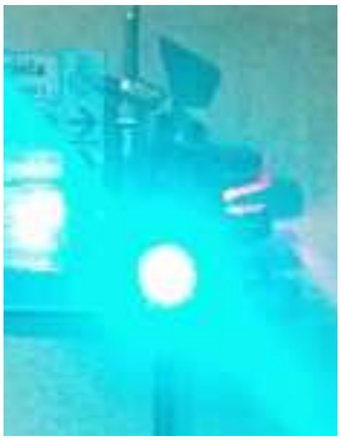


FIGURE 5.1: Pedestrian traffic light is overexposed.

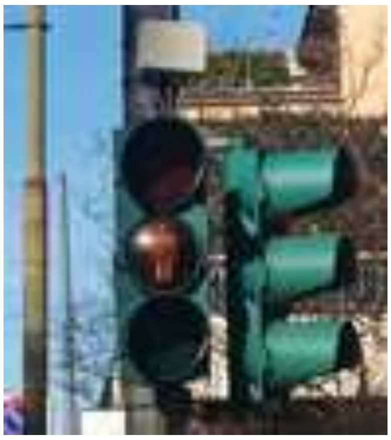


FIGURE 5.2: Pedestrian traffic light is underexposed.

for example, their size, luminous intensities and colors that have to be consistent in all European countries. Luminous intensities are specified in two classes, with a common minimum and two maxima according to the class. Values are different according to the color and are reported in Table 5.1.

	red	yellow	green
min	$100cd$	$200cd$	$400cd$
Max Class 1	$400cd$	$800cd$	$1000cd$
Max Class 2	$1100cd$	$2000cd$	$2500cd$

TABLE 5.1: Luminous intensities range in the reference axis according to European Standard 12368 [83].

Chromaticities are delimited in the CIE XYZ space according to the values reported in Table 5.2.

	chromaticity boundaries	boundary
red	$y = 0.290$	red
	$y = 0.980 - x$	purple
	$y = 0.320$	yellow
yellow	$y = 0.387$	red
	$y = 0.980 - x$	white
	$y = 0.727x + 0.054$	green
green	$y = 0.726 - 0.726x$	yellow
	$x = 0.625y - 0.041$	white
	$y = 0.400$	blue

TABLE 5.2: Chromaticities range according to European Standard 12368 [83].

In Italy, as in many other countries, differently shaped lights are used to transfer messages to different classes of road users. For example, the rounded light is used for drivers, while the "body-shaped" light is used for pedestrians. Two different shapes are used in Italy for pedestrians lights: one for green light, the other for yellow and red lights (see Figures 5.6, 5.7 and 5.8). While the actual shape of the figure appearing through the lens can vary from country to country (in some cases even within the same country), the proposed solution can be easily adapted to most existing

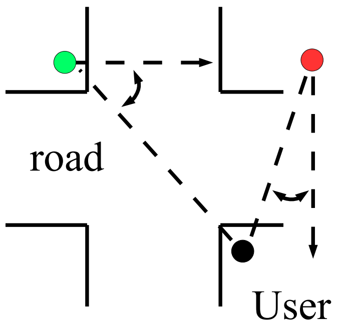


Figure 5.3: Example of 'maximum rotation angle'.

standards by simply re-tuning the detection parameters and by using different template images (see Section 5.3.4). Also, if the proposed technique is used in countries with very particular light conditions (e.g., a bright sunny day in the desert) it could be necessary to accordingly tune the acquisition parameters with the methodology presented in the following.

Among other physical properties of the traffic light, its position with respect to the observer is particularly relevant. Indeed, given the application, only traffic lights with bounded distance from the observer should be detected. For example, considering the width of urban roads, in the experiments the minimum and maximum horizontal distances adopted are 2.5m and 20m, respectively. Analogously the signal head should not be too high or too low with respect to the observer. Hence the vertical distance is bounded. For example, in the experiments the minimum and maximum vertical distances adopted are 0.5m and 4m, respectively. Finally, the user is interested only in the traffic lights that point towards him/her. Consider for example Figure 5.3: the direction of the red traffic light (red circle) is roughly the same angle as the line passing through the traffic light and the user (black circle). Hence, that traffic light should be detected. Vice versa, the green traffic light (green circle) is pointing away from the user and hence it should not be detected. The "maximum rotation distance" parameter defines the angular distance between the direction of the traffic light and the direction from the traffic light towards the user. In the experiments a "maximum rotation distance" of 45° is adopted. In a typical crossroad like the one in Figure 5.3, this value prevents the identification of a diagonally opposite traffic light that, generally, shows an opposite color with respect to the one shown by the traffic light the user is interested in.

Henceforth some of the terms defined in European Standard 12368 [83] are used. In particular, the "signal head" (see Figure 5.4) is the device composed by different "optical units" (see Figure 5.5), each one with its "lens". For example, in Italy, there are three optical units in each signal head. The "background screen" is the opaque and dark board placed around the optical units to increase the contrast. Also, the term "active optical unit" (AOU in the following) refers to the optical unit that is lighted in a given instant (as in Figure 5.5). Finally, "optical unit color" is the color of an optical unit when it is active. Examples of different visual appearances of the AOU are shown in Figures 5.4 to 5.8.

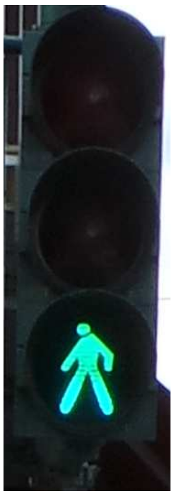


FIGURE 5.4: Signal head.

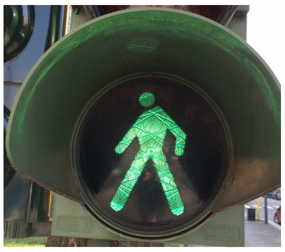


FIGURE 5.5: (Active) optical unit.

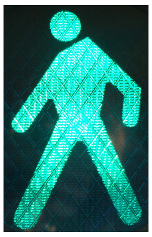


FIGURE 5.6: Green AOU.

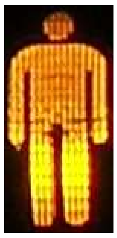


FIGURE 5.7: Yellow AOU.

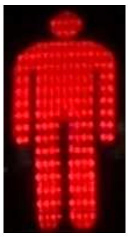


FIGURE 5.8: Red AOU.

5.2 The TL-detector application

The architecture of TL-detector is similar to the one adopted for ZebraX (described in Chapter 4) and is made of three main modules: *TL-recognizer*, *TL-logic* and *TL-Navigation* (see Figure 5.9).

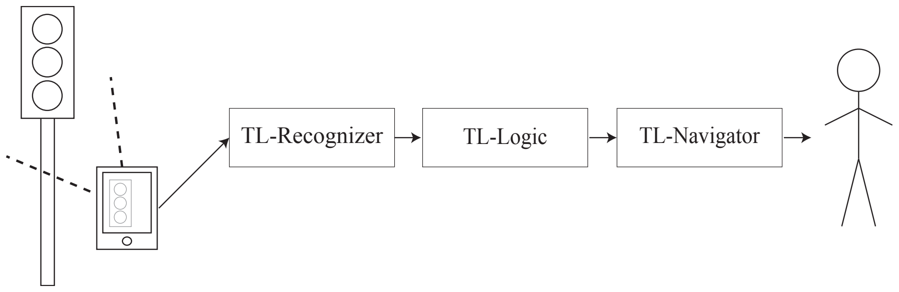


FIGURE 5.9: Structure of the main application modules.

The TL-recognizer module computes the position and color of a pedestrian traffic light in a given image, relying on data sources available on off-the-shelf smartphones. Similarly to ZebraX, TL-recognizer uses the camera to capture video frames that can then be analyzed with computer vision techniques and inertial sensors.

The TL-logic module combines different results of TL-recognizer and computes messages to guide the user. Example 5.1 shows a simple form of reasoning.

Example 5.1. *One run of TL-recognizer detects a red traffic light in a certain position. TL-logic computes a 'wait' message to instruct the user not to cross. After the recognition, TL-logic uses accelerometer and gyroscope data to estimate how the device is being moved and hence where the traffic light is expected to be in the next frame. Indeed, the following run of TL-recognizer identifies a green traffic light in the expected position. Consequently TL-logic can conclude that the traffic light has now turned green and therefore generates a 'cross' message for the user.*

The TL-Navigation module is in charge of conveying the messages to the user through audio, haptic (vibration) and graphical feedback. This module must address the challenge of providing audio information to the user without diverting the user's attention from the surrounding audio scenario, which is essential to acquire indispensable information (e.g., an approaching car, a person walking by, etc.).

In the following, we focus on the recognition process adopted by the TL-recognizer module. The TL-logic and TL-Navigation modules are implemented with basic functionality in order to allow the conduction of human-based tests and their extension is left as a future work.

5.3 The recognition process

The recognition process is organized in five steps (see Figure 5.10). During image acquisition a frame is captured by the device camera using specifically designed exposure parameters. The horizon computation step adopts the same technique illustrated in Section 4.2.1 to compute the equation of the horizon line in the image reference system.

The other three steps are aimed at identifying the AOUs that appear in the image. The overall computation is presented in Algorithm 1 and is logically divided into: extraction of candidate AOUs, pruning of candidate AOUs and validation of AOUs.

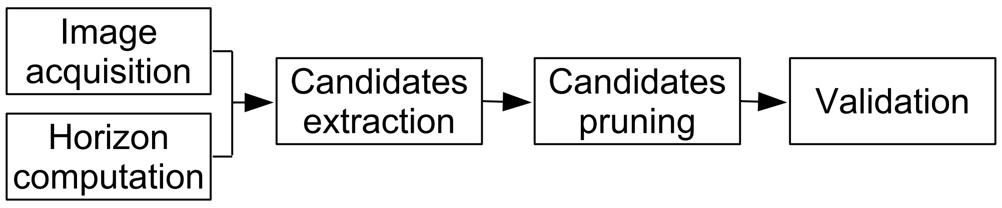


FIGURE 5.10: Organization of the recognition process

The image-processing algorithm takes in input the results of the acquisition phase: an image i (encoded in the HSV color space) and the horizon line equation h. There are other system parameters that form the algorithm input: three range filters f_g, f_y and f_r, one for each optical unit color; three template images t_g, t_y and t_r, each one representing the three lenses and, finally, a threshold value $T \in (0, 1)$ used in the validation step. The output of the algorithm is a set of identified AOUs, each one represented by its color and its contour in the input image.

5.3.1 Image acquisition

The exposure of the image to be acquired is a key point. Light conditions during day and night are extremely variable, while luminance coming from traffic lights is pretty stable. Since smartphone camera automatic exposure balances the mean luminance of every point in the entire image, its use can result in underexposed or overexposed AOUs (see Figures 5.1 and 5.2). For this reason, the proposed solution disables the automatic exposure feature of the mobile device and sets a fixed exposition value (EV) chosen among a small group of EVs that have been pre-computed to encompass the luminance variations. These variations are mainly due to traffic

Algorithm 1: Image processing (non optimized version)

Input: image i; horizon line equation h; range filters f_g, f_y and f_r; template images t_g, t_y and t_r; threshold value $T \in (0,1)$.
Output: a set R of active optical units. Each element of R is a pair $\langle o, c\rangle$ where o is the AOU contour and c the color.
Constants: g, y and r represent the three optical unit colors (i.e., green, yellow and red).
Method:

```
1:  $R \leftarrow \emptyset$ {algorithm result}
2:  for all (color $c \in \{g, y, r\}$) do
3:     {Extraction of candidate AOU}
4:     $i' \leftarrow$ apply $f_c$ to $i$ {$i'$ is a binary image}
5:     $O \leftarrow$ extract the set of contours from $i'$
6:     for all (contour $o \in O$) do
7:        {Pruning of candidate AOU}
8:        $o' \leftarrow$ rotate $o$ by the inverse of the inclination of $h$
9:        if ($o'$ does not satisfy “distance” or “width” properties) then
10:          continue {prune $o$}
11:       end if
12:       {Validation}
13:       $p \leftarrow$ image patch, extract from $i$, corresponding to the MBR of $o'$
14:       $p \leftarrow$ resize $p$ to have the same size of $t_c$
15:       $\alpha$ is the result of normalized cross correlation between $t_c$ and $p$
16:       if ($\alpha > T$) then add $\langle o, c\rangle$ to R
17:    end for
18: end for
```

light class (see Section 5.1), and acquisition noise due to distance, misalignment, veiling glare, pixel saturation etc.

Before selecting candidate EV values, the intensity and chromaticity of light coming from a set of traffic lights were empirically verified. Table 5.3 reports the values measured for four of them, as an example of the high variability.

traffic light number	AOU color	Lux	x	y
1	green	2671	0.0875	0.6075
	yellow	1138	0.5839	0.4155
	red	740	0.7068	0.293
2	green	491	0.2785	0.495
	yellow	1199	0.5676	0.4471
	red	723	0.6568	0.3425
3	green	754	0.2193	0.5025
	yellow	1502	0.5755	0.4129
	red	955	0.6854	0.3142
4	green	1941	0.0727	0.5091
	yellow	2065	0.587	0.4121
	red	1082	0.7048	0.2951

Table 5.3: Intensity and chromaticity of four sample traffic lights.

Although the standard for traffic light luminous intensity is clearly defined, variability in the real world (i.e., in the streets) can be very high, both in terms of illuminance and chromaticity. The reasons are many: class (see Section 5.1), technology of light bulbs, dirt on the lens, aging, etc.

To identify the correct EV, a series of pictures were taken at different times of the day and distances, starting from the theoretical EV computed from the European Standard luminous intensity [83] on a ±5 stops bracketing, with step 1. From this set of shots, a subset of EVs were selected to cover the major part of the variance of correctly exposed lenses, in four light conditions.

The four light conditions are: very high light intensity (e.g., a sunny day at noon), high light intensity (e.g., a partially cloudy day at noon, or a clear day when the Sun is not high in the sky), mid light intensity (e.g., a cloudy day, or a clear day at dawn or dusk), low light intensity (e.g., night). Note that, for our purposes, light condition is highly influenced by the time of day and by weather conditions (e.g., sunny, cloudy, etc...), while other meteorological conditions (like rain) do not affect light intensity. To automatically identify the light condition, the following approach is adopted: before starting recognition, a picture is taken with fixed camera parameters (ISO 100, aperture F8.0, shutter speed 1/125). Then, value M is computed as the mean, for each pixel, of the V channel from the HSV color space. This value characterizes the light condition. Table 5.4 shows how light conditions are specified as well as the camera parameters that yield best shots in each of them. It may appear counterintuitive but at night time the exposition is shorter; this reduces the optical veiling glare on the edges of the body shaped lens.

Light intensity	M	ISO	Aperture	Shutter speed
Very High	$120 < M$	100	F8.0	1/160
High	$60 < M \leq 120$	100	F8.0	1/200
Mid	$5 < M \leq 60$	100	F8.0	1/250
Low	$M \leq 5$	100	F8.0	1/500

TABLE 5.4: EV parameters yielding the best results for each light intensity.

In order to adopt the proposed approach to image acquisition, a device must provide camera APIs to manually set a specific combination of ISO value, shutter speed and aperture settings. However we observed that, on our target platform, this isn't always possible. Image acquisition with fixed EV was implemented on both Android 4.x and Android 5.x. With Android 4.x it is possible to set the values for ISO, shutter speed and aperture through the `Camera.Parameters` object[1]. It should be observed that, while the `Camera.Parameters` object is defined for all Android APIs up to level 21 (excluded), not all of its methods produce effects on all devices. Indeed, on most devices the methods to manually set ISO, shutter speed and aperture do not produce any effect and do not disable auto exposure.

Android 5.x offers different APIs to access the camera and its parameters. The package containing the classes is called `Camera2` [2]. These classes offer several new APIs to control camera parameters and, based on our experience, these APIs are actually supported by most devices, including the Nexus 5, which was used for the experiments.

A final comment on gamut spaces. The high variability in terms of both European standard ranges and actual measured chromaticities of the AOUs (see Table 5.3) turned out to be wider than the average image variance due to possible changes of gamut space in the acquisition device. Thus, varying the parameter settings (see Section 5.5) is sufficient to compensate this variance.

[1] http://developer.android.com/reference/android/hardware/Camera.Parameters.html
[2] https://developer.android.com/reference/android/hardware/camera2/package-summary.html

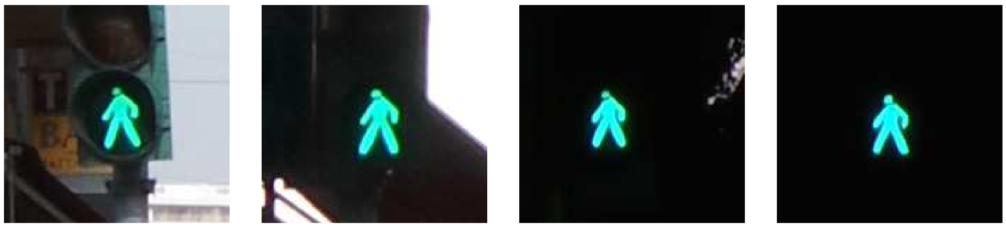


Figure 5.11: Details of four pictures taken in different illumination conditions.

Figure 5.11 shows details of four pictures, each one representing a green AOU in a different illumination condition. The pictures were taken with the camera parameters described above. From left to right, the four light intensities are: very high, high, mid, and low. These results are examples of the stable acquisition (see Figures 5.1 and 5.2 for a visual comparison with automatic exposure).

5.3.2 Extraction of candidate active optical units.

After image acquisition, for each optical unit color c (i.e., green, yellow and red), TL-recognizer identifies a set of image portions, each one representing a candidate AOU. To achieve this, the proposed technique first applies a range filter and then groups contiguous pixels. This approach relies on the fact that AOUs have high luminosity values and are surrounded by regions with low luminosity values (i.e., the optical unit background).

The range filter is defined over the HSV image representation and is used to identify the pixels with high luminosity values (see Line 4 in Algorithm 1). A different filter is defined for each optical unit color c. We observed that optical units in Italy adopt two different types of illuminants, either an incandescent light bulb or LED lamps. Optical units adopting an incandescent light bulb sometimes tend to present brighter colors in the center. Range filter boundaries are designed to account for such issue. The result of the application of the range filter is a binary image whose white pixels are segmented into blocks of contiguous pixels (see Line 5). This is obtained through the technique proposed by Suzuki and Abe [80]. The result is a list of contours, each one composed of a set of points.

Example 5.2. *Consider the portion of image shown in Figure 5.12a. Figure 5.12b shows the application of the range filter for the yellow optical unit color on the H channel. Figures 5.12c and 5.12d shows the same filter for the S and V channels, respectively. Details on the filter ranges are provided in Section 5.5. Figure 5.12e shows the logical conjunction of the previous three figures, i.e., the result of the range filter. Finally, Figure 5.12f shows the contours extracted from the image.*

A possible optimization that may reduce the computational complexity of the technique, already adopted in a previous contribution [39], consists in looking for candidate AOUs only in the subset of the image situated above the horizon, where they are most likely to be found. The implementation of such optimization is left as a future work.

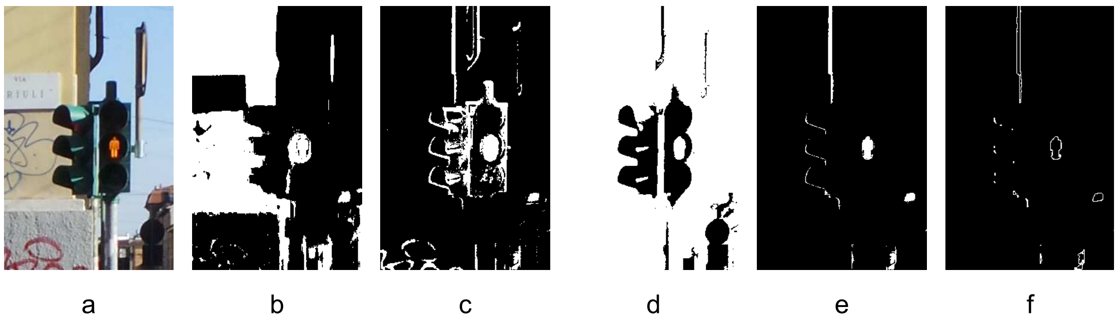


FIGURE 5.12: Extraction of candidates AOUs. (a) Portion of original image, (b) filter on H, (c) filter on S, (d) filter on V, (e) conjunction of filter results, (f) extracted contours.

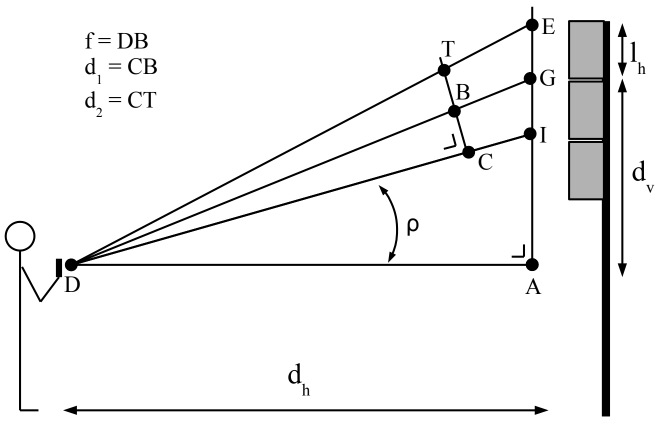


FIGURE 5.13: "Distance".

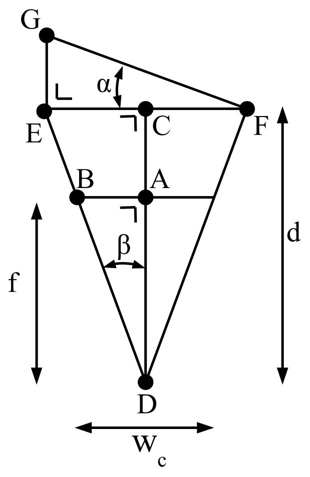


FIGURE 5.14: "Width".

5.3.3 Pruning of candidate active optical units.

After extracting the contours from the source image, the algorithm removes the contours whose geometrical properties are not compatible with those of an AOU. This pruning phase helps prevent false positives and it also improves computational efficiency, as it reduces the number of times the validation process needs to be run. Pruning is based on two properties: "distance" and "width".

The "distance" property is based on the idea that the optical units to be recognized should not be too far or too close from the user (see Section 5.1). To capture this intuition, each contour is assumed to be an AOU (whose size is known). Then, its distance along the horizontal and vertical axes from the device camera is computed. These distances are then compared with threshold values and, if the AOU is too close or too far away along any of the two axes, the contour is discarded. Property 3 shows how to compute the horizontal and vertical distances.

Property 3. Let ρ be the device pitch angle, d_1 and d_2 the directed minimum and maximum vertical distances between the contour and the center of the image (in pixel), f the focal distance (in pixel), l_h the height of the optical unit lens (see Figure 5.13 for a graphical representation). The horizontal and vertical distances (d_h and d_v, respectively) between the device and the optical unit are:

$$d_h = \frac{l_h \cdot \cos(\arctan(d_2/f) + \rho) \cdot \cos(\arctan(d_1/f) + \rho)}{\sin(\arctan(d_2/f) - \arctan(d_1/f))} \tag{5.1}$$

$$d_v = d_h \cdot \tan(\arctan(d_1/f) + \rho) \tag{5.2}$$

There are two aspects related to the "distance" property that are worth observing. First, the formulae are based on the contour height, which is computed after rotating the contour by the inverse of the horizon inclination. This makes the proposed technique 'rotation invariant' in the sense that it is not affected by accidental rotation of the device. The reason for using the height as the reference length is that, by using the device accelerometer, it is possible to compute the device pitch (i.e., the inclination with respect to the ground) that is then used to compensate for projection distortion. The second aspect is that, in practice, the "distance" property checks the vertical size of the contour and discards the contours that are too small or too big. Indeed, small contours correspond to potential AOUs that are too distant from the user, hence not relevant for the recognition. Analogous reasoning can be applied for contours that are very large.

The "width" property is used to prune all contours whose width is not compatible with the width of an optical unit lens. Property 4 shows how to compute the width of the object represented by the contour. Note that distance d between the camera and the traffic light is easily computed from d_h and d_v.

Property 4. Let w_c be the contour width, f the camera focal distance (in pixel), α the angular distance between the image plane and the plane of the optical unit lens and d the distance between the camera and the optical unit. The width of the object represented by the contour is:

$$w = \frac{d \cdot w_c}{f \cdot \cos(\alpha)} \tag{5.3}$$

There is a major difference with respect to the computation of the "distance" property: the relative angle α between the image plane and the plane of the optical unit lens (see Figure 5.14) is not known. Consequently it is not possible to compute the exact width of the contour, but it is possible to bind it in a range. The minimum value of the range represents the case in which α is zero (i.e., the device camera is pointing directly towards the traffic light), while the maximum value represents the situation in which α is equal to the 'maximum rotation distance' (see Section 5.1). If the width of the optical unit lens (which is known) is not contained in the range, the contour is pruned.

5.3.4 Validation of active optical units.

Each contour that passes the pruning step has geometrical properties compatible with an AOU; still, it is not guaranteed that it actually represents an AOU. To validate a contour, the proposed solution extracts from the input image the image portion (called "patch", in the following) corresponding to the contour minimum-bounding rectangle (MBR).

Note that the contour is rotated (see Algorithm 1 Line 8). For this reason, in theory, it should be necessary to apply the same rotation to the original image before extracting the patch. Since it is computationally expensive to rotate the entire image, the patch is rotated on-the-fly when it is constructed.

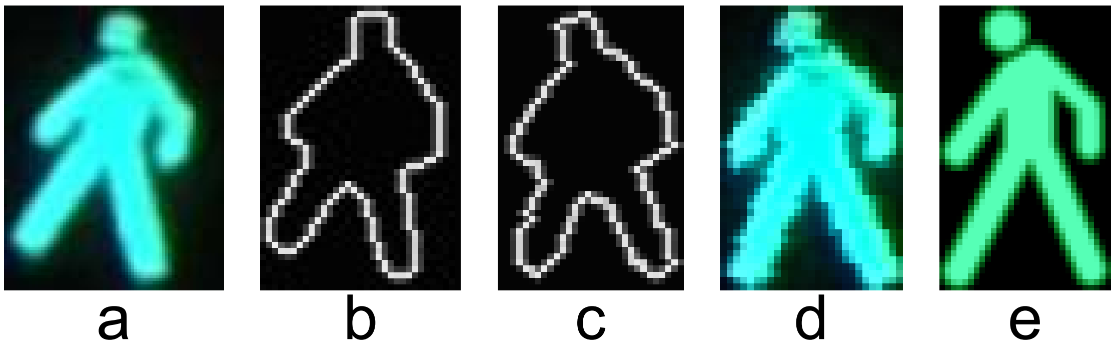


FIGURE 5.15: Validation of candidates AOUs. (a) Portion of original image, (b) Contour, (c) Rotated Contour, (d) Image Patch (rotated), (e) Template image.

The patch is then resized to the same size as the template, which is a system parameter. Finally, the two figures (patch and template) are compared with the fast normalized cross-correlation technique [51], chosen as the technique to evaluate the similarity between two images. The patch is considered to be an active optical unit if the result of the comparison is larger than a given threshold T (see Line 16 in Algorithm 1). The methodology adopted to select the threshold is described in Section 5.5.

Example 5.3. *Figure 5.15a shows a portion of an original image. Figures 5.15b shows the contour, as extracted during the extraction step, while 5.15c shows the rotated contour computed during the pruning step. Figure 5.15d shows the extracted patch. Note that the extracted patch is smaller than the template shown in Figure 5.15e (in the figure they are shown with the same size, but the patch has a smaller resolution). For this reason the patch is first resized to have the same size as the template and then the two images are compared. In this example, fast normalized cross-correlation returns a value of* 0.82 *that, as shown in Section 5.5 is larger than* T*, hence the contour is recognized as a green AOU.*

5.4 Algorithm improvements

In addition to the core recognition procedure described in Section 5.3, the proposed technique implements a number of improvements aimed at increasing both the reliability of the results and computational performances.

5.4.1 Improving recognition of red and yellow AOUs

As shown in Section 5.5, the boundaries of the range filters for the red and yellow colors overlap. As a consequence, it is relatively frequent that a red AOU is confused with a yellow one, and vice versa.

To avoid this problem, the following optimization is introduced. The main loop starting at Line 2 (see Algorithm 1) is iterated for two colors only (instead of three): green and 'yellowRed', i.e., a single color representing both red and yellow AOUs. To distinguish between red and yellow AOUs, a procedure is run during the validation phase, after extracting the patch p (Line 13). This procedure counts, in the patch p, the number of pixels with a purely red hue ($160 \leq h \leq 179$)

and those with a purely yellow hue ($10 \leq h \leq 30$)[3]. If the number of red pixels is larger than the number of yellow ones, the patch is then assumed to be red and is compared with the red template. Otherwise the patch is assumed to be yellow.

As shown in Section 5.5, this approach helps reducing the number of cases in which yellow and red AOUs are confused.

5.4.2 Improving computational performance

As shown in Section 5.5, the computation time of the base recognition algorithm is about 1s on a modern smartphone (with maximum image resolution). While a delay of about 1 second in the notification of the current traffic light color could be tolerable, an additional problem arose during preliminary experiments: it is challenging, for people with VIB, to point the device camera towards the traffic light. To find the correct position, users needs to rotate the device left and right while paying attention to the device feedback (audio or vibration). This requires a responsive system and a delay of 1 second is not tolerable as it does not allow the user to find the traffic light position.

To speed up the computation, two different techniques are adopted: multi-resolution processing and parallel computation. Multi-resolution processing is based on the idea that the validation step requires to process images at a high resolution, while the extraction and pruning steps are reliable (in terms of precision and recall) even when images are processed at a smaller resolution. Running these two steps with images at a smaller resolution significantly improves the performances. For this reason, a resized version of the acquired image is processed during the extraction and pruning steps. Then, during the validation step, the image patch p is extracted (see Line 13) from the original high-definition image. 'Resize factor' is the parameter that defines to what extent the original images is resized. Technically, the number of pixels on both sides of the original image is divided by 'resize factor'. As shown in Section 5.5, this optimization drastically reduces the computation time. However, large values of the resize factor negatively affect algorithm recall, so the value of the resize factor should be carefully tuned.

Since modern smartphones have multi-core CPUs, a natural approach to improve the performance of computational intensive operations is to adopt parallel computation. In particular, two pools of threads are used: one aimed at parallelizing the extraction process (Algorithm 1, Line 2), the other aimed at parallelizing contours' processing (Algorithm 1, Line 6). The former pool has a number of threads equal to the number of colors, while the latter has a number of threads equal to the number of CPU cores.

5.5 Parameters tuning and experimental evaluation

Two main sets of experiments were conducted: one set, called "computational-based" is aimed at tuning the system parameters and at quantitatively measuring the performances of TL-recognizer. The second set, called "human-based" is aimed at qualitatively asserting the effectiveness of the proposed technique.

[3]Henceforth hue scale is reported in $[0, 180)$.

5.5.1 Experimental evaluation methodology and setting.

In order to ease the development of TL-recognizer and to guarantee reproducibility of the computational-based experiments, the following methodology was adopted: images of urban scenarios were recorded, each one with its associated information representing device orientation[4]. Each image was manually annotated with the position and the color of AOUs (if any). Finally, an Android app was implemented to read the stored images and to use them as input for TL-recognizer.

Two datasets of images each were created. The exposition of all the collected images has been chosen according to the methodology described in Section 5.3.1. The "tuning" dataset (501 images), was used for debugging and parameters tuning, while the "evaluation" dataset (1,252 images) was used for performance measurement. Both datasets are divided into four subsets, one for each illumination condition defined in Section 5.3.1. Details are reported in Table 5.5. The two datasets are publicly available[5]. Note that some of the pictures (in particular with mid and low illumination conditions) were taken while it was raining and results are not affected by this weather condition.

Set	Light intensity	Number of images with			
		no AOU	green AOU	red AOU	yellow AOU
Tuning	Very High	62	21	22	22
	High	62	21	21	19
	Mid	62	21	21	22
	Low	62	21	21	21
Evaluation	Very High	75	62	45	37
	High	105	96	104	52
	Mid	64	78	109	59
	Low	120	51	118	77

Table 5.5: Composition of the two sets of images.

During the computer-based experiments, a number of parameters were measured, including: precision, recall, computation time and number of "R-Y errors", i.e., the number of times a yellow AOU is confused with a red AOU or vice versa. Note that, from the point of view of a person with VIB that is about to cross a road, a yellow AOU has the same semantic as a red AOU i.e., the person should not start crossing. For this reason, when computing precision and recall, a R-Y error is still considered a true positive result. Note that, unless otherwise specified, **precision is always equal to one**, meaning that no traffic light is erroneously detected. Finally, note that computation time is measured excluding the time needed to acquire the input image.

To conduct human-based experiments TL-recognizer was implemented into TL-detector, a mobile application that collects live input from the camera and the accelerometer and that implements basic versions of the TL-logic and TL-Navigation modules. The application continuously runs TL-recognizer on the acquired frames and creates three messages for the user: 'not found', 'stop' and 'go': the first indicates that no traffic light was found, the second indicates that a red or yellow AOU was detected and the third one indicates that a green AOU was detected. To convey these messages, the application uses spoken messages (through the system text-to-speech

[4]Henceforth, the term "image" refers to the actual image with the associated device orientation information.
[5]http://webmind.di.unimi.it/CVIU-TrafficLightsDataset

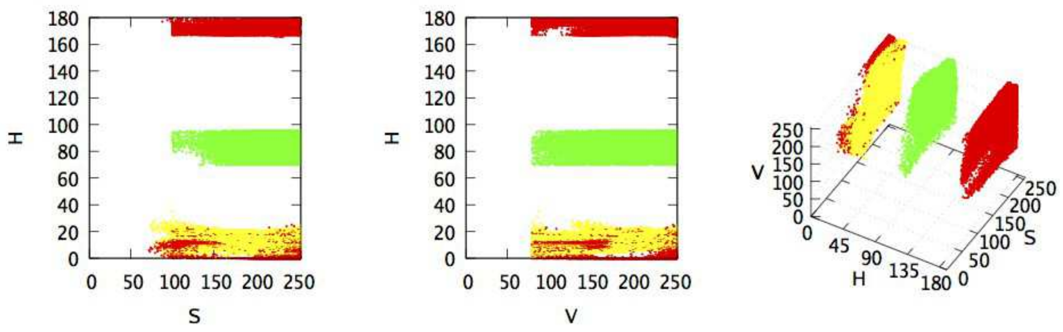


FIGURE 5.16: Pixels composing AOUs.

synthesizer), two clearly distinguishable vibration patterns (for 'stop' and 'go' messages) and a visual message for subjects that are partially sighted (the entire screen becomes black, red or green).

The experiment involved 2 blind subjects and 2 low-visioned subjects (unable to see the traffic lights involved in the experiment). The experiments took place in different illumination conditions. All subjects have been trained for about one minute on how to use the application. Then, in a real urban intersection, subjects were asked to walk towards a crossroad and to determine when it was safe to start crossing in a given direction (straight, left or right) i.e., when a green traffic light appears right after a red one. For each attempt, a supervisor recorded whether the task was successfully completed and took note of any problem or delay in the process. Each subject repeated this task five times. Finally, the subjects were asked to answer a questionnaire.

For what concerns the devices used during the experiments, the images were collected with a Samsung Galaxy Camera with Android 4.1. Computer-based and human-based experiments were conducted with a Nexus 5 device with Android 5, which, with respect to a Galaxy Camera, has a faster CPU and is also more ergonomic for the subjects involved in the human-based tests[6].

5.5.2 Parameters tuning.

The recognition technique presented in Section 5.3 uses several system parameters that need to be tuned. Section 5.3.1 describes the tuning process of image acquisition parameters. Other parameters that mainly affect system performance, are tuned as described in the following.

One set of parameters defines the boundaries of the range filters (see Algorithm 1). To tune these values each pixel composing AOUs (if present) was sampled in the 501 pictures composing the tuning dataset. This was obtained with a semi-automated process: first, a few pixels were manually sampled, hence defining broad ranges. Then, by running the algorithm with these ranges, a set of contours representing the AOUs were extracted, together with contours representing other objects. Thanks to picture annotations, the contours representing AOUs were automatically identified and the values of all pixels included in these contours were stored. White pixels (i.e., $v = 255$) and dark pixels were excluded from this set.

[6]The choice of using a Galaxy Camera to collect images was driven by the fact that, at that time, it was the only available device that allowed to manually define exposure settings.

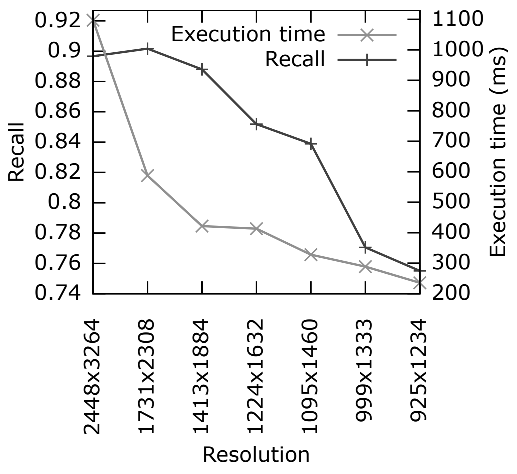


FIGURE 5.17: Impact of image resolution on computation time and recall.

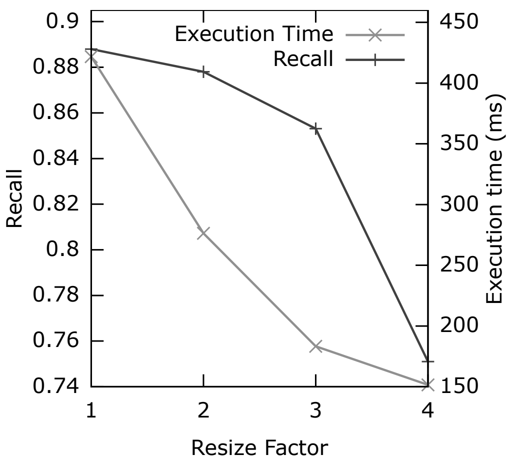


FIGURE 5.18: Impact of resize factor on computation time and recall.

The selected pixels are shown in Figure 5.16 where green, red and yellow dots represent a pixel for a green, red and yellow AOU, respectively. Given these results, the smallest ranges to include all pixels were defined . Results are shown in Table 5.6. Note that, since the yellowRed color lies on both sides of the hue circular axis, two range filters are defined and their disjunction yields the result.

Optical unit color	H min	H max	S min	S max	V min	V max
Green	70	95	100	255	80	255
yellowRed (first)	0	25	100	255	80	255
yellowRed (second)	166	180	100	255	80	255

TABLE 5.6: Range filters boundaries.

Threshold T is another important parameter that requires to be tuned. The following methodology was adopted: the image processing algorithm was run for each image in the tuning dataset. For each extracted patch (see Algorithm 1) the value of the normalized cross correlation was stored, together with a boolean value representing whether the patch is actually an AOU or not (this is derived from the annotations). Among all patches in all images in the tuning dataset, the larger cross correlation value for a patch that does not represent an AOU is 0.586. Threshold T is set to this value hence guaranteeing, in the tuning dataset, a precision of 1.

Figure 5.17 shows the impact of the resolution on both recall and computation time. As expected, computation time decreases almost linearly, since most of the costly operations are linear in the number of pixels in the image. At the same time, recall slowly decreases when using images with up to 3 times less pixels (i.e., 1413×1884) that guarantee a recall of 0.887. With smaller images, recall decreases at a faster rate. For these reasons, images with a resolution of 1413×1884 were used in the tests. Note that, while in the tests the images are resized from their original size to 1413×1884, in the TL-recognizer prototype this operation is not necessary: images are directly acquired at a similar resolution (i.e., 1536×2048) and this also significantly speeds-up the image acquisition process.

5.5.3 Impact of the algorithm improvements

With the basic version of the algorithm, the proposed technique incurs in the 'R-Y error' in 20 cases in images from the tuning set. This means that, considering only the 168 images containing red and yellow AOU, the frequency of this error is above 10%. By using the improvement described in Section 5.4.1, the number of these errors is reduced by 75% with 5 errors and a frequency of less than 3%.

Figure 5.18 shows computation time and recall for different values of the resize-factor parameter. As expected, there is a trade-off between computation time and recall (this is very similar to what was observed for the resolution parameter). By observing the results shown in Figure 5.18 it is possible to conclude that value 3 is a good trade-off: computation time is halved (with respect to value 1), while recall decreases only by 0.03. For larger values (e.g., 4) there is no substantial improvement in the computation time, while recall decreases by more than 0.1.

Finally, it has been measured that with parallel processing computation time diminishes by about 40%: from an average computation time of 183ms to 113ms. Table 5.7 shows the system performance measured on the tuning dataset after having tuned the system parameters and implemented the algorithm improvements.

Testset	Precision	Recall	Computation time
Tuning	1	0.85	113ms
Evaluation	1	0.81	107ms

Table 5.7: Performances of TL-recognizer in the tuning and evaluation datasets.

5.5.4 Results with the evaluation testset

Table 5.7 shows the results obtained with the evaluation dataset. Performance results are very similar to those obtained with the tuning dataset.

While conducting the evaluation, it has been observed that computation time is influenced by the total number of contours that are processed. For example, images with an irregular background (like Figure 5.19) take much longer to compute than average images. For example, Figure 5.20 shows the contours extracted from Figure 5.19: the bright background behind the trees results in more than 8000 contours to be processed. Clearly the great majority is discarded thanks to 'distance' and 'width' constraints, but still 80 of them need to be validated. While the overall result is correct (no traffic light is detected), the computation time for this frame is more than 500ms, about 5 times higher than the average time.

The above observation raises a more general question: how does computation time vary in different illumination conditions? In sunny days it is more likely to have bright surfaces that generate a high number of contours, like in Figure 5.19. Indeed, the average computation time with high light intensity is 196ms. Vice versa, with low light intensity (e.g., at night), since fixed camera parameters are used, the input image is almost entirely black, with the exception of traffic lights and other sources of light, like street lamps and car beacon lights. For example, in Figure 5.21 a single contour is extracted for the green color (there is a small green AOU in

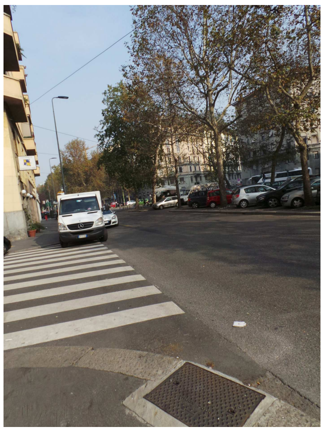


FIGURE 5.19: Frame in a sunny day.

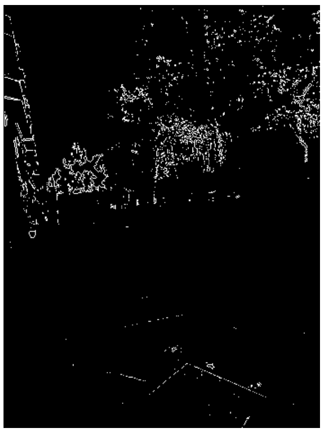


FIGURE 5.20: Contours extracted from Figure 5.19.

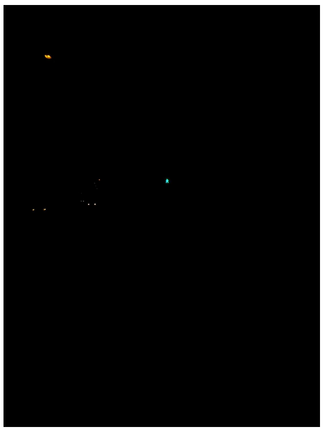


FIGURE 5.21: Frame during night.

the center of the image) and 5 contours are extracted for the 'yellowRed' color (in the figure, in addition to the green AOU, there are 5 small bright dots corresponding to two car beacon lights and a street lamp). Hence, with low light intensity, the computation time is 52ms, on average. In the two intermediate illumination conditions i.e., high and mid light intensities, the average computation times are 124ms and 90ms, respectively.

The proposed technique performs well also in particularly challenging lighting conditions, like some test cases with sun directly behind the traffic light (see Figure 5.22 for an example). Nonetheless, the AOU is correctly identified in 40% of the images, with a mean template matching score of 0.62.

5.5.5 Results of the human-based evaluation

Overall, all subjects have been able to successfully complete the assigned tasks. The only exception was with the first attempt made by the first subject: since he was pointing the camera too high up and almost towards the sky, the traffic light was always out of the camera field of view. The problem was solved by simply explaining to the subject how to correctly point the camera.

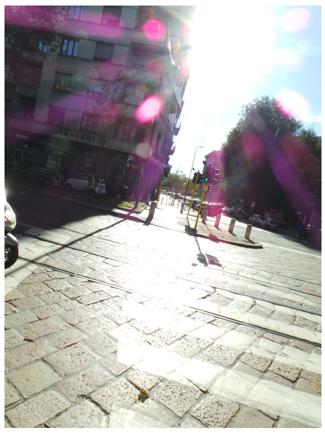


FIGURE 5.22: Example of a test case with sun directly behind the traffic light.

This was explained during the training phase of later experiments involving different subjects. Note that this problem could also be solved by monitoring the pitch angle and by warning the user if the he/she is pointing too high or too low.

During this experiment it has been observed that the two blind subjects needed a slightly longer time (up to about 5 seconds) to find the traffic light. This is due to the fact that they could not precisely predict where the traffic light was and hence needed to rotate left and right until the traffic light entered the camera field of view. On the contrary, the two partially sighted subjects managed to find the traffic light almost instantaneously even if they could not see it. One possible motivation is that the two partially sighted subjects had a better understanding of their current position with respect to the crossroad and a more developed ability to predict the position of the traffic light.

For what concerns the questionnaire, all subjects agree that the application is easy to use and useful. There are some comments that are worth reporting. One subject observes that this application would be very useful because some traffic lights are still not equipped with acoustic signals. Also, even when acoustic signals are available, they may not work properly and sometimes it is difficult to find the button to activate the signal (in Milan acoustic traffic lights need to be activated by a button positioned on the traffic light pole). Another subject observes that he would use this application only when an acoustic traffic light is not available, because it is not convenient to hold the device in one hand while holding the white cane on the other one. All subjects agree on the fact that the vibration pattern is the best way to get the message. Indeed, it could be difficult to hear audio messages because of traffic noise, as observed by one subject. Visual instructions are also not practical, according to both low-visioned subjects, as they are not always clearly visible.

5.6 Summary and future directions

In this chapter we present TL-detector, a system to recognize pedestrian traffic lights aimed at supporting people with visual impairments in road crossings. This work introduces two main contributions to the state of the art. The first contribution is the image acquisition technique. Contributions from previous research [39, 73] acquire images leaving automatic exposure control to the camera software. However, automatic exposure balances the mean luminance of every point in the entire image and may produce underexposed or overexposed images. Instead, the challenge addressed by TL-detector is to capture images with the best possible exposure, regardless of the illumination condition. It delivers on this objective by disabling automatic exposure and varying the mobile camera's ISO value, shutter speed and aperture settings. The second contribution is an analytical computer vision technique to detect the state of pedestrian traffic lights. Two innovative aspects of the technique are that it adopts multi-resolution processing and parallel computation in order to speed up computation.

Experimental evaluation shows that the proposed solution implements a robust method to acquire images with proper exposure and delivers on the objective of guaranteeing robust recognition in different illumination conditions. Indeed, with a precision of 1 and a recall of 0.81, our proposed solution outperforms a previous solution [73] that guarantees a precision of 1 with a recall of about 0.5. TL-detector is also efficient, as it can run several times a second on existing smartphones. Positive results were also obtained with a preliminary evaluation conducted on subjects with VIB: they were able to detect traffic lights in different illumination conditions.

As a future work, user interaction should be carefully studied with the aim of providing all the required information without distracting the user from its surrounding environment. The design of effective user interfaces will become even more challenging if TL-detector is integrated with other solutions that collect and convey to the user contextual information like, for example, ZebraX.

Regarding exposure robustness, improvements could be derived from the adoption of HDR techniques to extend the acquisition dynamic range. In this case tests should be performed to verify the trade-off between reliability gains and computational costs.

An effort will also be devoted to the development of a commercial product based on TL-detector. Indeed, it could be possible to integrate this software with the ZebraX solution presented in Chapter 4 and also in commercial applications like "iMove". This will require to tune the system in order to detect pedestrian traffic lights in countries other than Italy. In order to ease such process, a (semi) automated technique to perform parameter tuning should be investigated.

Chapter 6

Access to images and didactics

In recent years educational applications for touchscreen devices (e.g., tablets) become widespread all over the world. For example, more than $80,000$ educational applications are available for iOS devices[1]. While these devices are accessible to people with VIB, access to educational applications to support learning of STEM subjects is limited because of inaccessible graphics.

In previous chapters, non visual interaction modes based on sonification proved to be effective in augmenting tactile maps and providing guidance in road crossings. In this chapter we build on the experience made using non-visual representation of information while addressing orientation and mobility challenges to approach a different problem: to convey image information to people with VIB. Two main approaches are considered: one based on audio icons and another relying on image sonification. In order to evaluate the applicability of these approaches, we report our experience in the development of three applications for touchscreen devices. Two applications are specifically designed to support people with visual impairments or blindness while studying STEM subjects: *MathMelodies* and *AudioFunctions*. The third application, *Invisible Puzzle*, is designed to evaluate six novel sonification techniques to represent generic binary images.

6.1 Learning Maths with MathMelodies

MathMelodies is an iPad application that supports primary school children in learning Mathematics, designed and implemented to be accessible and enjoyable by both visually impaired and sighted children. The software has been first developed as a university prototype and then, also thanks to a crowdfunding campaign, engineered and distributed as a commercial application[2].

In this section we describe the main design challenges of MathMelodies. We adopted a user-centered design methodology, driven by a number of tests and on-the-field evaluations. In particular, we report the results of three sessions of evaluation: an expert-based evaluation, a test conducted with the first prototype of the app and a qualitative evaluation conducted on the commercial version of the application, involving both sighted and visually impaired children.

[1]Source: http://www.apple.com/education/ipad/apps-books-and-more/
[2] https://itunes.apple.com/us/app/math-melodies/id713705958?mt=8

Finally, we describe the feedback we have received so far and how it impacts the application design.

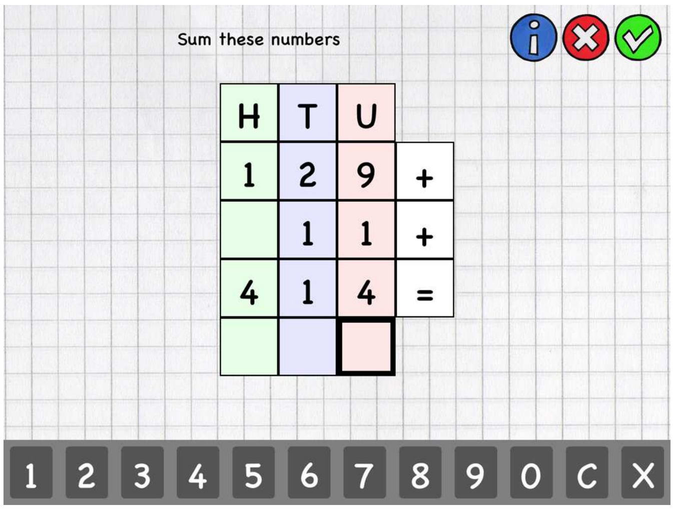
6.1.1 Application design

During the design and development of MathMelodies we faced three main challenges. First, the application must present exercises that are accessible to visually impaired children. In order to determine the best approach to enable access to exercises, we developed a prototype implementing both the "audio icons" and "image sonification" based approaches. With the sonification based approach, the application presents a generic image that can be explored through audio feedback with a solution similar to the one adopted in a previous contribution [94]. For example, the application can guide the student to identify a triangle by reproducing a sound when the boundary is touched. Vice versa, with the audio icons based approach, the application shows some objects on the screen, each one associated with an audio feedback that represents the object itself. In this case, the audio feedback does not depend on the position of the touch within the object. For example, touching the figure of a dog, the application plays a sound of a dog barking. Similarly, a digit is read when it is touched.

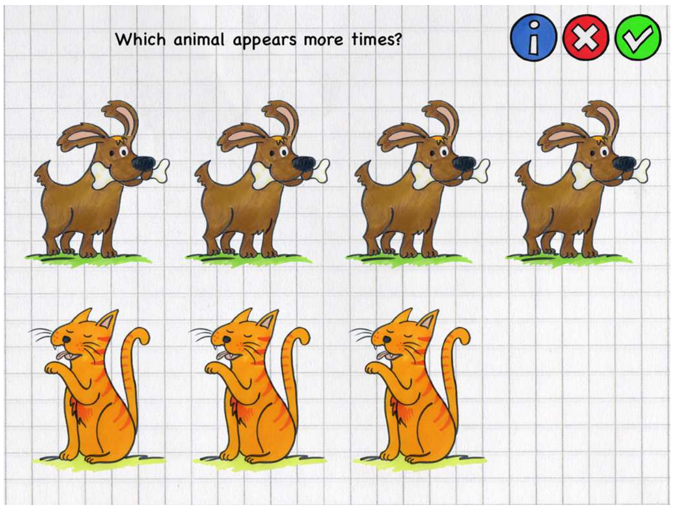
A preliminary evaluation (described in Section 6.1.3) highlighted that the interaction paradigm based on audio icons is less cognitively demanding and more enjoyable. According to these results, we designed a set of 17 different types of exercises relying on the audio icons approach. In order to further simplify the interaction model, we decided to organize the objects into a grid layout that, as we observed in our evaluation, helps reducing the time and mental workload required to explore the entire screen. Another choice driven by the need to simplify interaction consisted in the definition of two input techniques: a simplified on-screen keyboard to insert the digits only (e.g., for the addition exercises, see Figure 6.1(a)) and a multiple choice dialog (e.g., to answer an exercise like the one shown in Figure 6.1(b)).

The second design challenge is to enable reinforcement learning. Children should be stimulated to repeat exercises many times, hence correcting their mistakes and learning by reinforcement. To address this challenge, we designed exercises to have up to 6 difficulty levels. Each exercise is randomly generated, according to its type and the difficulty level set, every time it is presented. For example, in the "easy" addition exercise the child is asked to add two single-digit numbers, while at an harder level (designed for 3rd grade students) the aim is to add three numbers, each one with up to three digits as in Figure 6.1(a). Another important aspect to stimulate children to play the same exercise several times is to entertain them. We pursued this objective by presenting, in most of the exercises, what we call "audio-icons": amusing drawings, each one associated with an easy-to-recognize and funny sound. Also, the application gives a reward to the child, in the form of a short piece of music, when a correct answer is provided. As a future work we also intend to create a more sophisticated reward mechanism that takes into account the number of wrong answers the child provided before giving the right one. For example, zero mistakes can be rewarded with 3 "golden stars".

The third design challenge is to motivate children to keep on doing exercises and practicing. To this purpose, exercises are immersed in a tale, divided into chapters, organized in increasing difficulty levels (two chapters for each grade). Each chapter is further divided into "pages",



(a) Addition exercise with simplified keyboard.



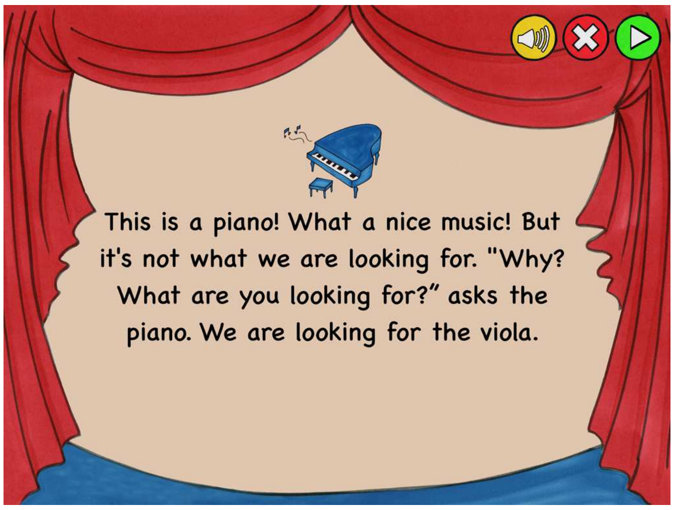
(b) Counting exercise.

FIGURE 6.1: Two exercises of MathMelodies.

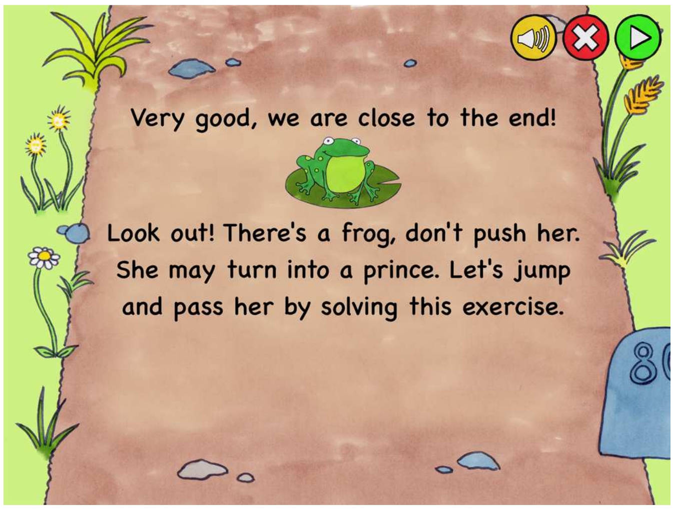
each one comprising a background image, some text (read by a speech synthesizer) and some "audio-icons" often associated with an enjoyable sound (see Figure 6.2). Pages are intertwined with exercises and there are about 30 exercises in each chapter. Every time each exercise is completed correctly, a short piece of melody is played as a reward for the student. At the end of each chapter, all the pieces of the melody are played together to form a full song. Overall, the tale and the audio-icons have the objective of triggering children's interest. This is similar to the approach adopted in most textbooks that heavily rely on colorful images. The difference is, clearly, that in MathMelodies this solution works for visually impaired children too.

6.1.2 Implementation of MathMelodies

The effort to develop MathMelodies can be divided into two main activities: the actual app implementation (i.e., the code writing) and content creation: the story text, images (backgrounds and icons), and audio (sounds and music). Before focusing on the technical issues that arose


This is a piano! What a nice music! But it's not what we are looking for. "Why? What are you looking for?" asks the piano. We are looking for the viola.

(a) A page with a piano audio-icon.


Very good, we are close to the end!

Look out! There's a frog, don't push her. She may turn into a prince. Let's jump and pass her by solving this exercise.

(b) A page with a frog audio-icon.

FIGURE 6.2: Two pages of MathMelodies story.

during the app implementation, we would like to give the idea of the effort that was required to create the content, involving three different professionals (a writer specialized in child tales, an illustrator and a composer) with no prior experience in accessibility. The story is composed by six chapters, each one including about 50 pages similar to the ones depicted in Figure 6.2. Since the app has been localized into two languages (Italian and English), we produced about 600 pages in total. For what concerns the images, there are 25 backgrounds (e.g., the theater curtains in Figure 6.2) and about 100 audio-icons (like the dog in Figure 6.1(b) or the piano in Figure 6.2). Also, each chapter has an associated music that is played at the end of the chapter and that is also divided into small parts, each one played after each exercise for a total of about 200 (short) pieces of music.

While implementing the application we faced a number of technical issues and here we describe two of them. The first issue deals with the large amount of app content. Indeed, it is clearly impractical to define the app by hard-coding the content into the program. Instead, we defined a format for a "content file" that describes, for example, the structure of each chapter, each page, etc. A “content engine” in MathMelodies reads this file and presents the content to the user, in the form of exercises, pages, etc... Thanks to this approach, it is possible to define the app content independently from the app implementation. Also, since we developed a user-friendly tool to edit the “content file”, it is also possible for non-technicians (like the people that collaborated on content creation) to define the app content.

The second issue is related to the implementation of our object-based interaction paradigm that is built on top of the system accessibility tools. On iOS devices, there are two set of tools that make both the OS and user applications accessible to visually impaired users. One set of tools is designed for low-visioned users and includes the “zoom” screen magnifier, font adjustment and color inversion. While some ad-hoc gestures are defined to use the zoom functions, the overall interaction paradigm is analogous to the one for sighted users. The second set of tools is globally called “VoiceOver” and defines a totally different interaction paradigm. The overall idea is that, when the user taps on a graphical object (e.g., a button), VoiceOver gives it the focus and describes it both with a speech synthesizer and an external Braille display (if connected). To activate a focused object (e.g., to press a button), the user double taps anywhere on the screen. In addition to this basic behavior, VoiceOver has several additional gestures to make the interaction more efficient.

In order to enhance app usability for visually impaired users that rely on residual sight, we used large fonts and high contrast between the front objects (i.e., text or pictures) and the background image. Although we did not evaluate this solution with a sufficiently large number of low-visioned users, we expect the app to be accessible to most low-visioned students by using the default accessibility tools.

For what concerns blind users or low-visioned users that cannot totally rely on residual sight, some issues arose in the implementation of the object-based interaction paradigm. Indeed, the simplest solution to implement this paradigm would be to fully rely on VoiceOver (i.e., not implementing any custom behavior for app accessibility). This approach would make it possible to develop an app that is totally consistent with the system-wide interaction paradigm. However, this solution suffers from a major drawback, as it is not suitable to address all design challenges. For example, without defining any custom behavior it is not possible to develop the audio icons that, when VoiceOver is active, reproduce the associated sound upon getting the focus. Other features that call for a custom behavior are multi-tap exploration and automatic reading when a new page is shown.

Clearly, to achieve a deeper customization of the interaction paradigm a larger coding effort is required and it is quite involved to mimic VoiceOver standard behavior as well as to guarantee the consistency with the system-wide interaction paradigm.

6.1.3 Experimental evaluation

During the whole design and development process we took benefit from the feedback obtained from one of the designers who is blind and experienced in education for blind persons. In addition to this constant feedback, we organized three main evaluation sessions.

The first session was organized with four teachers expert in education for blind students[3]. The evaluation was divided in two steps. In the former, we presented a list of exercises derived from Italian educational directives and integrated with workbooks and online resources. We asked experts to evaluate the importance of each exercise in the education of a blind person and to rate how difficult it is to practice it with existing solutions. In the latter, we presented the preliminary prototype implementing sonification-based and object-based interaction paradigms. All four experts independently agreed on the fact that the object-based paradigm would be quicker to learn, more adaptable to a larger variety of exercises and less cognitively demanding for children.

The second session was conducted as a test with three blind children. After a short training with the prototype, we asked each child to solve three exercises adopting object-based interaction and one exercise adopting sonification-based interaction. All students have been able to complete and correctly answer exercise 1 (counting) and 2 (position in a table). Vice versa, one student has not been able to complete (and hence to provide an answer to) exercise 3, a spelling exercise, and exercise 4, which consists in recognizing a triangle using sonification-based interaction. Overall, all students reported that the object-based interaction is easier to understand and two of them also highlighted that it is funnier.

After the second evaluation session a more advanced prototype was developed including both the story and the exercises. This prototype was used in the third evaluation session that was conducted with three blind and two sighted students in primary school. The five students were asked to complete all the exercises in the first chapter consisting in counting exercises, sums, etc. All blind children were enthusiast while using the application. Two out of three reported that they were entertained and engaged especially by the sounds (e.g. the call of animals and the rewarding melodies). All of them experienced some difficulties in the early exploration of tables, and needed help by a sighted supervisor. However, after at most 2 minutes of supervised training, all children got familiar with the application and were able to solve the exercises autonomously providing, most of the times, the correct answer at the first attempt. The two sighted children enjoyed the application as well. One of the two children experienced some difficulties, at the beginning, in understanding how to answer. This was partially due to the fact that the child didn't pay much attention to the exercise explanation. After explaining how to answer, no more help was needed.

6.2 Exploring function graphs with AudioFunctions

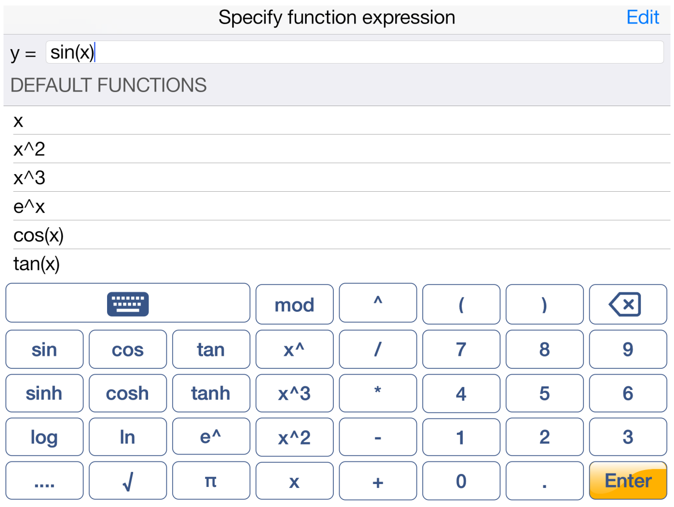
In this section we present AudioFunctions, an iPad prototype that highly increases the independence of math students as well as the comprehension of functions graphs. AudioFunctions presents two major improvements with respect to existing software solutions. First, being a tablet application, it benefits from non-mediated interaction, which is typical of touchscreen devices. This in turns makes it possible to design an interaction paradigm based on the use of proprioception. Indeed, the second improvement consists in a set of three techniques to explore a function graph, two of which highly rely on proprioception.

[3]From the center for the blind people in Brescia, Italy.

In the following, we introduce the AudioFunctions prototype and the design choices made during its development. The use of AudioFunctions can be divided in two main activities: the specification of the function expression, with its drawing properties, and the function graph exploration.

6.2.1 Specification of function expression and drawing properties

To specify a function expression, users can choose a template and then edit it (see Figure 6.3(a)). The template can be chosen from the list of "default" expressions (i.e., a pre-defined set of common functions, like $y = x$, $y = x^2$ or $y = sin(x)$) or from the list of recently used expressions, as they were edited by the user (in Figure 6.3(a) the list of recently used expressions is hidden by the keyboard). To edit the function expression AudioFunctions presents an ad-hoc keyboard, similar to the one of a scientific calculator, containing keys for the digits and for the most common arithmetic and trigonometric operators.


Specify function expression

Edit

y = sin(x)

DEFAULT FUNCTIONS

x

x^2

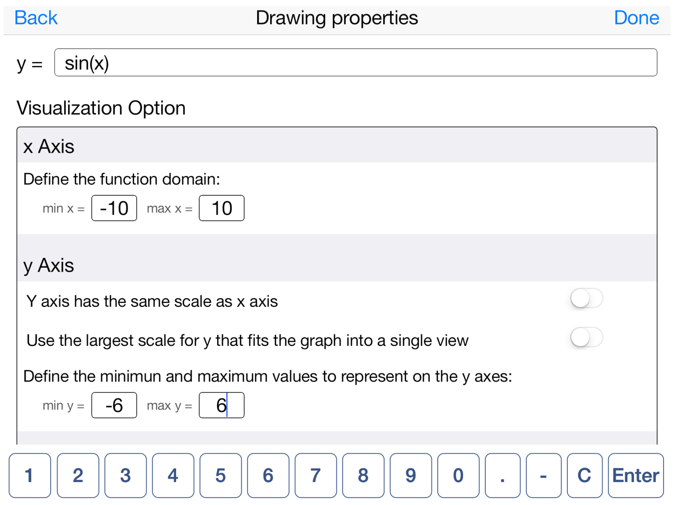
x^3

e^x

cos(x)

tan(x)

(a) Specification of the function expression using our ad-hoc keyboard.


Back

Drawing properties

Done

y = sin(x)

Visualization Option

x Axis

Define the function domain:

min x = -10 max x = 10

y Axis

Y axis has the same scale as x axis

Use the largest scale for y that fits the graph into a single view

Define the minimun and maximum values to represent on the y axes:

min y = -6 max y = 6

(b) Specification of drawing properties.

FIGURE 6.3: Specification of the function expression and its drawing properties.

Function drawing properties (see Figure 6.3(b)) can be specified, including options to define the domain and the scale on the two axes. With the first option, the user can set the function domain in terms of the minimum and maximum values of x to be represented. The second property is a boolean value indicating if the y axis should have the same scale as the x axis. If this property is set to "true" (the default value) then the next two options are disabled. In case this property is set to "false", with the third option the user can choose to automatically scale the y axis, which means that AudioFunctions chooses the largest scale for the y axis that allows the function graph to fits in the screen. If "automatic scale" is disabled, then the user can manually choose the scale for the y axis, indicating the minimum and maximum values to be represented.

6.2.2 Function graph exploration

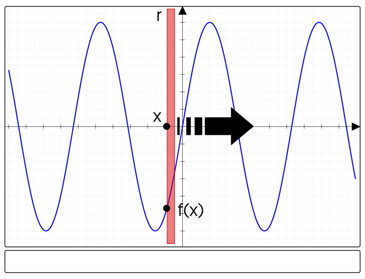
The function graph can be explored using three different "exploration modes" (see Figure 6.4). The first one, that we call "non interactive", is analogous to the solution proposed in Audio Graph Calculator and Sonification Sandbox: by using a "double two finger tap" gesture[4], AudioFunctions starts playing the function sonification, which is obtained as follows. As shown in Figure 6.4(a), AudioFunctions divides the function domain into a set of intuitively small intervals (e.g. the rectangle r). For each interval, given the sonification direction starting from the lowest and up to the highest x coordinate, the app computes the value of $y = f(x)$ where x is the minimum value of the interval and reproduces the "value-sonification" for y, i.e., a sound whose pitch is proportional to the value of y with respect to the range of y values.

Example 6.1. *With the function $y = sin(x)$, for $x \in [-10, 10]$, when $x = \pi/2$ we have $y = 1$ which is also the maximum value for y and hence the value-sonification for $x = \pi/2$ has the highest pitch. Vice versa, if we draw $y = x$, for $x \in [1, 10]$, the sonification for $x = 1$ has the lowest pitch, because 1 is the smallest value represented for y.*

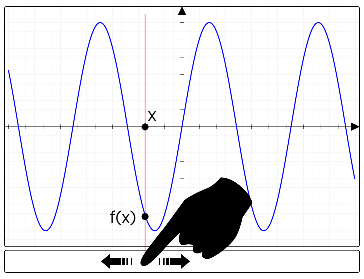
We call the second exploration mode "mono-dimensional interactive" (Figure 6.4(b)). The user can slide the finger along an horizontal bar positioned at the bottom of the view that represents the x axis. While sliding the finger, AudioFunctions uses the value-sonification technique to represent the value $y = f(x)$ where x corresponds to the current finger position. The clear advantage of this exploration mode is that, thanks to proprioception, the user can associate the value of y with the correspondent position on the horizontal axis. Also, the user can move forward and backward along the x axis, at the desired speed, hence, for example, focusing on parts of the function that are more relevant for the user (e.g., a minimum point).

The third exploration mode is called "bi-dimensional interactive". The overall idea is to make it possible for the user to follow with one finger the graphical representation of the graph. The shape of the graph is reconstructed thanks to proprioception. This mode adopts a different sonification, that we call "position-sonification", since the aim is not to encode the y value, but rather to guide the user while following the plotted line. When the user touches on the function line, position-sonification reproduces a sound with the highest pitch. When the user touches outside the line, the pitch diminishes as the minimum distance between the touched position and the line increases. For example, in Figure 6.4(c), point A is more distant from $f(x)$ than

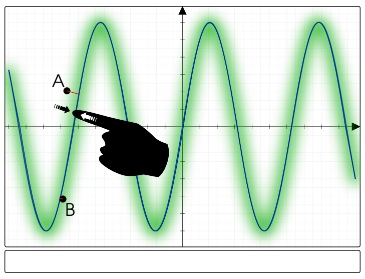
[4]This is the gesture that on iOS devices is associated, for example, to start and pause music reproduction.



(a) Non interactive. AudioFunctions automatically reproduces the value-sonification for all plotted values of $y = f(x)$.



(b) Mono-dimensional interactive. AudioFunctions reproduces the value-sonification for all values of $y = f(x)$ where x corresponds to the current finger position on the horizontal bar.



(c) Bi-dimensional interactive. AudioFunctions reproduces the position-sonification corresponding to the current finger position.

FIGURE 6.4: Function exploration screen and Exploration modes.

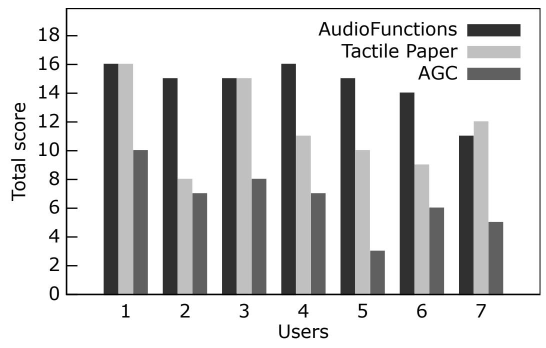
point B. Therefore, when the user touches A a low pitch sound will be played while touching B will yield a high pitch sound.

The two interactive modes have additional features. First, while exploring, AudioFunctions reproduces additional sounds in case the function intersects some “interesting points”, like intersection with the axis, local minimum and maximum and changes in concavity. Second, by double tapping, AudioFunctions reads details on the current position, including: the values of x and $f(x)$ and the function concavity in that point. This is useful because function concavity is easily understandable by sight, but hard to figure out with these sonification techniques. Finally, interaction with two fingers is supported. This is very useful when it is necessary to maintain a reference point during exploration. To achieve this, when a second finger touches the screen, AudioFunctions starts reproducing the sound associated to that finger, ignoring the first one until the second is lifted. For example, consider a case where the user needs to determine the distance between a local minimum and a maximum on the function graph. The user touches the screen and follows the function graph until a local minimum is found. At this point, the first finger is kept in place and a second finger is used to continue exploring the function graph until a local maximum is found. Thanks to proprioception, the distance between the first and second finger can be determined.

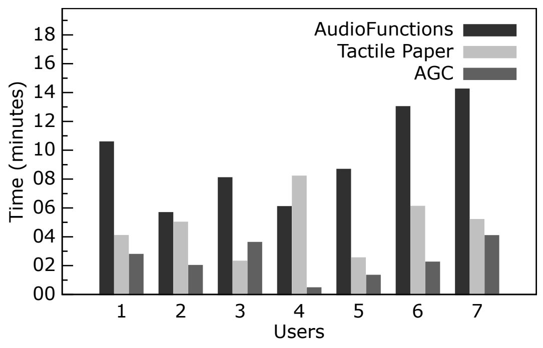
6.2.3 Experimental evaluation

The main objective of AudioFunctions is to let the user perceive the shape of a function graph. Therefore, we focused our experiments on determining how precisely a user can recognize the function properties during exploration. To measure the level of understanding of the function, we asked each user to explore the graph and to answer 8 questions (e.g., "what is the concavity of the function for $x = 0$?"). We scored each answer with a mark of 0 (totally wrong answer or no answer), 1 (partially correct answer) or 2 (correct answer). We run the experiment with 7 blind users, all with some education in Mathematics (at least high school) and acquainted with tactile drawings. During each test session we first described AudioFunctions in about 2 minutes and then we left 3 minutes to let the user get familiar with the app. After this 5 minutes training, we started the test that was divided into three steps, each one involving a different tool: AudioFunctions, tactile drawings and "Audio Graphing Calculator" (AGC), a desktop software to sonify function graphs [27]. The the three steps were presented in random order. During each step we chose a random function expression from a set of pre-defined functions, presenting the corresponding graphs to the subject and posing him/her the 8 questions. While answering each question the user was free to interact with the exploration tool. We recorded the answers and the time needed to provide them.

Figure 6.5(a) shows, for each user and technique, the sum of the scores from all questions (we recall that the maximum is 16). Intuitively, this metric represents the overall understanding of the function obtained by each user with each technique. We can observe that every user obtained much better results by using AudioFunctions with respect to AGC. AudioFunctions proved to be more effective also when compared with tactile drawings that, we recall, all the users were acquainted with. Indeed every user, except user 7, obtained better results with AudioFunctions than with tactile drawings (e.g., users 2, 4, 5 and 6).



(a) Function comprehension.



(b) Total time to answer.

FIGURE 6.5: Results of the experimental testing.

Figure 6.5(b) compares the total time required by each user to answer the 8 questions by using each technique. Results show that, by using AGC, users provided answers more quickly (about 2 minutes on average) than with tactile drawings (about 5 minutes on average) and AudioFunctions (about 9 minutes on average).

6.3 Exploring binary images with *Invisible Puzzle*

Starting from the experience made with the sonification techniques designed for AudioFunctions, we decided to design novel sonification techniques to represent binary (i.e. black and white) images, such as function drawings, diagrams and charts. In this section we present six new "sonification modes" i.e., software modules that combine an image exploration paradigm with a sonification technique. The sonification techniques adopt elements derived from previous proposals together with novel solutions, such as sound spatialization (employing both interaural level and time differences) and sound equalization filtering and are designed to be effective after little training.

Beyond designing an effective sonification technique, there is another major challenge: to tune its parameters and evaluate it with a large number of users. We present a methodology to compare sonification modes designed to deliver on two main objectives: first, to enable the quantitative comparison of the sonification modes (e.g., in terms of the average number of correct answers); second, to enable the development of an automated evaluation tool that makes it possible to conduct a large number of tests with limited supervision effort. The methodology is implemented in Invisible Puzzle, an iOS application that automates the process of training subjects, administering tests and collecting usage data. We present the results of the off-line statistical analysis of the collected data, highlighting performance differences between the sonification modes, also considering the demographics of test subjects.

We first introduce the six novel sonification modes, and how we adapted two solutions from existing literature. All sonification modes are based on a parameter mapping approach [34] where the sonified parameter is the luminance of a specific area of the image. Each sonification mode is characterized by three main components:

- *Exploration paradigm* - defines how the image can be explored, and which portion is considered for sonification given the position of the finger on the screen. Three exploration paradigms are defined, two based on a *probing* approach, one based on a *scanning* approach.
- *Audio rendering technique* - transforms the selected image portion into higher level information.
- *Sound generator* - generates the sound signals.

6.3.1 Exploration paradigms

In this contribution we describe a novel sonification mode based on the *bi-dimensional* (2-D) exploration paradigm. A larger number of novel sonification modes (5) are developed for the *uni-dimensional* (1-D) exploration paradigm.

2-D exploration

We designed the 2-D exploration paradigm to provide a benchmark paradigm likely to be intuitive for the user, as it mimics how a person with visual impairments or blindness explores drawings on swell paper. A similar solution is defined as "bi-dimensional interactive" by Taibbi et al. [81] and "local area sonification" by Yoshida et al. [94]. 2-D exploration allows the user to touch the screen and to sonify the specific pixel touched in that moment[5]. The rendered sound therefore depends on both the horizontal and the vertical position of the finger on the screen, hence the name *bi-dimensional (2-D) exploration*.

The audio rendering stage takes in input the luminance of the pixel being touched and re-scale it in a range of sound frequencies between 100 Hz for the lowest luminance value (i.e., black color) and 1000 Hz for the highest (i.e., white color). The resulting frequency value is then used by the sound generator, which produces a sinusoidal wave with that frequency and fixed amplitude.

1-D exploration

The 2-D exploration paradigm requires users to explore the image along two dimensions to perceive all image features. Our intuition, also confirmed by experimental results (see Section 6.3.7), is that 2D exploration is time consuming.

The 1-D exploration paradigm was designed to address this issue by allowing users to explore the image by touching the screen and moving their finger up and down on a single dimension. The sonified portion of the image does not correspond only to the touched pixel (as in 2-D exploration), but to the whole horizontal line (*flush line*) at the same height of the touch point. The horizontal (left-right) position of the finger on the screen is not relevant for this particular exploration paradigm.

The 1-D exploration paradigm is similar to the technique proposed by Dallas [22], as it simultaneously represents all image features on the flush line. However, there are two main differences: first, our solution is interactive as it adopts a *probing* approach, while the one by Dallas adopts a *scanning* approach. Second, while the solution proposed by Dallas sonifies image features along a vertical line, our approach sonifies a horizontal line. This choice was driven by the fact that we use audio spatialization (based on both interaural time and level differences) to convey additional information about the explored images. Indeed, it has been shown in the literature that it is more natural for the user to associate left-right spatialized audio information to graphical features on a horizontal line [89, 7].

In general terms, the luminance of pixels on the *flush line* is rendered generating a low-frequency sound for pixels located on the left part of the screen, gradually changing to high frequency for pixels located on the right part of the screen. Furthermore, sounds generated from pixels on the left part of the screen are spatialized on the left, gradually changing to the center and the right for pixels on other parts of the *flush line*. A schematic representation of the 1-D interaction and sonification modes can be found in Figure 6.6.

It is important to underline that all the sounds corresponding to a single *flush line* are reproduced at the same time, not sequentially moving on the line from left to right. This is due to the fact that we are designing a sonification mode with a probing approach, i.e., a real time interactive system. This requires the audio feedback to instantaneously describe the flush line as a whole.

[5]Clearly, the fingertip touches more than a pixel, however we rely on the mobile OS function that identifies a single pixel that intuitively represents the center of the touch.

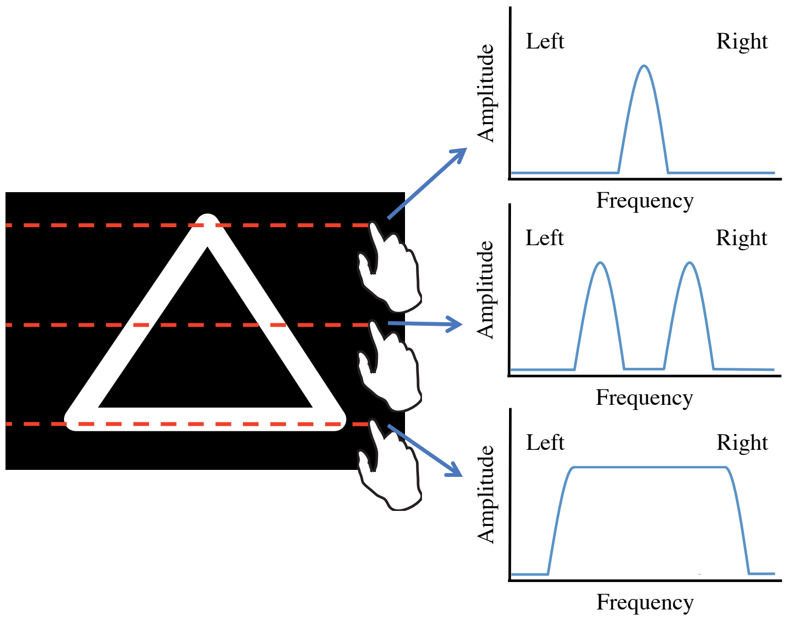


Figure 6.6: Schematic overview of the 1-D interaction and sonification modes.

For example, if the user slides the finger on the screen (e.g., from the top to the bottom), in each instant he/she will hear the sonification of the current flush line. This would not be possible if the flush line was described with a time-varying sound (e.g., a sound obtained by scanning the flush line from left to right in one second).

In order to allow for a clear discrimination between concurrent low and high frequency sounds, the spatialization was implemented using both Interaural Level Differences (ILD) and Interaural Time Differences (ITD). The ILD range was set to a maximum of 20 dB for left-right position, linearly scaled down to 0 dB for the center position. Similarly, the ITD range was set to a maximum of 1 ms. These values are consistent with spatial hearing literature [61].

The high-low frequency and left-right spatialization mapping were developed to be as intuitive as possible, taking inspiration from the keyboard of a piano, with the low frequency notes on the left and the high frequency notes on the right. The frequency ranges utilized for the sonification are consistent with the equal loudness curve, therefore with the frequency range for which the human hearing has enhanced loudness sensitivity [61].

Two different audio rendering techniques have been developed using this exploration paradigm, namely *Variable Frequency* (VF) and *Fixed Frequency* (FF).

6.3.2 Audio rendering techniques

1-D Variable Frequency

The 1-D Variable Frequency audio rendering technique has been designed in order to represent image features on the flush line at the highest possible resolution (i.e. there is a continuous mapping between the x coordinate of each pixel on the flush line and frequency and spatialization parameters of the generated sound). A luminance threshold is established. Each pixel on the *flush* *line* with luminance above this threshold is sonified with a sound generator, whose frequency is associated with the horizontal position of the correspondent pixel. The frequency range is scaled between 100 Hz for the first pixel on the left, and 1618 Hz for the last pixel on the right. Furthermore, low frequency sounds are spatialized on the left, gradually moving towards the right for high frequency sounds, as previously described.

With this particular audio rendering technique, a horizontal white line corresponds to a number of sounds equal to the line length in pixels (possibly a few hundreds). This could potentially create issues in terms of saturation of the output audio channel and, more importantly, it creates a redundancy of information: the ability of the human hearing system to discriminate between several sounds at different frequencies is in fact ultimately limited.

To address this problem, a minimum distance is established between sonified pixels. This is achieved as follows. Starting from the left side of the screen, when a pixel is found with luminance above the threshold, a few following pixels are not sonified regardless of their luminance. The number of these pixels is determined in order to allow for a maximum of 24 concurrent sounds. This value was established considering the maximum sensitivity of the human hearing system in terms of frequency bands detection (the Bark scale [95]).

Two sound generators are usually employed in literature for image sonification: pure tone and noise. We implemented both of them.

- **1-D VF Pure**. The sound generators produce pure sinusoidal sounds.
- **1-D VF Noise**. The sound generators produce narrow-band noise (1/3 octave band width).

1-D Fixed Frequency

Due to the particular features of the audio rendering process (i.e., variable frequency and fixed amplitude), both 1-D VF sonification modes allow for smooth frequency changes when exploring an image. However, several sounds with very similar frequencies might be present at the same time and at the same amplitude, creating comb filters and phasing, which are perceived as a marked *vibrato* effect. Our intuition is that this *vibrato* effect could be unpleasant for users.

To deal with this problem, we designed the 1-D Fixed Frequency audio rendering technique that generates sounds at 24 predefined frequencies. The *flush line* is divided into 24 equally-sized sectors. The average luminance of each sector is directly sonified modifying the amplitude of 24 sound generators, each reproducing continuously a signal at a fixed frequency, from 100 Hz for the generator correspondent with the sector at the extreme left of the screen, to 1440 Hz for the generator correspondent with the band at the extreme right. The number of sectors is consistent with the number of concurrent sound generators adopted in 1-D Variable Frequency.

For example, the amplitude of a generator correspondent with a sector in which there are only black pixels is 0, gradually scaled up to 1 (maximum) for a sector in which there are only white pixels. The sound of each generator is spatialized from the left to the right, considering the horizontal position of the associated sector.

In 1-D FF we use the two sound generators already defined for 1-D VF. However, considering the level of annoyance generated by listening for long period of times to pure tones and random-generated noise, we decided to also introduce a third solution which employs a music signal as sound generator. This option is expected to be more enjoyable.

- **1-D FF Pure**. The sound generators produce pure sinusoidal sounds.
- **1-D FF Noise**. The sound generators produce narrow-band noise (1/3 octave band width).
- **1-D FF Music**. Instead of 24 sound generators, one for each sector, a single sound generator consisting of a music track player is used. The sound is split into 24 bands, each with center frequencies going from 100 Hz to 1440 Hz in 1/3 octave bands. The average luminance of each sector on the *flush line* is associated with the amplitude of the corresponding band (left to right, low to high frequency, left to right spatialization). As an example, if the luminance of a sector on the right of the *flush line* is high, and for all other sectors is low, then only some high frequency components of the actual music will be audible, spatialized on the right. The track chosen for the playback is an 8-seconds extract from a pop song, continuously looped as soon as the user keeps the finger on the screen. Thanks to the presence of a drum-kit and of several tuned instruments (no voice), the frequency spectrum is rather broad (40 Hz to 20 kHz).

Due to the particular features of the audio rendering process (i.e., fixed frequency and variable amplitude), the 1-D FF Noise and 1-D FF Pure sonification modes generates stepped frequency variations without the *vibrato* effect perceivable in the VF modes. Furthermore, the variable amplitude, associated with the pixel luminance, allows for a higher compatibility with grayscale images if compared with the threshold rendering technique adopted for the VF modes. We leave the evaluation of this sonification mode with grayscale images as a future work.

6.3.3 Adaptation of existing solutions

In order to compare the sonification techniques described in the former sections with previous solutions, we implemented two sonification modes derived from existing literature [22, 94]. In the following, we report details on how they have been adapted and implemented.

1-D Non Interactive Pure

The first technique, derived from a previous contribution [22], is very similar to the 1-D Fixed Frequency Pure technique described in Section 6.3.2. It adopts an exploration paradigm that differs from our contribution on two key aspects. First, it uses a scanning approach instead of a probing approach, i.e. it uses a non-interactive exploration paradigm where the user cannot decide what portion of the image should be sonified. Second, it represents image features on a vertical flush line rather than on a horizontal one.

This was implemented in a new sonification mode, called “1-D NI Pure” (NI stands for “Non Interactive”). Considering the exploration paradigm, in 1-D NI Pure the sonification starts when the user touches the screen, and does not depend on the point being touched.

The image is horizontally divided in 24 equally sized columns. Upon touch, image features inside each column are automatically sonified in sequence, from left to right, in a total time of 4 seconds. This implies that each column is sonified for 1/6 second before automatically moving to the next one. The audio rendering technique adopted is similar to the one used for 1-D Fixed Frequency Pure (see Section 6.3.2). Each column is subdivided, along the vertical axis, in 24 sectors of equal size, each associated to a sound generator. The average luminance of each sector is sonified modifying the amplitude of the corresponding sound generator. Each generator reproduces a pure tone signal at a fixed frequency, ranging from 100 Hz for the sector on the bottom to 1440 Hz for the sector on the top of the image. Our implementation of this technique adopts the same pure tone sound generators already defined for our novel techniques.

2-D Pulse

The second technique, derived from a previous contribution by Yoshida et al. [94], adopts the same exploration paradigm introduced with our 2-D technique, i.e. it sonifies image features of the point being touched by the user. However, the sound rendering technique differs because it conveys two different parameters: the luminance of the point being touched and the distance between the touched point and the closest edge in the image. Originally (i.e. in Yoshida et al. [94]), this technique implemented two separate sonification modes: “Local area sonification”, which uses a scanning approach like the one adopted in 1-D NI Pure, and “Distance-to-edge sonification”, which uses a probing approach like our 2-D technique. Considering the complexity of such approach, and the fact that one of the goals of Invisible Puzzle is to allow individuals to start playing the game without any particular training, we have decided to implement only the Distance-to-edge mode.

Within “2-D pulse” (the name given to this implementation) luminance is represented in the same way as in our 2-D technique. Distance to the closest edge is pre-computed for each pixel using the Felzenszwalb algorithm [26] and stored in a look up table. When the user touches a pixel in the image, the corresponding value is retrieved from the look up table. In order to convey distance information to the user, the audio rendering technique maps the distance value to the period of a pulse train sound. The pulse train period for a point which is at the maximum distance from an edge is 1 second, linearly decreasing as the touch gets closer to the edge.

6.3.4 Implementation details

The sonification modes adopted in Invisible Puzzle have been implemented as an iOS library on top of an existing open-source framework called “The Amazing Audio Engine” (TAAE)[6]. By adopting this framework, it is possible to implement sound generation and filtering components in a timelier manner, without having to deal with the low level programming aspects of the iOS audio system. In TAAE, sound is produced using audio processing components of three different types. The first type, called “Channel”, is used to generate sound, either programmatically or by reading audio data from a file. The second type, “Filter”, receives sound from another component (e.g., a channel) and uses digital signal processing techniques to alter it. A third type of audio processing component, called “Channel group”, is used to mix together the output

[6]http://theamazingaudioengine.com/

Type	Name	Description	Input
Channel	Pure	Generates a pure sinusoidal sound	Sine wave frequency f
	Noise	Generates narrow-band noise	-
	Music	Generates a music signal by reading audio samples from a file	-
Filter	ITD	Applies Interaural Time Differences to the output of another element	Spatialization value $s \in [-1, 1]$ where -1 corresponds to full left spatialization and 1 to full right
	ILD	Applies Interaural Level Differences to the output of another element	
	Band Pass	Applies a band pass filter to the output of another element	Band pass frequency f
	Volume	Modifies the amplitude of the output of another element	Multiplier a
Channel Group	FF	Implements the fixed frequency audio rendering technique	Array of luminance values
	VF	Implements the variable frequency audio rendering technique	Array of points

TABLE 6.1: The nine personalized audio processing components.

of (possibly many) other components (e.g., two channels). The software library that implements the sonification modes is available on request[7].

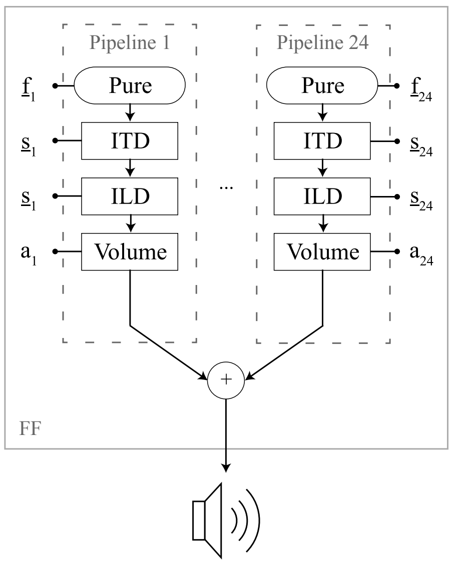
We extended the basic TAAE components to define nine personalized *audio processing components* (see Table 6.1). Each sonification mode is then implemented as a combination of some of these components into an *audio processing pipeline*. Sound generators (as defined above) correspond to channels, in the TAAE terminology. Similarly, audio rendering techniques are implemented through pipelines.

For example, the 2-D sonification mode is implemented with a pipeline containing only a Pure channel where the sine wave frequency varies accordingly to the luminance of the point being touched.

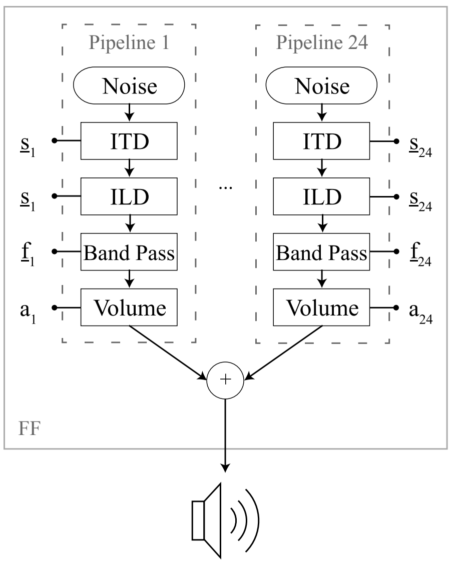
A more involved pipeline is required to implement 1-D FF Pure (see Figure 6.7(a)). Each of the 24 sectors of the flush line (see Section 6.3.2) is associated with a pipeline composed of 4 elements: a Pure channel, an ITD filter, an ILD filter and a Volume filter. The output of the 24 pipelines is mixed together by a FF channel group. When each pipeline is initialized, predefined values for sine wave frequency and spatialization are set. At runtime, the FF channel group receives an array of 24 values, each one representing the average pixel luminance in each sector, and dynamically modifies the volume multiplier of each pipeline accordingly.

A similar pipeline is used to implement 1-D FF Noise (see Figure 6.7(b)). There are two main differences. First, the Noise channel is used instead of Pure. Second, there is an additional Band Pass filter in each pipeline. At initialization time, the band pass frequency is set with a predefined value (which is the same frequency used as input for the Pure channel in 1-D FF Pure). 1-D FF Music is the same as 1-D FF Noise with the only difference that it uses the Music channel instead of the Noise one.

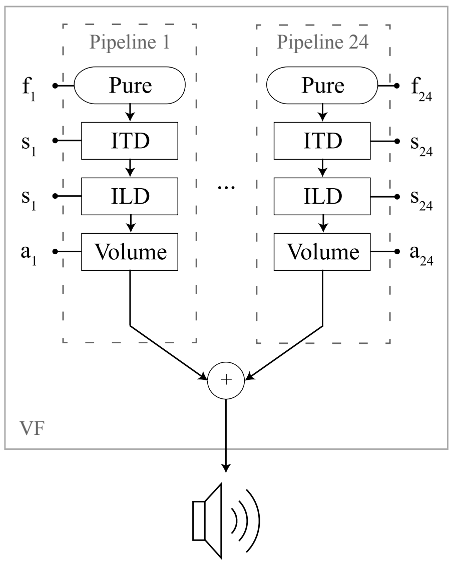
[7] https://ewserver.di.unimi.it/16taccessip/



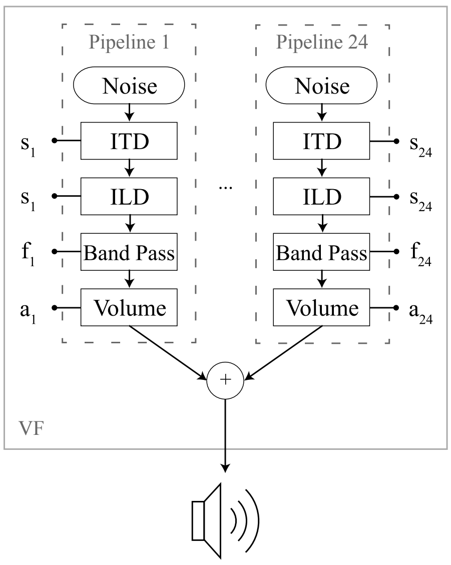
(a) 1-D FF Pure.



(b) 1-D FF Noise.



(c) 1-D VF Pure.



(d) 1-D VF Noise.

FIGURE 6.7: The audio processing pipelines implementing some of the novel sonification modes. Underlined letters ($\underline{s}$, $\underline{f}$) represent constants, while the others (a, s, f) denote parameters defined dynamically at runtime.

1-D VF Pure is similar to 1-D FF Pure, as it uses the same 24 pipelines (see Figure 6.7(c)) but differs in two key aspects. First, sine wave frequency and spatialization values are not defined at initialization time, but they are dynamically set as described in the following. Second, the input provided to the VF channel group at runtime is the array of points on the flush line. Each one of these points is dynamically associated to a pipeline whose sine wave frequency and spatialization values are proportional to the point's horizontal position. Finally, 1-D VF Noise (see Figure 6.7(d)) is similar to 1-D VF Pure with the difference of adopting the Noise channel instead of Pure and the presence of an additional band pass filter whose value is dynamically set by the VF channel group.

6.3.5 Evaluation methodology

The overall goal of the experimental evaluation is to understand how effectively each sonification mode allows subjects to perceive images. To deliver on this goal we follow this procedure: first, a sonification mode is introduced to the user, providing the necessary training. Then, the subject is asked to perform a series of tasks, where he/she must recognize images relying only on sound. During each task, we measure how fast and precisely the subject recognizes each image. In the end, a questionnaire is administered to collect some information about the subject, such as age, familiarity with video games, etc.

It is important to underline that each subject conducts the evaluation using a single sonification mode. There are two motivations for this choice. First, from the methodological point of view we want to exclude that results collected for one task are biased by the fact that the subject has previously experienced a different sonification mode. Second, from the subject's point of view, it is simpler to learn how to explore shapes with a single sonification mode.

Tasks

In each task the test subject is challenged to recognize a shape by using the sonification mode. Clearly, the shape to be recognized is not visible and it can only be perceived by sound. The subject can freely explore the shape without time constraints.

When the subject believes he/she has recognized the shape, he/she moves to the "answer phase". In this phase the subject cannot explore the shape any more. Instead, he/she is presented with four shapes, one of them being the ones that has just been explored. The subject has to recognize the shape he/she has just explored (i.e., correct answer) hence distinguishing it from the other three (i.e., wrong answers). In case of subjects with blindness, a textual description is read.

All images used in the tasks (including the image to recognize and the three wrong answers for each tasks) are reported as online resources[7].

While a subject faces a task, we collect performance and usage information. As motivated in Section 6.3.5, a subject can face the same task in more than one *attempt*. Hence, we collect data for each attempt in each task. Note that, after the first attempt the user knows which is the correct answer and hence we are mainly interested in analyzing the results for the first attempt.

For each attempt in each task we collect:

- Time required to complete the attempt (i.e. from the first touch on the screen to the moment in which the response is given);
- Exploration time (i.e. time in which the finger was touching the screen);
- Number of touches on the screen (only during exploration);
- User response (including whether it is correct or not) for each attempt.

Training

Considering subjects' training, the goal is to introduce the subject to the challenge (i.e. recognize hidden images) and to the basics of the interaction paradigm. This is achieved by playing a very short video (15 seconds). Three of these have been created, one for each exploration paradigm (the videos are available online[7]).

After playing the video, a "learning task" is presented in which the shape is visible (a textual description of the shape is also provided for subjects with blindness). Subjects can freely explore the image, becoming acquainted with the sonification mode.

After these two training steps a large portion of subjects (up to 50% for some sonification modes) is still unable to recognize a simple shape (i.e., a dot on the center right of the screen). In order to offer further training, we adopt two solutions. First, when the wrong shape is selected during a task, the correct answer is shown (or read) to the subject and then he/she is asked to repeat the same task. With this approach the subject can precisely associate the audio feedback with the shape that, in this case, is known.

The second solution to offer further training consists in adopting a gamification approach [23] so that exploration tasks are designed to gradually increase in difficulty. Indeed, we defined six groups of tasks, each one consisting of four tasks. The first three groups focus each on a particular type of shape: dots, line segments and polygons, respectively. The fourth group confronts the user with shapes from all previous groups, while the fifth and sixth are focused on more involved shapes. The fifth group comprises objects of different nature: French playing cards suits, vehicles, animals and fruits (examples in Figure 6.8). The sixth contains letters, both uppercase and lowercase, and numbers. An additional "learning task" (i.e., a task in which the image is visible) is presented at the beginning of each of the first three groups in order to provide training. Note that the first four groups have also been used in the evaluation reported in our previous work [29]. Consequently, the experimental evaluation presented in this section (which extends the one presented in our previous work) reports results both for tests conducted with the original four task groups and with the so called "long test" comprising also the two additional groups.

The increasing difficulty of tasks implies that new elements can be gradually introduced. For example, the first two tasks in the first group consist in a single dot, positioned in different locations on the screen. The third and fourth tasks are more complex, as they contain two dots, which in one case are located on the same *flush line*. Thanks to this approach it is possible to avoid that the user faces tasks that are too challenging (and frustrating) too early in the test. At the same time, by finely tuning the increasing difficulty of the tasks, it is possible to engage

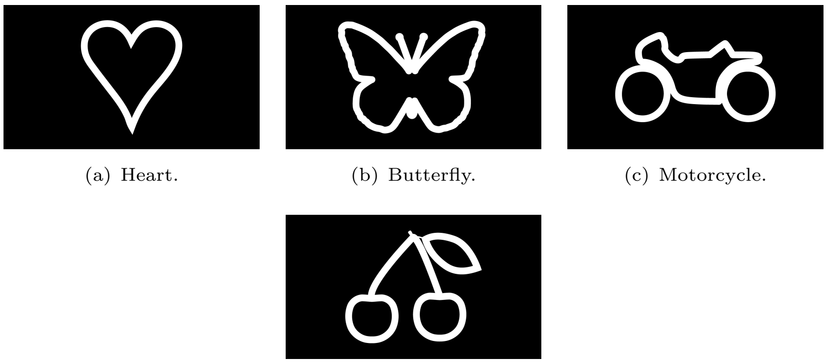

(a) Heart. (b) Butterfly. (c) Motorcycle. (d) Cherry.

FIGURE 6.8: Examples of objects included in the fifth task group.

Subjective evaluation (five-level Likert scale)	I enjoyed playing with Invisible Puzzle I would like to play with Invisible Puzzle again I have found some levels to be too easy I have found some levels to be too difficult I easily understood how to use Invisible Puzzle I easily understood how images are represented through sound I think that the hints provided by Invisible Puzzle were enough to learn how to use it Playing with Invisible Puzzle required me a lot of concentration I have found the sounds comforting/pleasant
Questionnaire - subject's data	Age Play with computer games (Less than 1 hour per week/1 hour per week/1 hour per day/More than 1 hour per day) I play a musical instrument or sing (yes/no)
Questionnaire - Comments	Free text

TABLE 6.2: Questionnnaire details.

the user by presenting challenges that are not too easy. This is in line with the "Flow theory" from Csikszentmihalyi [21].

Questionnaire

After completing all tasks, subjects are asked compile a questionnaire that has three main objectives: first, to obtain a subjective evaluation of the sonification mode, second, to collect personal information that can be associated with performance data and, third, to let the subject give a comment about the test. Table 6.2 reports the details of the questionnaire. Note that, since the evaluation procedure was implemented by our Invisible Puzzle application (described in Section 6.3.6), some of the questionnaire questions refers to Invisible Puzzle.

6.3.6 Invisible Puzzle

The minimum piece of software that enables the evaluation described in Section 6.3.5 consists in an application that implements the sonification modes. However, such a software would require a person that supervises the entire evaluation process, negatively affecting evaluation scalability.

In this section we present Invisible Puzzle, an iOS application that implements the sonification modes and also automates the evaluation, in the sense that no supervisor intervention is required except for finding test subjects and providing the smartphone running Invisible Puzzle. Indeed, Invisible Puzzle trains each subject to use the sonification mode, challenges him/her to complete the tasks, collects performance data and administers the questionnaire. While a subject is performing a test, the supervisor may observe the interaction and collect eventual thinking aloud comments.

The design of Invisible Puzzle

Invisible Puzzle was designed employing a user centered approach through a series of iterations. After each iteration, Invisible Puzzle has been tested with end users in supervised evaluations and, indeed, the whole process benefited from feedback given by many subjects, including four blind individuals (one of which is a member of the research group). Nevertheless, feedback from sighted subjects was also considered as the design of the initial training stage resulted to be even more challenging with these subjects.

From the interaction design point of view, the most challenging part was to introduce the user to two basic activities: to explore the screen and to terminate the exploration (with a double tap).

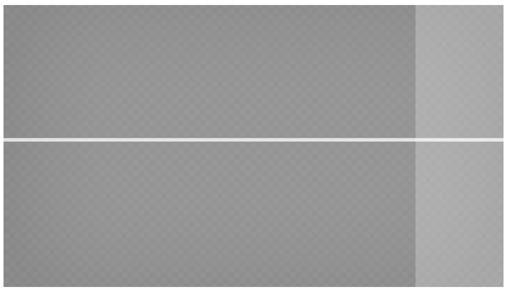
In its first version, Invisible Puzzle presented a textual explanation of these activities. However, we observed that most users did not understand the principle of the image exploration.

To address this problem we added, in the second version, a "learning task" presented after the textual description. However, we observed that many sighted users, in particular with the 2-D sonification mode, tended to tap on the visible dots rather than to slide the finger on the screen. As a consequence, in the following task with a hidden shape, users did not know how to interact.

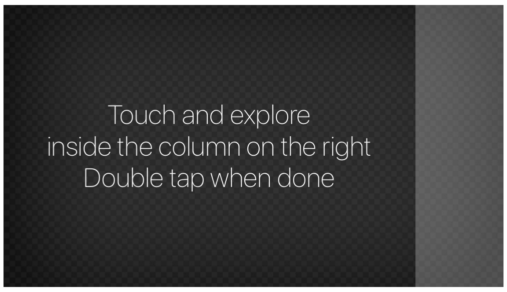
This problem was addressed by substituting the initial textual explanation with a video that included a speech-based description. This description is more detailed than the purely textual one, and includes the explanation of the challenge (i.e., to recognize hidden shapes). With this solution, most users were able to use the 2-D sonification mode, but some of them still had problems with the sonification modes based on the 1-D exploration paradigm. Users did not seem to understand that the image was sonified in its full width independently of the horizontal coordinate of the point being touched.

To address this problem, we changed Invisible Puzzle so that, when using the 1-D exploration paradigm, the user is constrained to slide the finger over a small column on the right edge of the screen (all touches outside this column have no effect). Furthermore, the *flush line* was visually represented on the screen (see Figure 6.9(a)).

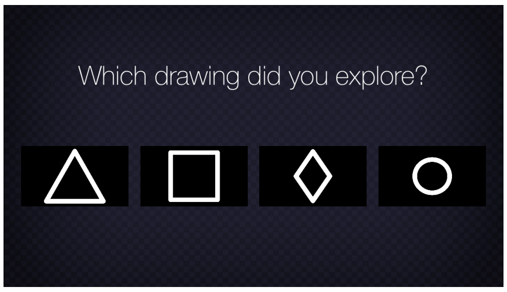
Finally, some subjects did not understand how to terminate a task and hence remained stuck on the first one. This issue was solved by displaying a text message (text-to-speech for blind subjects) reminding the subjects about how the task could be completed (double-tap on the



(a) Exploration view when the user is touching (1-D exploration paradigm).



(b) Exploration view when the user is not touching (1-D exploration paradigm).



(c) Answer view.

FIGURE 6.9: Interaction with Invisible Puzzle.

screen) every time that the subject did not touch the screen for a given amount of time (2 seconds). See Figure 6.9(b).

Thanks to this design process we developed the introductory part of Invisible Puzzle that allowed all subjects involved in the experiments to complete the tasks without the need of an external explanation.

To summarize, the interaction for completing a task works as follows: in the *exploration view* the subject explores a (hidden) shape through sonification (Figures 6.9(a) and 6.9(b)). When the subject believes to have recognized the shape, he/she double-taps on the screen. Invisible Puzzle then presents the *answer view* (see Figure 6.9(c)) that contains the explored shape together with other three ones, in random order. The user has to identify the explored shape by touching it. For users with blindness, the four possible choices are described with a text-to-speech synthesizer.

During informal tests conducted as part of the user-centered design, we observed a problem with the automated evaluation. Subjects were less motivated to concentrate on the tasks and to complete the evaluation procedure. This poses an additional challenge: the evaluation procedure should engage the subjects, encouraging them to devote an effort in completing the assigned task. This requires the procedure to be carefully tuned in order to avoid being boring (e.g. tasks should not be too easy) or frustrating (e.g. tasks should not be too difficult). The choice of tasks is the result of this tuning.

Remote data acquisition

Invisible Puzzle performs an instrumented remote evaluation [33] to collect: a) quantitative data during app usage; b) information about the test subject through a questionnaire;

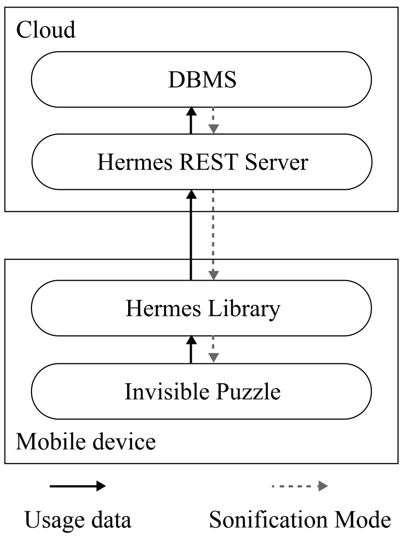


FIGURE 6.10: The "Hermes" remote logging system architecture.

The logging system is composed of three main elements, as shown in Fig 6.10: a data collection library (which is part of Invisible Puzzle), a REST (Representational State Transfer) server and a Database Management System (DBMS).

The data collection library has three main purposes. First, to query the server for the sonification mode that must be adopted in each test session. This preliminary operation is required in order to guarantee that all sonification modes are evenly evaluated even if test sessions are conducted in parallel with no supervisor intervention. The second purpose is to collect usage data and questionnaire results. Finally, the third purpose is to send the collected data to the REST server in a reliable way (i.e. performing local caching in order to be resilient to networking issues).

6.3.7 Evaluation results

In order to get statistically significant results, we involved a large number of subjects in the evaluation of Invisible Puzzle (49 persons with visual impairments and 178 sighted). Table 6.3 reports the number of subjects with visual impairments who took part to the evaluation of each sonification mode, together with the aggregated personal data we collected from the questionnaire, while Table 6.4 reports the same metrics for sighted subjects.

Indeed, as we discuss in Section 6.3.8, the comparison between the results of the two groups can help identifying similarities and differences in the cognitive process behind image exploration [44]. Additionally, results from the two groups can help obtaining statistically significant data to guide future development and experimental studies for both visually impaired and sighted subjects. Also, while this contribution is motivated by the needs of people with visual impairments, it cannot be excluded that, in some particular contexts, sighted users can also benefit from sonification techniques. These include situations in which the user cannot look at the device's screen, for example when he/she is operating particular machines or tools that require full visual concentration. The need for sonification techniques might also arise in situations in which

Sonification	Subjects	Age		Playing music		Playing videogames			
		m	σ	yes	no	$> 1h/day$	$1h/day$	$1h/week$	$> 1h/week$
2-D	6	27.33	7.15	4	2	0	0	0	6
1-D VF Pure	6	35.33	5.85	6	0	0	0	2	4
1-D VF Noise	7	32.29	6.58	7	0	0	0	1	6
1-D FF Pure	6	29.17	8.35	5	1	0	0	0	6
1-D FF Noise	6	27.83	7.31	5	1	0	0	3	3
1-D FF Music	6	30.67	5.32	5	1	0	0	0	6
1-D NI Pure	6	36.33	11.25	3	3	0	0	1	5
2-D Pulse	6	29.17	5.60	6	0	0	0	3	3
Total	49	31.04	7.51	41	8	0	0	10	39

Table 6.3: Count and personal data of subjects with blindness collected from the questionnaire (“m” = mean).

Sonification	Subjects	Age		Playing music		Playing videogames			
		m	σ	yes	no	$> 1h/day$	$1h/day$	$1h/week$	$> 1h/week$
2-D	24	25.46	6.21	8	16	3	2	7	12
1-D VF Pure	22	27.41	9.48	11	11	7	6	5	4
1-D VF Noise	21	24.90	10.09	9	12	6	9	3	3
1-D FF Pure	23	26.43	4.55	15	8	4	4	2	13
1-D FF Noise	21	24.62	6.21	13	8	10	5	4	2
1-D FF Music	25	30.16	11.61	9	16	3	8	3	11
1-D NI Pure	21	34.19	4.19	9	12	3	3	5	10
2-D Pulse	21	24.00	4.32	5	16	4	6	6	5
Total	178	26.00	7.78	79	99	40	43	35	60

Table 6.4: Count and personal data of sighted subjects collected from the questionnaire (“m” = mean).

the user is interacting with a device that has no screen at all, like some wearable devices that are becoming popular nowadays (e.g., wristbands).

Experimental setting

All tests were conducted on iPhone 5 and iPhone 5*S* devices (that have the same screen size) and during the tests subjects wore Apple *EarPod* headphones. Tests with sighted users were conducted both in Italy and in the UK, while tests with blind users were conducted in Italy only[8]. Tests were conducted in various environments: in most of the cases the authors' office, while in few cases the students' library, laboratories and also the cafeteria. In all cases, tests have been conducted in environments with limited ambient noise.

Tests were not supervised, in the sense that subjects were asked to conduct a test and no other information was provided, apart for the expected duration of the test (i.e., approximately 15 minutes). The authors served as supervisors and invited students, friends and colleagues to use the application. The device was provided by the supervisor, who also was in charge of starting Invisible Puzzle and (de)activating VoiceOver. During the evaluation the supervisor was in the same room as the subject, but was not observing the test. or example, a common situation is a test taking place in the supervisor's office; while the supervisor works at his computer, the subject tries to complete the different tasks within Invisible Puzzle.

[8]Invisible Puzzle is localized in Italian and English.

Sonification	Num. of Subjects		Correct answers		Exploration time (s)		Task time (s)	
	tot	long test	%	σ	m	σ	m	σ
2-D	6	0	77.0	7.5	25.8	18.4	38.6	32.3
1-D VF Pure	6	6	67.7	12.1	28.0	20.6	39.2	33.6
1-D VF Noise	7	6	65.1	21.9	29.6	27.6	40.6	35.7
1-D FF Pure	6	0	82.2	22.8	15.2	11.5	21.6	18.3
1-D FF Noise	6	6	67.7	6.1	29.5	22.1	36.3	23.0
1-D FF Music	6	0	62.5	13.1	18.0	12.9	24.5	17.4
1-D NI Pure	6	6	56.2	15.8	DNA	DNA	36.5	46.2
2-D pulse	6	6	75.0	14.7	37.5	43.6	47.0	48.0
Total	49	30	69.1	16.4	26.3	25.5	35.6	34.6

TABLE 6.5: Results with blind subjects ("m" = mean). All results computed for the first 16 tasks.

Some of the tests (i.e. those used in our earlier publication [29]) consisted in 4 groups of 4 tasks, for a total of 16 tasks. 2 additional groups (i.e. 8 additional tasks) were added to the remaining tests (i.e., 30 tests with visually impaired subjects and 42 with sighted subjects), for a total of 6 groups (24 tasks).

Results with blind subjects

Table 6.5 reports mean values (m) and standard deviations (σ) for the parameters measured during the use of Invisible Puzzle, for each sonification mode. Results refer to the initial 16 tasks by the 49 subjects with visual impairments.

Considering that in this particular case the performance of the subjects was characterized by both speed (i.e. exploration time) and accuracy (i.e. percentage of correct answers), an initial analysis was performed in order to look for potential interactions between these two parameters. A Pearson product-moment correlation was run to determine the relationship between exploration time and percentage of correct answers. There was a medium, positive correlation between these, which was statistically significant ($r = 0.33, n = 49, p = 0.021$). A Multivariate Analysis of Variance (MANOVA) was then performed. Using Pillai's trace, we found that the sonification had a significant effect on both the exploration time and percentage of correct responses [$V = 0.66, F(14, 82) = 2.36, p = 0.008$].

Univariate analysis was then performed, starting with the percentage of correct answers, which is displayed in the box plot in Figure 6.11. This value represents the percentage of tasks in which subjects gave the correct answer in the first attempt. 1-D FF Pure yields better results, on average, if compared with the other sonification modes. Considering that the dataset is normally distributed, a one-way ANOVA analysis was conducted, showing that the differences are not statistically significant [$F(4, 251) = 1.63, MSE = 783, p = 0.167$]. The results observed for the percentage of correct answers have been correlated with other parameters (e.g., whether the subject plays a musical instrument), but no statistically significant difference emerged.

1-D FF Pure yields better results also considering exploration time (as shown[9] in Figure 6.12). Also in this case, considering that the data set is normally distributed, a one-way ANOVA

[9]The exploration time metric does not apply to 1-D NI Pure.

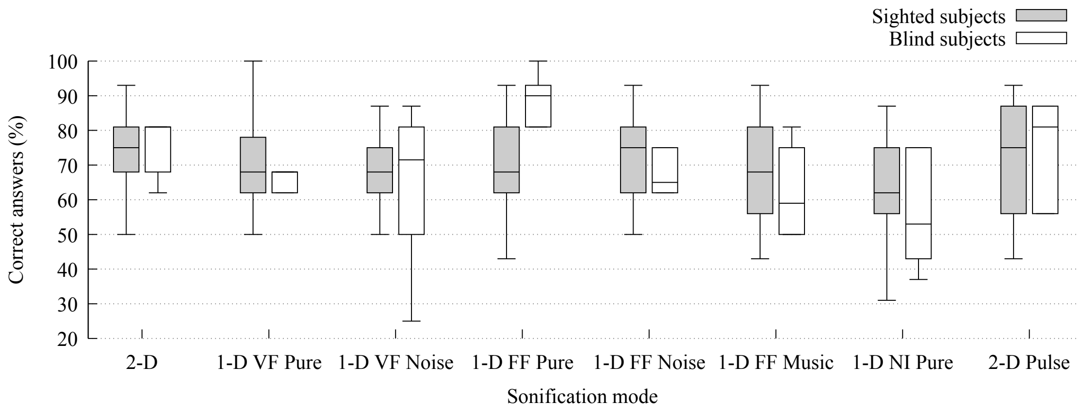


FIGURE 6.11: Boxplot representing the correct answers for each sonification mode.

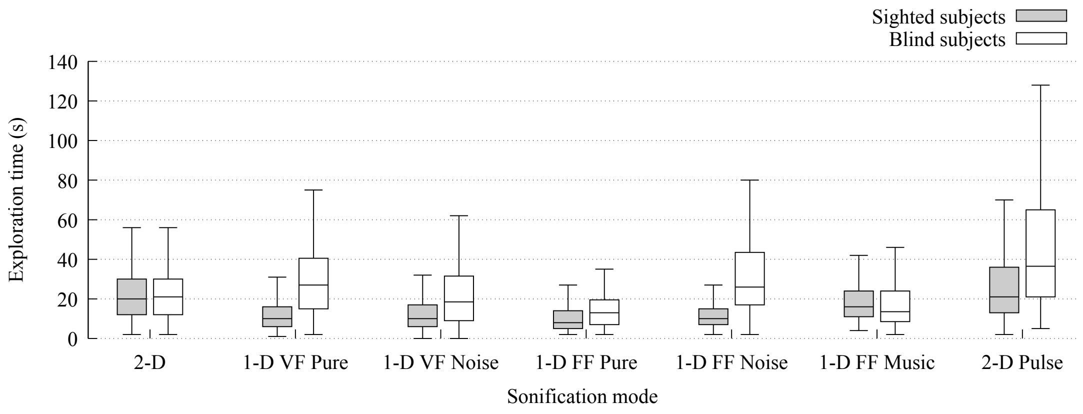


FIGURE 6.12: Boxplot representing the exploration times for each sonification mode.

analysis was used. The differences are statistically significant [$F(7, 776) = 12.50$, $MSE = 543.72$, $p < 0.001$]. Tukey post-hoc analysis shows that 1-D FF Pure is significantly better than 2-D ($p = 0.035$), 1-D VF Pure ($p = 0.004$), 1-D VF Noise ($p < 0.001$) and 1-D FF Noise ($p = 0.001$). It is important to underline that inferential analysis was carried out considering the time taken to complete every first attempt, and not the average time for each subject.

It is possible to note that the average task time is 12.5 seconds larger than exploration time. This is due to the fact that task time includes the exploration time plus the time spent selecting the answer. For users who are blind, this requires listening to the description of each of the four shapes, and selecting the chosen one.

Considering the results of the questionnaire (see Table 6.6), it emerges that all sonification modes require high concentration (there is no statistical significant difference among the different sonification modes). Furthermore, subjects enjoyed Invisible Puzzle more with 1-D FF Pure, which is in line with the fact that this sonification mode allows subjects to quickly complete the tasks with a high percentage of correct answers. The general satisfaction in using this sonification mode is reflected in the intention to play again with it, and in the evaluation of how pleasant is its sound. Indeed, subjects found the sound of 1-D FF Pure as pleasant as the sound of 1-D FF Music, which was actually designed with the specific aim of producing a pleasant sound.

As reported in Section 6.3.5, tasks have increasing difficulty. It is therefore interesting to consider the different results (in terms of percentage of correct answers) in the different chapters. This

Sonification	Concentration		Enjoy		Play again		Pleasant sound	
	m	σ	m	σ	m	σ	m	σ
2-D	4.6	0.5	3.6	1.2	3.8	0.9	3.6	1.0
1-D VF Pure	4.8	0.4	3.5	1.2	3.1	1.1	2.8	1.1
1-D VF Noise	4.7	0.4	3.2	1.1	3.1	0.8	3.5	0.5
1-D FF Pure	4.3	0.5	4.5	0.8	4.3	1.0	4.5	0.5
1-D FF Noise	4.1	0.7	3.8	0.4	3.5	0.8	3.8	0.7
1-D FF Music	4.0	0.8	3.5	0.8	3.3	0.8	4.5	0.8
1-D NI Pure	4.6	0.5	3.0	0.8	3.0	0.6	3.0	1.4
2-D pulse	4.8	0.4	3.6	0.5	3.6	0.5	2.5	0.5
Total	4.5	0.6	3.6	0.9	3.4	0.9	3.5	1.0

TABLE 6.6: Results of the questionnaire for blind individuals ("m" = mean).

Sonification	Group 1		Group 2		Group 3		Group 4		Group 5		Group 6	
	%	σ	%	σ	%	σ	%	σ	%	σ	%	σ
2-D	83.3	12.9	79.1	24.5	91.6	12.9	54.1	24.5	NA	NA	NA	NA
1-D VF Pure	66.6	20.4	70.8	18.8	75.0	22.3	58.3	25.8	29.1	29.2	70.8	29.2
1-D VF Noise	57.1	18.8	71.4	36.5	67.8	40.0	64.2	19.6	25.0	31.6	50.0	27.3
1-D FF Pure	91.6	20.4	79.1	29.2	83.3	30.2	75.0	22.3	NA	NA	NA	NA
1-D FF Noise	75.0	22.3	75.0	22.3	58.3	25.8	62.5	37.9	29.1	24.5	66.6	12.9
1-D FF Music	58.3	25.8	70.8	29.2	66.6	12.9	54.1	36.7	NA	NA	NA	NA
1-D NI pure	45.8	18.8	58.3	30.2	62.5	20.9	58.3	25.8	16.6	20.4	29.1	24.5
2-D pulse	83.3	12.9	79.1	18.8	75.0	38.7	62.5	26.2	41.6	30.2	50.0	31.6
Total	69.9	23.3	72.9	25.9	72.4	27.5	61.2	26.5	28.3	26.8	53.3	28.4

TABLE 6.7: Percentage of correct answers for each group of tasks (Blind individuals).

is reported in Table 6.7. While inferential analysis does not show any statistically significant difference, we can observe that 1-D FF Pure has good performance in all groups (i.e. it is the sonification mode with best performance in groups 1, 2 and 4, while in group 3 it is the second after 2-D). In contrast, 1-D NI Pure is the sonification mode with the worse percentage of correct answers (it has the lowest value in all groups but group 4). Considering the last two groups (4 and 5, whose results are available for 5 sonification modes only), we can observe that in group 5 all sonification modes (except 2-D pulse) do not allow the subjects to distinguish the figures; indeed, the average number of correct results is about the same as the baseline value of 25% (we recall that the correct answer should be chosen in a set of four possible candidates). Instead, in group 6 all sonifications except 1-D NI Pure have much better results, above the baseline. This suggests that, in this case, we probably failed to tune the difficulty of the tasks in the fifth group.

Results with sighted subjects

Table 6.8 reports the results of the test conducted with 178 sighted subjects.

Similarly to the analysis conducted in Section 6.3.7, a Pearson product-moment correlation was run to determine the relationship between exploration time and percentage of correct answers. There was a small, positive correlation between these, which was statistically significant ($r = .29, n = 178, p < 0.001$). A Multivariate Analysis of Variance (MANOVA) was then performed. Using Pillai's trace, we found that the sonification had a significant effect on both the exploration time and percentage of correct responses [$V = 0.66, F(14, 340) = 11.98, p < 0.001$].

Sonification	Num. of Subjects		Correct answers		Exploration time (s)		Task time (s)	
	tot	long test	%	σ	m	σ	m	σ
2-D	24	0	75.0	13.0	22.1	14.1	24.6	14.8
1-D VF Pure	22	0	69.6	15.7	11.6	7.3	13.4	8.0
1-D VF Noise	21	0	67.8	11.9	11.6	7.9	12.9	7.9
1-D FF Pure	23	0	72.0	11.1	10.5	8.2	11.8	8.6
1-D FF Noise	21	0	73.5	12.9	12.5	9.0	13.4	9.2
1-D FF Music	25	0	68.0	15.7	21.5	14.7	23.0	15.5
1-D NI Pure	21	21	61.9	15.4	DNA	DNA	11.2	6.8
2-D Pulse	21	21	72.0	15.2	26.2	21.9	29.2	23.2
Total	178	42	70.0	14.2	16.6	11.8	17.6	14.4

TABLE 6.8: Results with sighted subjects ("m" = mean). All results computed for the first 16 tasks.

Univariate analysis was then performed. Considering the percentage of correct answers (see box plot in Figure 6.11), the 2-D sonification mode is slightly better than the 1-D (e.g., mean 75% with 2-D and 73.5% with 1-D FF Noise). The sonification that exhibits worse results according to this metric is 1-D NI Pure, with a value of 61.9%. Considering that the data sets are normally distributed, a one-way ANOVA analysis was conducted. The results show that there are no statistically significant differences between the eight sonification modes [$F(7, 170) = 1.90, MSE = 0.02, p = 0.072$].

Interestingly there are statistically significant differences [$F(1, 176) = 16.92, MSE = 0.02, p < 0.001$] between subjects that do not play musical instruments (66% correct, $\sigma = 14.3$) and those who do (74% correct, $\sigma = 12.7$). On the other hand, no statistically significant difference can be found between individuals who play computer games (one hour per week or more, 69% correct, $\sigma = 14.3$) and individuals who do not (less than one hour per week, 65% correct, $\sigma = 15.6$).

The performances with the various sonification modes exhibit clearer differences in terms of exploration time, as shown in Figure 6.12. 2-D, 2-D pulse and 1-D FF Music require a longer exploration time if compared with the other four sonification modes. One-way ANOVA analysis reveals that the differences between the seven sonification modes are statistically significant [$F(7, 2840) = 115.0, MSE = 148.37, p < 0.001$]. As expected, post-hoc Tukey analysis highlights that 1-D VF Pure, 1-D VF Noise, 1-D FF Pure and 1-D FF Noise are not significantly different among themselves, but each of them is significantly different (i.e. it has a shorter exploration time) with respect to the other sonification modes i.e., 2-D, 1-D FF Music and 2-D Pulse ($p < 0.001$ for each pair). 2-D is not significantly different from 1-D FF Music ($p = 0.995$), but both sonification modes are significantly different (shorter exploration time) from 2-D pulse ($p < 0.001$ in both cases).

Considering again the exploration time, statistically significant differences [$F(1, 2846) = 7.19$, $MSE = 189.55, p = 0.007$] are found between subjects that do not play musical instruments (mean exploration time 16.1s per task, $\sigma = 13.7$) and those who do (mean exploration time 14.7s per task, $\sigma = 13.8$).

Task time is approximately 2.8 seconds higher than exploration time for all sonification modes. The statistical differences between sonification modes are similar to those emerging for the exploration time metric. One-way Anova shows statistically significant differences [$F(7, 2846) =$

Sonification	Concentration		Enjoy		Play again		Pleasant sound	
	m	σ	m	σ	m	σ	m	σ
2-D	3.8	0.9	3.4	0.7	3.1	1.0	2.2	1.0
1-D VF Pure	3.8	1.0	3.9	0.7	3.5	0.9	2.8	1.1
1-D VF Noise	3.9	0.9	3.9	0.8	3.4	1.1	3.0	0.9
1-D FF Pure	3.5	0.9	4.4	0.6	4.1	0.5	3.6	1.0
1-D FF Noise	4.4	0.8	3.8	1.1	3.9	1.1	3.2	1.0
1-D FF Music	4.4	0.7	4.0	0.6	3.3	1.0	4.2	0.8
1-D NI Pure	4.0	0.9	3.3	0.9	3.0	1.2	2.9	1.2
2-D pulse	4.3	0.7	3.3	0.7	2.7	1.0	2.0	1.0
Total	4.0	0.9	3.7	0.8	3.4	1.1	3.0	1.2

TABLE 6.9: Results of the questionnaire for sighted individuals ("m" = mean).

$103.24, MSE = 167.96, p < 0.001]$ between the groups. Post-hoc Tukey analysis highlights that 1-D VF Pure, 1-D VF Noise, 1-D FF Pure, 1-D FF Noise and 1-D NI Pure are not significantly different among themselves but each of them is significantly better (i.e. allows smaller task time) than the other three sonification modes (2-D, 1-D FF music and 2-D Pulse). Furthemore, 2-D and 1-D FF music are significantly better than 2-D Pulse.

Considering the qualitative data collected through the questionnaire (see Table 6.9), other interesting aspects emerge. First, higher concentration is required for some of the sonification modes (1-D FF Noise, 1-D FF Music and 2-D Pulse), but no statistically significant difference emerges. Instead, there are statistically significant differences for the level of enjoyment. 1-D FF Pure is significantly enjoyed more than 2-D, 1-D NI Pure and 2-D Pulse ($p < 0.001$). The same results hold for reported intention to play again with Invisible Puzzle. 1-D FF Pure is significantly better than 2-D, 1-D NI Pure and 2-D Pulse ($p = 0.023$, $p = 0.010$ and $p < 0.001$, respectively). Finally, 1-D FF Music is the sonification mode with the most appreciated sound, and the difference is significant with respect to all other sonification modes ($0.000 < p < 0.025$)) except 1-D FF Pure.

Table 6.10 reports the percentage of correct answers in the different groups of tasks. No statistically significant differences emerge among the sonification modes in the four groups of tasks except for 1-D NI Pure, which shows significantly worse results than the others sonification modes in the first group $[F(7, 170) = 5.40, MSE = 0.04, p < 0.001]$. It appears that this sonification mode is less intuitive than the others, and that some additional training in the first group of tasks is necessary. This is confirmed by the fact that for task groups 2, 3 and 4 no significant differences emerge if compared with the other sonification modes.

From Table 6.10 we can also observe that the average percentage of correct answers is much lower in the fourth group of tasks. This was expected; as discussed in Section 6.3.5, the tasks in the fourth group are more challenging than those in the previous ones.

6.3.8 Discussion

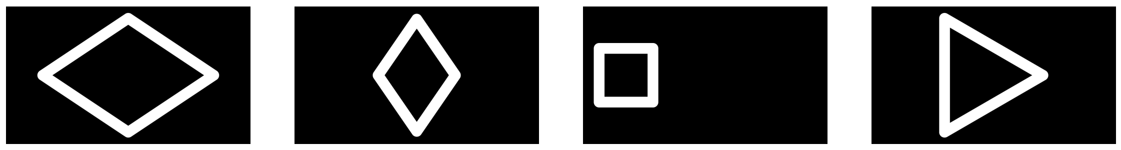
This contribution focuses on two main aspects: the effectiveness of the sonification modes and the scalability of the evaluation. Looking at the subjects' performances[10] using the various proposed

[10]In this section, unless explicitly stated, we refer to the result obtained with subjects with blindness.

Sonification	Group 1		Group 2		Group 3		Group 4	
	%	σ	%	σ	%	σ	%	σ
2-D	89.5	14.5	88.5	16.4	81.2	21.1	40.6	27.3
1-D VF Pure	80.6	21.7	81.8	24.6	77.2	21.6	38.6	26.4
1-D VF Noise	88.1	16.9	77.3	19.2	66.6	18.2	39.2	25.7
1-D FF Pure	86.9	18.2	71.7	21.7	83.7	17.8	45.6	20.8
1-D FF Noise	86.9	18.7	85.7	12.6	72.6	23.5	48.8	26.7
1-D FF Music	80.0	20.4	81.0	23.1	69.0	30.0	42.0	23.6
1-D NI pure	58.3	24.1	71.4	21.3	76.1	27.9	41.6	22.8
2-D pulse	82.1	23.9	84.5	23.0	75.0	26.2	46.4	22.7
Total	81.7	21.6	80.3	21.0	75.2	23.9	42.8	24.3

TABLE 6.10: Percentage of correct answers for each group of tasks (sighted individuals).

sonification modes, we can conclude that, despite the very short training, users can successfully recognize geometric shapes after a few seconds of exploration. Considering for example the third group of tasks, subjects are asked to recognize geometric figures of different nature (e.g. squares, circles, diamonds) and size. We expected these tasks to be challenging for subjects adopting the 1-D exploration paradigm, because they must deal with sounds generated by two or more points on the same *flush line*. For example, considering the fourth task in the third group (see Figure 6.13), despite the similarity between the correct answer (a wide diamond) and one of the possible alternatives (a narrow diamond), the percentage of correct answer is 76% among all subjects who used one of the 1-D sonification modes. Results are even better for subjects adopting the 1-D FF Pure sonification mode, considering that all the individuals managed to correctly identify the right answer. Using the five novel 1-D sonification modes, subjects gave a correct answer in 70% of the cases for all tasks of group three. If we consider the results of 1-D FF Pure it can be observed that 4 subjects (out of 6) correctly recognized all the shapes, with a mean exploration time of 18, 5 seconds. It is important to remember that these subjects only practiced with the technique during task groups one and two. On average, before starting group 3, each of these users practiced for less than 3 minutes (i.e., 176s).



(a) Wide diamond. Correct. (b) Narrow diamond. Wrong. (c) Square on the left. Wrong. (d) Triangle with one base vertically aligned on the left. Wrong.

FIGURE 6.13: Answers of the fourth task in the third group.

Looking at the scalability of the evaluation, a specific challenge has been to quickly introduce the user to the exploration paradigm. Thanks to the user-centric approach described in Section 6.3.6, all subjects successfully completed the evaluation procedure without requiring external intervention. This does not mean that the introductory explanation (i.e. the video) provides an exhaustive explanation on its own; it is the combination of the explanation with the following interactive trial that allowed subjects to get proficient with the sonification modes. To support this conclusion, it can be considered that with the 1-D sonification modes only 44% of the subjects with visual impairment provided a correct answer in the first task. This means that, before starting the first task, more than half of the blind subjects did not actually understand how the sonification mode worked. However, more than 78% of subjects that gave a wrong answer in the first task gave a correct answer in the second one which is set, approximately, at the same difficulty level. This suggests that errors committed in the first task helped subjects to understand how the sonification mode worked.

A final challenge faced when designing the evaluation procedure was to engage subjects, so that they could complete the procedure without distractions. Results show that Invisible Puzzle achieves this objective. In particular, there are some sonification modes that are more suited to subject engagement. We originally expected 1-D FF Music to be more engaging. Instead, it resulted that listening to music required higher attention, and the sonification mode that resulted more enjoyable by both sighted subjects and subjects with visual impairment was 1-D FF Pure.

Comparing the performance of sighted and visually impaired subjects (see Tables 6.8 and 6.5), we can observe that, on average, sighted subjects had a faster exploration time. This is statistically significant for all sonification modes (e.g. for 1-D FF Pure [$F(1, 462) = 20.81$, $MSE = 81.27$, $p < 0.001$]), except 1-D FF Music. Interestingly, for 1-D FF Music blind subjects have a significantly lower exploration time [$F(1, 494) = 4.64$, $MSE = 207.25$, $p = 0.032$]. No significant differences could be observed regarding the percentage of correct answers among all tasks. However, considering the percentage of correct answers separately for each group of tasks (see Tables 6.7 and 6.10) two different trends can be observed. In groups 1 and 2 sighted subjects have a significantly higher level of correct answers ([$F(1, 225) = 11.09$, $MSE = 0.05$, $p = 0.001$] for group 1 and $F(1, 225) = 4.24$, $MSE = 0.05$, $p = 0.041$] for group 2). In the third group, sighted subjects are slightly better, on average, but no statistically significant difference can be observed. Finally, in the fourth group blind subjects have statistically significant better results ([$F(1, 225) = 21.02$, $MSE = 0.06$, $p < 0.001$]). This suggests that subjects with visual impairment require a longer initial training, but ultimately reach higher performance levels. This can partially be motivated by the fact that part of the training is still based on visual clues (e.g. the initial video). After a few minutes of training, individuals with visual impairment become significantly more effective in recognizing the proposed shapes, possibly due to the fact that they explore them more carefully (i.e. they devote a longer time to explore).

Other observations can be made regarding the qualitative remarks reported by the subjects. First, several subjects, especially those with visual impairment, underlined (either by telling to the operator or by writing it in the comment section) that they had fun playing with Invisible Puzzle (e.g. "Very entertaining and challenging!"[11], "Nice"). At the same time, some subjects remarked that the early stage of the game was not trivial, as it required them to figure out by themselves how the sonification technique worked. Furthermore, most of the subjects that faced the long version of the test (i.e., with 24 tasks in total) reported that the game was too challenging. In particular, they reported that it was very difficult to recognize objects (group 5). For example: "The level with objects is too tough. I couldn't distinguish the car from the airplane".

Also, some subjects reported to be annoyed by the pure-tone sound and by the high pitch (e.g. "The sounds were a bit annoying", "Sound is annoying, in particular in the challenging levels").

Some subjects noted that, in the most challenging tasks, they did not have a clear understanding of the hidden shape, but they were able to correctly answer thanks to the fact that Invisible

[11]Comments in Italian have been translated.

Puzzle presented a multiple choice, which effectively restricted the range of possible shapes. Furthermore, almost all of the visually impaired subjects reported that it was easier to recognize a shape when the context was known (e.g. in group 6 subjects knew that they were exploring letters and digits). This might also explain the fact that the results obtained in group 6 are much better than those from group 5. In group 5 subjects knew that they had to recognize "objects", which is a very generic term, while in group 6 they knew they had to recognize letters and digits, which dramatically reduced the range of possible answers.

Some users reported problems in distinguishing circles from polygons. Others noted that, more in general, figures with curved lines (e.g., letters, the car, etc.) were harder to recognize. An additional problem was also found with individuals with visual impairment; in some cases, they did not know the shape associated to the description of an object (e.g., lowercase letters).

A final remark concerns the interactivity of the application. Some subjects with visual impairment that used 1-D NI Pure expressed their concerns for the lack of interactivity of this sonification mode.

While the evaluation approach based on Invisible Puzzle eases the process of collecting evaluation data, it still has two limitations: first, it requires a supervisor to invite the subject, that hence need to be personally in touch with the supervisor. Second, the test is conducted on a device provided by supervisor, requiring the supervisor and the test subject to be in the same physical place. To further scale up the number of tests, it is necessary to allow subjects to conduct the evaluation on their own device. In this case there would be no supervisor at all.

We are actually working in this direction: the experience derived from the design of Invisible Puzzle and the results about the sonification modes allowed our team to develop a more advanced version of Invisible Puzzle, that is publicly available on the AppleStore[12] in the form of a game. While users play with Invisible Puzzle, we remotely collect usage data. In particular we expect that, by advertising Invisible Puzzle in communities of people with visual impairments, it will be possible reach a broad sample of the population, remotely collecting usage data from subjects with different visual impairments. Also, if Invisible Puzzle matches its intended aim of being entertaining, users will likely play with it in more than one session, hence making it possible to conduct a longitudinal study.

6.4 Summary and future directions

In this chapter we present three solutions where different non-visual interaction modalities are adopted to convey image information.

The first solution is MathMelodies, a tablet application to support math learning of primary school children. One of the primary contributions of this work is the identification of a non-visual interaction paradigm adopting audio-icons to enable access to math exercises limiting the cognitive effort required by children. Also, we report the challenges encountered while developing the application and the many design decisions taken. Among these decisions, the adoption of a tale as a narrative background that entertain children between exercises and the dynamic

[12] https://itunes.apple.com/us/app/the-invisible-puzzle/id1051337548

generation of exercises with increasing difficulties, all with the goal of engaging children in order to keep them practicing maths. To the best of our knowledge, MathMelodies is the first app for math learning on mobile devices that is specifically designed to be accessible to visually impaired children. Also, as the results of the experimental evaluations show, MathMelodies is actually accessible and entertaining. MathMelodies has been available on the AppStore[13] for about two years. So far, MathMelodies has been incrementally improved according to the continuous feedback provided by many users all over the world. In particular, many suggestions of improvement have been received from educators for blind and sight impaired students and were implemented in recent releases.

We extended MathMelodies to automatically collect usage data and, as a future work, we intend to use such data to evaluate the app itself. At the time of writing, MathMelodies is collecting data from about 200 users, more or less 10 times as much as we could reasonably expect to involve in the evaluation conducted with physical presence of the users. Another future improvement consists in developing a collaborative system that allows the final user (or the teachers) to directly collaborate in the development of the app content. The definition of this crowdsourcing system can drastically reduce the development costs of the next versions of MathMelodies and ease the scalability of this solution.

The second solution is AudioFunctions, a tablet prototype that makes it possible for visually impaired students to explore function graphs. AudioFunctions presents three exploration modes. The first interaction mode, called "non-interactive", extends an existing solution [22] with a new sonification technique. The second, "mono-dimensional interactive" is similar to the first but implements a probing approach instead of a scanning approach [93]. Finally, "bi-dimensional interactive" allows the user to follow the graphical representation of the graph, similarly to the "distance to edge" sonification introduced by Yoshida et al. [94]. AudioFunctions also introduces several improvements over traditional desktop applications to explore function drawings, like Audio Graphing Calculator and Sonification Sandbox [27, 87]. First, being a tablet application, AudioFunctions enables direct interaction with the tablet's touchscreen, thus allowing users to take benefit from proprioception. Second, AudioFunctions adopts additional sounds to represent function features like, for example, local minimum and maximum, concavity and intersection with the origin. The experimental evaluation conducted with 7 users shows that AudioFunctions makes it possible to obtain a much better understanding of the the function graph than existing software solutions. AudioFunctions allows the users to have a better understanding also when compared to tactile paper. This was not expected, as the users only trained with AudioFunctions for a few minutes, while they were all acquainted with mathematical exercises on tactile paper.

While the aim of AudioFunctions was to explore different interaction paradigms, as a future work we intend to focus on the sonification technique, to compare different solutions and identify the one that best suites each exploration mode. In particular, a sonification technique to sonify multiple values at the same time should be investigated. We also plan to engineer AudioFunctions and distribute it on the AppleStore. This would allow a large distribution of the app, which in turn can have positive effects on future research. Indeed, by remotely collecting usage data, it could be possible to evaluate the solution with a much larger number of users, possibly in the order of hundreds or thousands.

[13] https://itunes.apple.com/us/app/math-melodies/id713705958?mt=8

Experience made with MathMelodies and AudioFunctions inspired us to investigate more general approaches to enable access to images and, in particular, to binary drawings. We deliver on this objective in our third solution, Invisible Puzzle. Indeed, Invisible Puzzle introduces three main contributions to the state of the art. The first contribution consists of six novel sonification modes. A single sonification mode is based on a bi-dimensional exploration paradigm adopting an approach similar to the "bi-dimensional interactive" one described in Section 6.2.2. The remaining five modes adopt a mono dimensional exploration paradigm similar to the technique proposed by Dallas [22]. However, our paradigm extends the previous solution on two aspects. First, it adopts a probing approach instead of a scanning approach, making users choose the portion of image to be sonified. Second, it sonifies image features along a horizontal line instead that on a vertical line, adopting sound spatialization and sound equalization filtering to represent the horizontal position of (possibly) multiple image features at the same time. Another difference between Invisible Puzzle and existing solutions is that new types of sound generators are evaluated. Invisible Puzzle implements a sound generator based on pure tones similar to what is implemented in other contributions [22, 60, 94], but also a sound generator based on pink noise and one adopting music files.

A second contribution is a methodology to compare sonification modes addressing two main challenges: to enable the quantitative comparison of sonification modes and to automate the provision of tests in order to conduct a large number of tests with limited supervision effort.

The third contribution is a mobile application adopting the illustrated methodology to evaluate the novel sonification modes. The application has been used to perform tests with about 200 users (49 blind) and the statistical analysis of the automatically collected data highlighted statistically significant differences between the performance of different sonification modes.

Invisible Puzzle can be extended along a number of directions. First, it could be possible to adopt more sophisticated techniques to evaluate how clearly a subject identifies a figure. With the current version of Invisible Puzzle subjects explore a figure and then have to identify it among a set of alternatives. An different solution consists in asking the user to first explore a hidden image and then to redraw it on the device. The drawn shape can then be automatically compared to the hidden one. We believe that this solution, possibly coupled with the already adopted multiple choice question, could give more insights on the actual user's understanding of the image.

Another possible improvement consists in evaluating to which extent more complicated images can be perceived. This also include the use of grayscale images and, by designing new sonification modes, color images. While in theory the three solutions based on 1-D FF can be already used on real-world grayscale images, their effectiveness for this kind of application needs to be carefully evaluated.

As a future work we also intend to take into account the impact of ambient noise on the performances. Indeed, while our study was conducted in environments with low ambient noise, we cannot exclude that this factor could have an impact. This consideration is even more important if we consider the public version of Invisible Puzzle that can be used anywhere.

Finally, we intend to introduce additional features to make exploration more interactive, like the possibility to zoom in and out in the image or the separation of the image into layers, so that each layer can be explored separately from the others.

Solutions presented in this chapter highlighted a common issue: conducting user evaluations required an overwhelming effort. For example, a large amount of time was devoted in finding people with VIB willing to take part in the evaluation and in reaching them to administer the test. This issue has been partially addressed in Invisible Puzzle, where provisioning of tests is simplified by automatically providing training, administering tests and collecting usage data. However, the problem of identifying and reaching test subjects is left to be solved. A solution to address this issue is presented in the next chapter.

Chapter 7

Large Scale Evaluation

In previous chapters we introduced several Mobile Assistive technologies to support people with VIB. The design of these technological solutions is typically guided by supervised experiments with few participants, such as formative studies [90], Wizard-of-Oz experiments [15], and evaluation studies [59]. These approaches may be attractive for the advantages they offer. Researchers can conduct experiments with prototype applications, or in some cases, even prototypes without working software. They can also conduct such experiments in controlled situations and with users whose characteristics (e.g., form of disability, age) are known in advance. However, these approaches are also limited in many ways. First, it is not possible to explore many real world scenarios. Second, these studies generally involve participants that live in close proximity to the physical location where the experiment is conducted, leading to the possibility of cultural bias. Third, these experiments are susceptible to the Hawthorne effect [2], where users may act differently when they know they are being watched. Finally, and most important, these approaches are not scalable both in terms of number of involved subjects and length of the study, as stressed in previous research [31].

In this chapter we present how we adopted an evaluation methodology typical of HCI research to address these issues. The idea is to collect real world usage data from assistive applications published on the App Store. Collected data is then analyzed both with inferential and exploratory methods using statistical tools, in order to identify usage properties of the application like, for example, commonly used functions and preferences for applications settings. In the following we introduce the remote data collection system we developed for the task and the evaluation methodology adopted to evaluate a well known MAT: *i*Move.

7.1 Remote data collection

In order to collect usage data from applications published on mobile app stores we developed a remote logging system called *Icarus*. The system is made of three components: a client library that can be easily integrated in iOS applications, a REST server and a non-relational database back-end (see Figure 7.1).

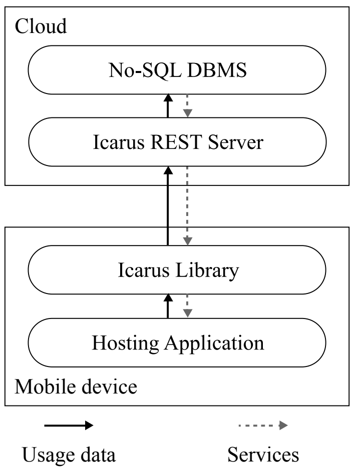


FIGURE 7.1: Architecture of the Icarus remote logging system.

The client library has methods to support sending data to the remote server, allowing the programmer to log any data structure that can be represented through JSON[1], a popular data-interchange format. The library tolerates a faulty network connection and automatically caches unsent data, scheduling it for future delivery. The library also has the added functionality of collecting and sending data about exceptions, events that disrupt the normal flow of the program's instructions and may crash an application. This functionality is particularly useful to debug application errors that may be hard to produce in the local testing environment of the programmer like, for example, errors due to different languages or localizations.

Data is collected in compliance with European regulations on data protection[2] and logs are recorded in anonymized form. Each log includes a unique pseudo-identifier associated with an anonymized user. A pseudo-identifier univocally identifies a pair ⟨ anonymized user, mobile application ⟩, and is generated the first time a mobile application adopting *Icarus* is launched on the user's device. It is important to note that the pseudo-identifier contains no information that could be used to discover the user's identity. Pseudo-identifiers are used to filter logs in order to find those originated by the same user of an application and reconstruct interaction history.

Each log record has two main components. The first component is automatically populated by the library and contains data about the user and the device on which the application is running: the user's pseudo-identifier, the device model, the system language, whether VoiceOver is enabled or not, the application version and log creation timestamps in the user's time zone and UTC. The second component contains custom data specified by the programmer. Figure 7.2 shows a log record collected by the Icarus library.

The REST server has two main functionalities: it serves as a front-end to the non-relational database and provides an extensible architecture to allow the definition of "services", programs that can perform queries on the database and return data to applications. For example, we developed a service to perform aggregation queries on the database and return data for a real-time dashboard showing statistics about Invisible Puzzle.

[1]http://www.json.org

[2]Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data, OJ L 281, 23.11.1995, 31-50.

```
{
    "_id" : ObjectId("5728b94cf4adc31c0585c06e"),
    "appdata" : {
        "uuid" : "CBFF5BDF-42B0-4FF8-9374-1F6FCAD428E8",
        "voiceover" : true,
        "device" : "iPhone8,2",
        "lang" : "en-US",
        "appname" : "Invisible Puzzle",
        "appversion" : "43"
    },
    "timestamp" : {
        "utc" : ISODate("2016-05-03T14:45:01.014+0000"),
        "user" : ISODate("2016-05-03T09:45:01.014+0000")
    },
    "debug" : false,
    "userdata" : {
        "challenge" : NumberInt(2),
        "event" : "exploration",
        "level" : {
            "id" : "01C",
            "attempt" : NumberInt(8)
        },
        "time_exploration" : 13.52493405342102,
        "headphones" : true,
        "available_sonifications" : [
            "INTERSECT_PURE"
        ],
        "sonification" : "INTERSECT_PURE"
    }
}
```

FIGURE 7.2: Example of a log record collected by Invisible Puzzle.

Finally, we adopt the MongoDB[3] non-relational database to store log records. We chose this solution because MongoDB is a document-oriented database that does not require to specify a fixed schema. The structure of our log records is different between applications and even records from the same application may change structure over time. The absence of a fixed schema allows the system to adapt to log structure modifications with no human intervention either on the server or database. An additional motivation for the adoption of MongoDB is its ability to scale and to perform distributed data aggregation tasks.

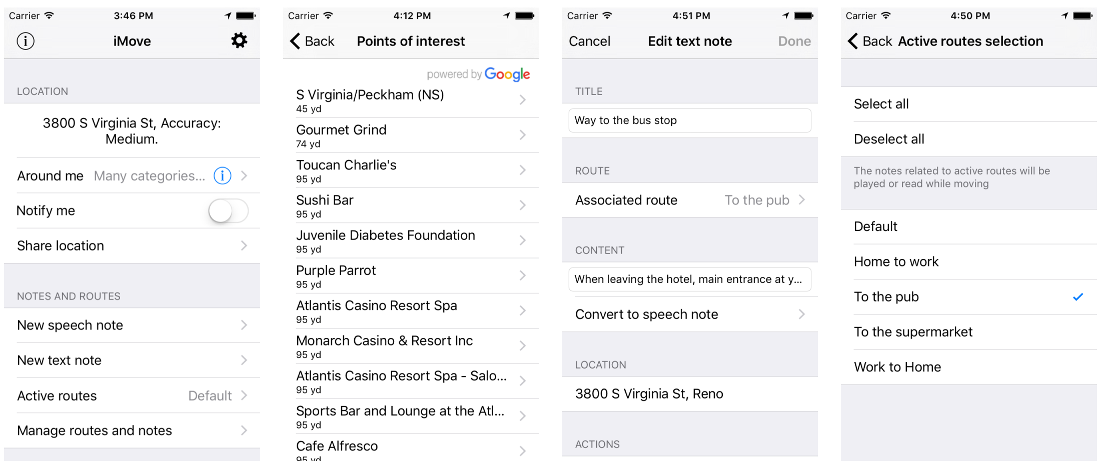
We integrated *Icarus* in mobile applications published by *EveryWare Technologies*[4], a spin-off company of the University of Milan founded by members of our research group. Particularly, *LightDetector*, *i*Move, *Invisible Puzzle* and MathMelodies[5]. The library has been gradually integrated, starting with *i*Move, from December 2015 and collects a monthly average of about 200.000 log records from 4.300 distinct users.

7.2 Evaluating *i*Move

*i*Move is an iOS application that is accessible through the VoiceOver screen reader and magnifier. The app informs users about outdoor geo-referenced information such as current address, nearby Points Of Interest (POIs), and *geo-notes* i.e., user-defined notes associated to a geographical location. Users can access this information either explicitly, e.g., ask for current address in the root screen (Fig. 7.3(a)) and list of nearby POIs (Fig. 7.3(b)), or periodically while in motion

[3] https://www.mongodb.com/
[4] http://www.everywaretechnologies.com
[5] *LightDetector*: https://itunes.apple.com/us/app/light-detector/id420929143
iMove: https://itunes.apple.com/us/app/imove/id593874954
MathMelodies: https://itunes.apple.com/us/app/math-melodies/id713705958
Invisible Puzzle: https://itunes.apple.com/us/app/the-invisible-puzzle/id1051337548

by turning on the "Notify me" toggle button in the root screen. The frequency of such periodic updates can be tuned both in terms of time and proximity (i.e., a minimum temporal/spatial distance between two readings). Geo-notes can be created and edited as audio recordings or text entries (Fig. 7.3(c)) and they are organized into "routes" (Fig. 7.3(d)).



(a) Root screen. (b) POI screen. (c) Edit text-note screen. (d) Route selection screen.

FIGURE 7.3: Main screens of the iMove application.

*i*Move is designed to be highly customizable: users can specify the categories of POIs they are interested in, activate automatic readings of surrounding information, and modify settings related to system verbosity. Therefore, beyond user visited screens, actions, and received notification, we also collect data related to their settings modifications.

7.2.1 Dataset overview

Since *i*Move version 2.0, released on December 8, 2015, the application implements the *Icarus* remote logging system we introduced in Section 7.1 that makes it possible to collect anonymous app usage information. In *i*Move, we partition log entries into four different categories of usage data:

Screen logs capture user navigation between *i*Move screens. Each screen log records the screen name and an "enter" or "exit" label when a user enters or exits a screen.

Action logs record *i*Move function activation by a user such as recording a new speech note.

Notification logs are generated when the application automatically provides information to the user (e.g. when the user gets close to a POI).

Preference logs are generated every time a user changes *i*Move settings. A preference log lists the name of the modified parameter, its old value, and its new value.

The *i*Move dataset was collected during the December 2015 - April 2016 period and contains a total of 771, 975 log records across 17, 624 unique user pseudo-identifiers ($\mu = 43.8, \sigma = 105.15$)

log records per user with range 1 - 7,299. A detailed description of the released dataset is available online[6].

From the feedback we received by email and on the appstore, we realized that a number of users, who we call "incidental" users, installed the application without realizing its functionality and its intended use for people with visual impairments. For example, some users confused iMove with *iMovie*, a popular application for video editing.

To filter out these users, we introduce the concept of "interaction session" (or simply, session): a period of time during which a user frequently interacts with the application (e.g., navigates in the screens, performs actions or receives system notifications). A session is extracted from app usage data as a sequence of consecutive log entries such that: i) the sequence begins with a "screenRootEnter" record, which signals that the user opened the main screen of the application, and ii) there is at least a 5 minutes gap between the session starting log and the previous log. This constraint captures the intuition that the user might temporarily exit the app for a short time within an interaction session.

Based on the intuition that users who are uninterested in iMove would not use it for more than one session, we consider only users having two or more sessions. There are a total of $4,055$ such users generating a total of $255,004$ logs ($\mu = 62.89, \sigma = 211.51$ logs/user with range 2-7,296).

7.2.2 Use properties across all users

We analyze log records from all $4,055$ users with the goal of highlighting iMove use properties such as commonly used functions and user preferred values for interaction parameters. Using both inferential and exploratory methods we examine four categories of log records: preferences, screen activity, actions, and notifications.

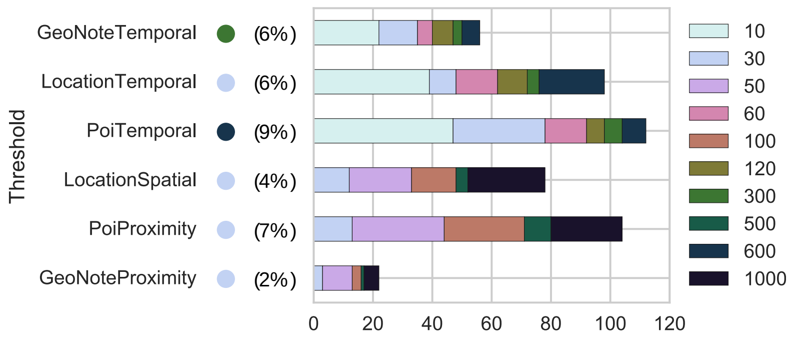
One interesting aspect of iMove is the support of user-defined geo-notes, where users can either record a speech note associated with a location or type it as text. While both options are available, we expect that the former will be the one adopted by the users since the purpose of the app is to support mobility and it is has been observed that typing in mobility is particularly challenging for people with visual impairments [58]. Specifically, we formulate and examine the following hypothesis:

H1: iMove users will favor speech over text for input modality when creating geo-notes.

Results and interpretation

Preference logs account for 3.41% of the total log records. Figure 7.4 reports, for each preference setting, its default value and how many times it has been set to a given value. We observe that the parameter "keepUserInformed", which toggles all notifications, was changed far more frequently. This interaction was expected by our intuition that users will frequently toggle off when they do not want to be disturbed by notifications. Anticipating such an interaction during the design of iMove, we positioned the toggle button in the root screen (see Figure 7.3(a)).

[6]http://webmind.di.unimi.it/assetsim16/



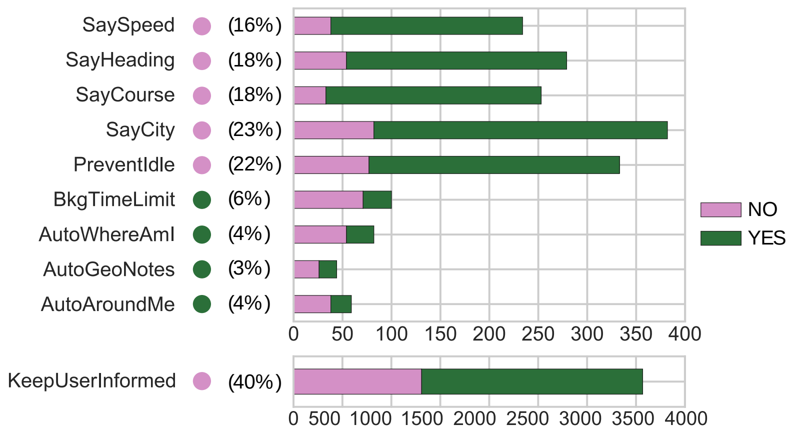


FIGURE 7.4: Number of preference records generated from the subset of users that modified the default values. Colored circles indicate preferences' default values, while the percentages represent how many users changed the value at least once, among those who visited the corresponding settings screen.

Indeed, 22.2% of the users changed this value twice or more, while 20.9% of the users changed it more than once for at least one session.

We also explore log records for other parameters, whose semantics are detailed online[7], to assess the default values provided by *i*Move. This analysis cannot take into account only the values changed by the users. Since all logged changes necessarily involve modification of default values, the logged data does not inform us of how many users intentionally choose to stick with the default value for a given parameter. To estimate this, we compute, for each parameter, the percentage of users that changed the parameter value at least once, among the users that actually visited that parameter's settings screen (values are reported in Figure 7.4).

For example, only 4% of the users who entered the "Settings_location" screen actually changed the value of the "locationSpatialThreshold" parameter. On the other hand, 22% of the users who entered the System settings page changed the "prevent screen lock" option that by default is set to false. Similarly, 23% of the users changed the preference "sayCity" and more than 16% of the users changed the "saySpeed", "sayHeading" and "sayCourse". These are parameters whose default values are candidates for change in future versions of the app. More generally, we observe the four parameters above are all related to the type of information provided to the user

[7] *i*Move parameter semantics is detailed in `http://webmind.di.unimi.it/assetsim16/#param_semantics`.

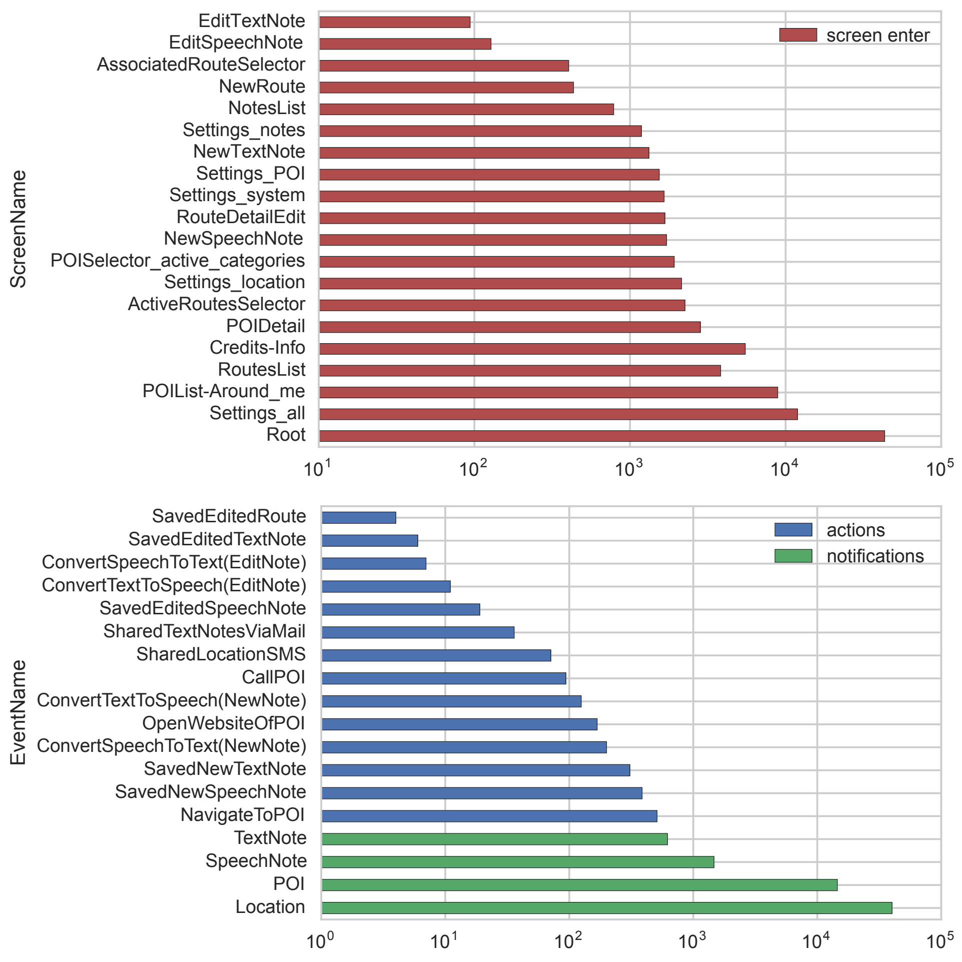
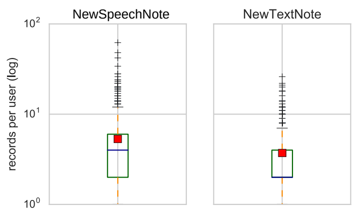


FIGURE 7.5: User log records distributed across the screens, actions, and notifications logs.

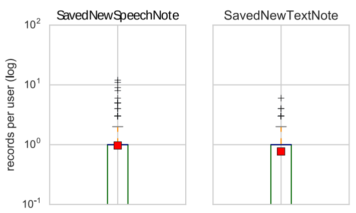
when a location notification occurs. To avoid verbosity in the application, we limited location notifications to the name and number of the street by default. Apparently, many users prefer to have more detailed information.

Screen, Notification, and Action logs account for 66.23%, 29.55%, and 0.76% of the total 255, 004 log records, respectively. Figure 7.5 illustrates the distribution of these records across the subsequent categories. We observe that "Location" is the most common notification followed by "POI" and the two geo-notes. Interestingly, the "NavigateToPOI" function, suggested by many users and introduced with app build 31, is the most frequent user action. Geo-notes notifications ("SpeechNote" and "TextNote") are less frequent than "Location" and "POI" notifications, accounting for 3% of the total notifications. This is due to the fact that 83% of the users never created a geo-note. Among users creating a geo-note, the percentage of geo-note notifications is 10% of the total notifications.

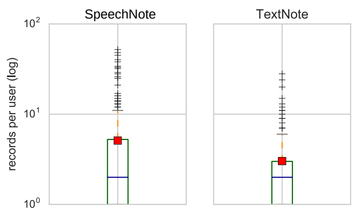
Figure 7.6 shows the distributions of per-user screen, action, and notification logs related to speech and text geo-notes (box indicates quartiles, center-line indicates median, square symbol indicates mean, whiskers indicate 1.5 inter-quartile ranges, and crosses indicate outliers). In *support of hypothesis H1*, there is a significant difference between the pairs of these graphs determined by Mann-Whitney U test. Specifically, users visit the "NewSpeechNote" screen significantly more times than the "NewTextNote" screen ($p < 0.001$) and perform significantly more



(a) Screens.



(b) Actions.



(c) Notifications.

FIGURE 7.6: Distribution of log records highlighting differences between speech and text geo-notes logs.

"SavedNewSpeechNote" actions than "SavedNewTextNote" actions ($p < 0.05$). Not surprisingly, users receive significanlty more "SpeechNote" notifications than "TextNote" notifications ($p < 0.05$).

7.2.3 Voiceover-based user comparison

As mentioned in Section 7.1 for each log record we collect the VoiceOver field, which reports whether VoiceOver was active when the record was generated. This field is particularly relevant for our analysis as it allows us to distinguish users that are likely to have severe visual impairments. Therefore, we partitioned the *i*Move users into two groups: VO-group users (VO-users) have at least one VoiceOver-active record and NVO-group users (NVO-users), have no VoiceOver-active records.

We formulate and examine the following hypotheses:

H2: VO-users will have different settings preferences than NVO-users.

H3: VO-users will make more intense use of *i*Move as measured by the number of actions and notifications as well as the span of days using the app.

Results and interpretation

The VO-group consists of 1,025 users whereas the NVO-group includes the rest 3,030 users. We observe that while VO-group includes a smaller percentage of the overall *i*Move users (25.28%), the number of records generated by this group accounts for more than half of the logs (56.34%) along with a higher mean records per user ($\mu = 140.16, \sigma = 403.91$) than the NVO-group ($\mu = 36.74, \sigma = 45.05$). We also observe a small positive correlation in our dataset between the number of records for a user and the percentage of records with VoiceOver activated for the same user.

Users in VO-group generated logs with a high mean percentage of active-VoiceOver records ($1\% - 100\%$, $\mu = 95.26\%$, $\sigma = 16.6\%$). This suggests that, while by definition a user in VO-group can only have one record with VoiceOver-active, in practice users in VO-group have VoiceOver activated almost all the time during use of *i*Move. We suspect that users in VO-group are mostly people with severe visual impairments and a few users with low vision that sporadically activate

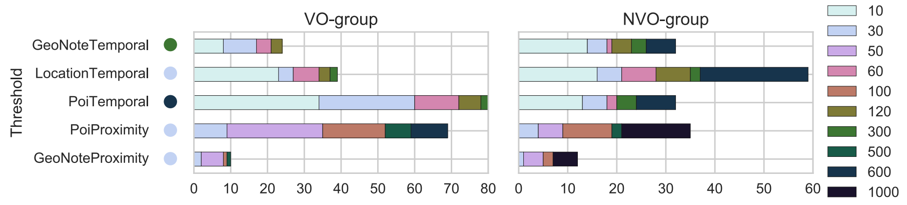
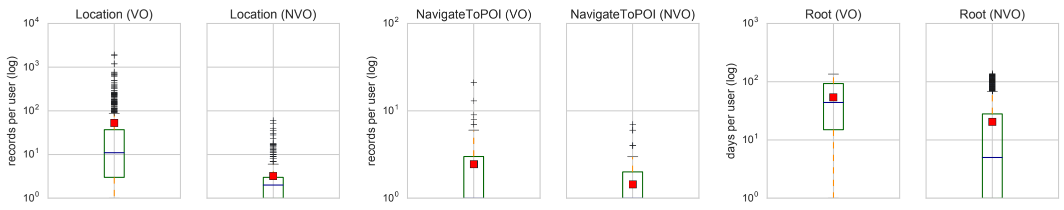


FIGURE 7.7: Preference log records across users in VO-group and NVO-group.



(a) Location notifications.
(b) Navigate to POI action.
(c) Period of use.

FIGURE 7.8: Differences between users in VO-group and NVO-group.

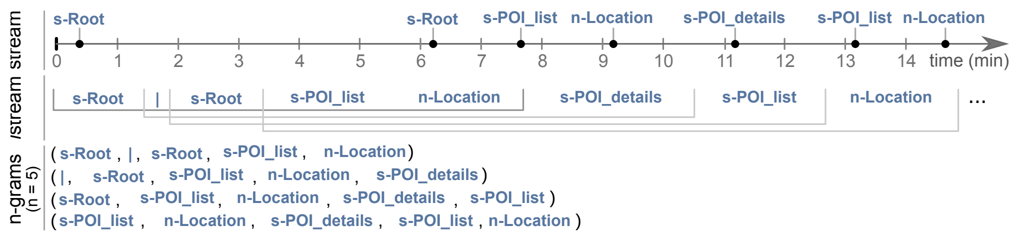
VoiceOver while users in NVO-group either use magnifier in their interaction with the app or are non visually impaired ("incidental" users, see Section 7.2.1).

Figure 7.7 illustrates side-by-side the distribution of threshold preference from both groups. In *support of hypothesis H2*, we find that users in VO-group set smaller temporal and spatial threshold values determined by Mann-Whitney U test ($p < 0.05$). Even though different threshold parameters have different semantics, smaller temporal values result in more frequent notifications, while smaller spatial values for "PoiProximity" and "GeoNoteProximity" indicate preference for notification only in close proximity to the target place (POI or geo-note). These findings suggest that users in VO-group prefer to receive information more frequently than users in NVO-group and only in close proximity to the target.

To examine hypothesis H3, we consider the number of notifications and actions, as well as the period of *i*Move use per user in each group and compare their mean ranks with the Mann-Whitney U test. In *support of hypothesis H3*, we find that users in VO-group receive significantly more notifications ($p < 0.001$) such as the "Location" notifications shown in Figure 7.8(a)). Similarly, users in VO-group perform significantly more actions ($p < 0.001$), for example Figure 7.8(b) shows how the number of times a VO-user asks for directions to navigate to a POI is significantly higher than for a NVO-user. Users in VO-group use the application for a significantly longer period than the NVO-users ($p < 0.0001$), where the period of use is measured as the span of days between the first and last time a user enters the *i*Move root screen. On average, this duration is of 53.95 days for users in VO-group and of 20.45 days for users in NVO-group (as shown in Figure 7.8(c)).

7.2.4 User clustering based on *i*streams

While the exploratory and inferential analyses in the previous sections reveal interesting patterns, they do not take into account the sequential relationship between the log entries. In order to learn richer patterns of interaction, we use unsupervised learning techniques on record streams, which preserve the temporal structure of the data. We anticipate that users naturally fall into



stream
istream
n-grams (n = 5)
s-Root
s-Root
s-POI_list
n-Location
s-POI_details
s-POI_list
n-Location
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 time (min)
s-Root | s-Root s-POI_list n-Location s-POI_details s-POI_list n-Location ...
(s-Root , |, s-Root, s-POI_list, n-Location)
(|, s-Root, s-POI_list, n-Location, s-POI_details)
(s-Root, s-POI_list, n-Location, s-POI_details, s-POI_list)
(s-POI_list, n-Location, s-POI_details, s-POI_list, n-Location)

Figure 7.9: Mapping interaction streams to n-grams.

clusters based on common interaction patterns with *i*Move. The automatic discovery of these clusters can help us identify: what are the major interaction categories; which is the most prevalent interaction; and what is the relationship between different types of interactions. This clustering is performed on the 1,025 users residing in VO-group, who are likely to have severe visual impairments and, as shown above, make intensive use of the application.

HCI researchers have adopted prior work in machine learning, natural language processing and network analysis, to better understand user behavior, with the social network analysis performed by Wang et al. [88] being the closest to our work. Our methodology builds upon previous methods to understand and support assistive orientation of people with visual impairment. One of the inherent challenges in analyzing our data is that users can interact with the app either by actively navigating the screens and using their functions, captured by screen and action logs, or by physically changing their location thus generating notifications logs. We introduce the notion of sessions (defined in Section 7.2.1) into our feature engineering (described below) to yield more intuitive and high level descriptions for the discovered clusters.

Specifically, we represent each user by the stream of interactions (*i*stream) with the app. We map users to a feature space extracted from these streams, construct a similarity graph by comparing users in this feature space, and identify clusters of similar users by graph partitioning. Finally, we interpret the meaning of the clusters by isolating primary features that are responsible for forming the clusters. To assist future researchers in adopting this methodology for analysis of their data, we describe the above steps, implementation, assumptions, and the hyper-parameters used in our clustering.

Obtaining user *i*stream. We define an *i*stream as a sequence of interactions between the user and *i*Move, extracted from user's log records ordered by timestamp. It captures both the type of the log entry (i.e. screen, action, or notification) and the magnitude of time gaps between two consecutive log entries. Precise time gap values are omitted if the log entries belong to the same session (defined in Section 7.2.1) and are represented by the symbol "—" if they denote session boundaries. Figure 7.9 illustrates an example of this approach for obtaining a discrete user *i*stream.

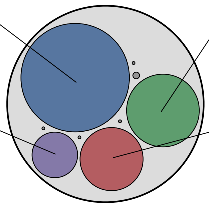
Mapping users to an intuitive feature space. We treat *i*streams as text sentences and adopt n-gram-based text representation, a common practice in natural language processing. We consider three classes of records: screen enters, actions and notifications. Each of these three classes is defined as a set of atomic strings, which are dented by A_s (screen enters), A_a (actions), and A_n (notifications). For example, the string "s-Root" $\in A_s$ represents an entrance in the root screen; "a-navigateToPOI" $\in A_a$ represents the action of getting the navigation instructions to a POI; and "n-Location" $\in A_n$ represents the location notification. We define an *i*stream as

c1 (370 👤): Check list and details of nearby POIs

feature space	**normalized frequency** in cluster	not in cluster
(s-POI_list, n-Location, s-POI_details, s-POI_list, n-Location)	0.013078	0.001167
(s-Root, |, s-Root, |, s-Root)	0.002553	0.073674
(n-Location, n-Location, n-Location, n-Location, n-Location)	0.000192	0.046369

c4 (154 👤): Short sessions with few notifications

feature space	**normalized frequency** in cluster	not in cluster
(|, s-Root, n-Location, |, s-Root)	0.024224	0.001708
(n-Location, |, s-Root, n-Location, |)	0.036877	0.001331
(|, s-Root, n-POI, n-Location, |)	0.023818	0.001058



c2 (247 👤): Long sessions of location and POI notifications

feature space	**normalized frequency** in cluster	not in cluster
(n-Location, n-POI, n-Location, n-Location, n-Location)	0.021846	0.000411
(n-POI, n-Location, n-Location, n-Location, n-Location)	0.016155	0.000411
(n-Location, n-Location, n-Location, n-Location, n-Location)	0.175919	0.000615

c3 (215 👤): Very short sessions of only Root screen entrances

feature space	**normalized frequency** in cluster	not in cluster
(s-Root, |, s-Root, |, s-Root)	0.222907	0.001778
(|, s-Root, |, s-Root, |)	0.186638	0.000567
(n-Location, n-Location, n-Location, n-Location, n-Location)	0.000361	0.056407

FIGURE 7.10: Clustering results highlighting the four main identified clusters.

a sequence $S = (s_1 s_2 ... s_m)$, where $s \in A_s \cup A_a \cup A_n \cup \{|\}$ and m is the total length of the $i$stream. We define F_n as the set of all possible n-grams (n consecutive elements) from all the users' $i$stream sequences: $F_n = n\text{-gram}(S_1) \cup n\text{-gram}(S_2) \cup ... \cup n\text{-gram}(S_{\#users})$. For each user $i$stream we calculate the normalized frequencies of the n-grams in F_n. We experimented with different values of n in the n-gram and chose 5-grams for our analysis, though 4-grams and 3-grams reveal similar clusters. As discussed by Wang et al. [88], intuitively, a larger value of n for the n-gram captures longer subsequences that are unlikely to repeat as a pattern in the $i$stream. For the above calculations we use the NLTK platform [13].

Constructing a similarity graph. We create a fully connected graph where each node represents a user and each edge between a pair of users represents the weight based on their pairwise similarity score. To calculate the similarity score between two users, we compute the cosine similarity of their n-gram feature vectors using scikit-learn [68].

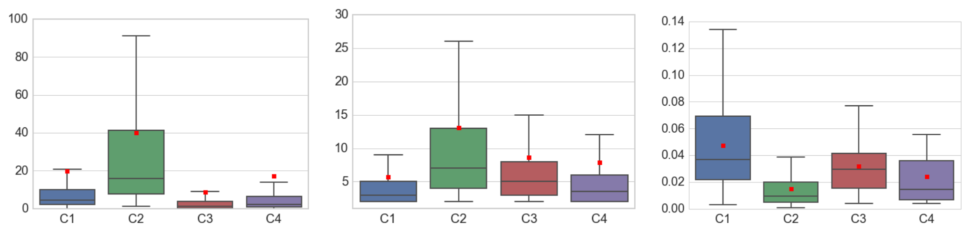
Clustering and identifying primary features. We partition the graph into clusters of similar users with community detection using the Louvain method[8] [14]. To interpret cluster meaning, we isolate the primary features responsible for a cluster by performing feature selection based on Chi-square statistics (χ^2) [92]. For each cluster, we build a classifier that distinguishes users belonging to that cluster from the remaining users. Then we select the top k features with the highest discriminating power in separating the two classes using the "SelectKBest" method from scikit-learn [68].

Results and interpretation

The clustering procedure generates 9 clusters with a modularity of 0.47, where modularity [63] is a widely-used metric to assess the quality of a graph's partition into communities. Loosely speaking, it measures the density of edges inside clusters to edges outside clusters with values in the $[-1, 1]$ range, where a higher value indicates better clustering. Five of the detected clusters contain a total of 6 outlier users which we omit from the following discussion, hence focusing on four clusters with many users. Figure 7.10 visualizes the resulting clusters and the top 3 features with the highest discriminating power per cluster.

The first cluster (C1) contains 370 users. From the 5 primary features: two indicate that short sessions, in which the user simply opens the application without further interaction, appear with lower normalized frequency for users in C1 than those outside C1; one indicates that long sessions with many consecutive location notifications appear with low frequency as well; last, the remaining two primary features indicate that sessions in which the user navigates iMove screens

[8]Library: `http://perso.crans.org/aynaud/communities`



(a) Sessions length (minutes). (b) Number of sessions. (c) Ratio of s-POIdetails records.

Figure 7.11: Analysis of the four clusters.

with the list of POIs and their details have higher frequency for users in C1 than the rest. We can infer that users in this cluster often open the application to check the list of nearby POIs and their details.

The second clusters (C2) contains 247 users. From the 5 top features characterizing this cluster, three indicate high frequency of location and POI notification sequences in a single session for users in C2; and the remaining two primary features indicate low frequency of “empty” sessions, e.g., “— screenRootEnter — screenRootEnter —”. These features suggest that C2 is a set of users running the application for long sessions during which they frequently receive many location and POI notifications.

The third cluster (C3) contains 198 users. In this case four of the 5 primary features denote high frequencies of short “empty” sessions; and feature points to lower frequency of consecutive location notifications within the same sessions for users in C3 than outside C3. These features suggest that C3 contains users that start the application, do not wait for any notification, and then close the application. We speculate C3 users often open *i*Move simply to read (though VoiceOver) the current address.

The fourth cluster (C4) contains 154 users. All 5 primary features have high frequencies of short sessions with some location and POI notifications. Our interpretation is that these users start *i*Move and listen to one or two notifications without any further interactions.

To get a confirmation of the semantics we associate to each cluster, and to further study these clusters, we analyze user characteristics across clusters. We consider the average session length per user, computed as the distance between the timestamps of the last and first records in each session. As shown in Figure 7.11(a), users in C2 have longer sessions than other users. This supports our earlier interpretation based on the primary features. Figure 7.11(b) shows that users in C2 also have a higher number of sessions, followed by users in C3 and C4. We can interpret this observation in two ways. First, given the particular use of the app (keeping *i*Move active while moving), users in C2 tend to use it more frequently (e.g., every day, commuting to work). A second interpretation is that more experienced users of *i*Move tend to use it for longer sessions and hence belong to C2. Distinguishing these two cases requires additional analysis that we leave as future work. Last, Figure 7.11(c) shows that C1 users have a higher rate of records corresponding to POI details screen enters. This is in support of the primary features extracted for this cluster, identifying C1 as a user group with an higher frequency of sessions that explore POI-related screens.

7.2.5 Discussion

This section presents an analysis of users interactions with *i*Move, a mobile app that supports the orientation of people with visual impairment. The initial dataset contains more than 17,000 users, many of which are "incidental" users, not really interested in the functions of the app. To filter these users out, we adopted a session-based heuristic that eliminates 77% of the users and 67% of the log records.

The data analysis performed on about 4,000 remaining users highlights a number of *i*Move use properties, including commonly used functions and users' preferred values for settings parameters. In summary:

- While initial *i*Move settings favored sporadic and brief notifications, we observed that users, in particular those with severe visual impairments, prefer to have frequent and detailed information about the current location, which should include city, speed, heading and course.
- Applications similar to *i*Move are recommended to activate the "prevent screen lock" option by default.
- *i*Move users favored speech over text for input when creating notes associated to geographical locations.
- We observed that points of interest (POIs) were important in *i*Move functionality. Many users checked the list of nearby POIs (the third most visited screen) and the most popular action was navigating to a POI.
- VoiceOver users (VO-users) received more notifications, made intensive use of core *i*Move functions, and used the app for longer periods than other users. While *i*Move was designed with blind users in mind, the observed differences with non-VoiceOver users, possibly including people with low vision, raises concerns about the app design in support of this population.

*i*Move was designed with a main user target in mind: people with VIB that would keep the app active along a route to get notifications. By clustering about 1,000 *i*Move VO-users based on common interaction patterns, our user target base was successfully identified from one of the major clusters (C2), which contained 25% of the VO-users. In addition, our clustering method was able to capture and provide semantics for the remaining 75% of the VO-users with three more clusters; indicating those users who interact with the app in short sessions. We speculate that users in those clusters avoid interacting with the app while moving, because they do not want to be distracted or do not feel comfortable walking while holding their smartphone. Hence, they use the app in short bursts when they feel comfortable.

The identification of additional user clusters, other than C2, can help improve *i*Move by designing new interaction patterns and functions that support these usage patterns. For example, since many users (those in C1) often open the app to check nearby POIs, it may be possible to optionally show the list of POIs in the first app screen. Similarly, we speculate that users in C1 often open the app to check the current address and then close it. To support these operations, researchers could investigate different interaction modalities like an accelerometer-based interface to determine when the user wants to read the current address while the device is in the user's pocket.

This contribution highlights a number of possible future works. First, the analysis was conducted from data collected in a period of four months during which *i*Move has been downloaded on average more than 4, 000 times a month. We expect the number of users to grow linearly with time so that, in a few months, it will be possible to conduct the same analysis on a larger set of users and adopt hierarchical clustering that can potentially refine our higher-level clusters into more descriptive sub-clusters. On the other hand, collecting data for a longer period will enable better analysis of a user's learning curve and evolution of interactions over time, possibly characterizing the behavior of novice users with respect to experienced ones.

In the future it will also be possible to collect additional types of log data. For example, while it is not possible to collect users' location or user-defined geo-notes due to privacy concerns, it may be possible to collect additional context-related information, like users' speed and whether users are walking or traveling on a bus/car.

Also, understanding of the application may be improved by collecting qualitative data from users. Currently, users can provide feedback by using the "contact the developer" functionality implemented in the "about" screen, which redirects to a pre-compiled email message. We also noticed that users provide valuable feedback in App Store reviews like, for example:

- *Great app. However, can you add support for different measurements such as yards and ft for the us? Also want turn by turn directions.*
- *i'v tested this app on a number of routes i frequently use and have found it to be extremely useful. the app works well when used in conjunction with other GPS apps such as Navigon. menus work well with Voiceover and speech notes work as expected. one feature I would like to see added and would certainly find useful is the option of backing up speech notes to Icloud, offering the option of deleting old notes would also be beneficial. thanks, Paul.*

An issue of this approach to feedback collection is that there is no way to associate each feedback with the pseudo-identifier of the user who originated it, hence preventing the reconstruction of her interaction history. As a future work, we should investigate a way to allow the collection of qualitative feedback from within the application.

From the point of view of users' clustering, there are three directions along which we intend to extend this contribution. First, we want to explore hierarchical clusters and dimensionality reduction approaches that can further improve our clustering quality and preserve an interpretable feature space. Second, we intend to investigate the link between preferences for user settings and the automatically detected user clusters. Third, we intend to experiment with clustering techniques for effectively identifying "incidental users" so that it is possible to remove them more reliably.

We see the results, methods, and data provided in this section to improve existing applications, provide guidance, and advance the state of art in the field of assistive orientation and navigation – ultimately leading to a better experience of independent mobility for people with visual impairment.

7.3 Summary and future directions

In this chapter we present two contributions. First, we illustrate *Icarus*, a remote logging system that has been developed to enable the collection of large scale usage data from mobile applications. The system has been integrated in some assistive solutions, presented in this dissertation, that have been engineered by *EveryWare Technologies* and published on mobile app stores. These applications are providing a large amount of usage data from real world users.

It is worth noting that there are contributions in HCI literature that adopt an opposite approach with respect to what is presented in this chapter: to use cognitive models to generate simulated user interactions. CogTool [41] is a software to automate the evaluation of user interfaces by predicting execution times for particular sequences of actions using a human cognitive model. Indeed, the approach based on Icarus has two main advantages over CogTool. First, Icarus collects usage data that better represent the target population of users with VIB as, to the best of our knowledge, the cognitive model adopted by CogTool does not simulate interaction from subjects with VIB. Second, Icarus collects usage data originating from real world usage scenarios that could provide useful insights about the context of use of the application. As a future work, interaction with *i*Move should be analyzed with CogTool and the results of such analysis compared with those obtained with our approach.

The second contribution presented in this chapter is the analysis of usage data collected from the *i*Move application. The analysis, performed on data originated by about 4000 users, highlighted three main aspects. First, it highlighted a number of usage properties of *i*Move, like the preferred value of some settings. These values can be used to update the default settings in future releases of the application. Second, it allowed to identify the list of the most popular functionalities of the application. Such list may be adopted to drive further development of the application or to identify basic functionalities that should also be introduced in other applications. Third, users have been clustered depending on their usage habits. By observing the distinctive features of the clusters it is possible to understand the typical usage scenarios for each cluster.

Research presented in this chapter inspires many future works. As a large amount of usage data is being collected by the *Icarus* system, new data analysis will be required to reach the following goals: first, to highlight new insights on existing MATs; second, to evaluate novel research ideas implemented as MATs; third, to inspire future accessibility research. Also, new analysis techniques should be investigated to allow a better understanding of the collected data. For example, techniques based on hierarchical clustering and machine learning.

Finally, in order to collect qualitative feedback from users, a new functionality should be added to the *Icarus* system to occasionally present questions to the user while an application is being used. For example, users may be asked to answer questions about the usability of a specific functionality. Also, quantitative data already known to the library may be used to target questions to specific users. An example is to inquire subjects who often use a particular functionality about possible changes to be introduced.

Chapter 8

Conclusions

Mobile devices are multipurpose tools that, thanks to innovative features like the presence of onboard sensors (e.g. GPS, accelerometers), cameras, an ubiquitous internet connection and enough computing power, can be used for a large variety of tasks. For example, they can be used to make and receive calls, surf the web, listen to music, navigate unknown environments and play games. The source of this multipurpose nature is the ability to install applications developed by third party developers that extend the device's capabilities.

Mobile devices and, particularly, smartphones are accessible to people with VIB and, by developing applications implementing MATs, they can be used to support people with VIB in many daily activities. Indeed, a person with VIB often relies on support devices. For example, the white cane is used to explore a person's immediate surroundings, light detectors are used to perceive the amount of light in an environment and tactile drawings enable access to visual information like maps, function diagrams and charts. However, the traditional approach based on multiple, single-purpose, support devices has some limitations. For example, devices must be bought by the user and they may be expensive. Also, they must be carried around and they may be large and heavy.

Mobile applications implementing MATs, can address these issues and present many advantages over the traditional approach. First, they can replace multiple tools regularly used by people with VIB, like the light detector and tactile maps. For example, we have shown in Chapter 6 how a MAT can be used to represent function diagrams, an activity that otherwise poses some challenges. Second, MATs can be used together with existing tools to augment their possibilities. For example, in the context of urban navigation, the white cane can be augmented by a MAT that, adopting computer vision techniques, detects features that are out of the range of the white cane, like the presence of a crosswalk and the state of pedestrian traffic lights. We addressed this scenario in Chapter 4 and Chapter 5. Finally, MATs enable new activities that cannot be performed without relying on mobile devices like, for example, autonomously reading text with OCR while on the move [20] or crowdsourcing the description of objects [11].

An issue encountered since the early stages of our research is the difficulty in finding test subjects with visual impairments. This problem may hindrance the results of accessibility research on two main dimensions. First, evaluations were conducted with a limited amount of subjects, not enough to obtain statistically significant results supporting the validity of the proposed solutions. Second, evaluations may be conducted with samples of the population that may not be representative of different forms of visual impairment and specific needs of subjects with different demographics (e.g. the elderly, persons with more than one disability, subjects with different instruction).

In order to address these issues, we implement an evaluation methodology that performs "instrumented remote evaluations" [33], automatically providing all the training required to make users proficient with a MAT, administering tests and collecting usage data to be analyzed off-line with statistical methods. The success of evaluations conducted with this methodology depends on the amount of test subjects involved. For example, we used the methodology to conduct the evaluation of Invisible Puzzle (see Section 6.3.6) with 227 subjects (49 with visual impairments). While the amount of subjects involved in the evaluation of Invisible Puzzle is larger than that of other MATs like AudioFunctions, it is still limited when compared to evaluations conducted for traditional applications [69]. The main reason behind this limitation is that Invisible Puzzle still requires to have a test supervisor that identifies and meet candidates and oversees the test.

We propose to broaden the audience that takes part in our evaluations by collecting usage data from applications available to the general public through the App Store. Delivering on this objective poses several challenges. First, research prototypes must be engineered in order to be usable in autonomy by general users on their devices. This effort has a cost that often can not be sustained by research laboratories within universities. Second, users must be engaged by the application and willing to use it over time. This goal can be achieved by including game elements that entertain users and push them in pursuing a challenge of increasing difficulty. Third, the community of person with VIB must be aware of the existence of the application and willing to download (eventually purchase) and use it.

We address these challenges by collaborating with *EveryWare Technologies* (EWT)[1], a spin-off company of the University of Milan founded by members of our research group. Through EWT we engineer our research prototypes, sustaining the involved costs mainly through partnerships with associations for the blind (an example is the *i*Move application, sponsored by *Retina Italia Onlus*[2]) and crowdfunding campains (we did it for MathMelodies[3]). Making MATs available to the general public is still not enough to have a large number of persons use our solutions: a large effort is devoted to advertise our MATs on accessibility websites or blogs [4] and through associations. The advantage of this approach is twofold: first, it allows us to reach a large pool of potential users. Second, feedback received from reviewers and users (collected, for example, in specialized blogs) complements collected usage data to better evaluate MATs and inspire further research.

There are many future research directions in the broad field of MATs. We could extend current research by developing new MATs to address other everyday problems of people with VIB, also integrating existing solutions. For example, a topic that is currently gaining attention in the accessibility research community is that of supporting indoor navigation of people with VIB [5]. Solutions in this field may be integrated with *i*Move, ZebraX and TL-detector to offer a

[1] http://www.everywaretechnologies.com
[2] http://www.retinaitalia.org
[3] https://www.indiegogo.com/projects/math-melodies
[4] For example: http://www.applevis.com, https://groups.google.com/forum/#!forum/viphone

single application to address many issues people with VIB face while navigating in both indoor and outdoor environments. Another interesting direction consists in adopting machine learning techniques to allow MATs to derive information about their user and adapt their interaction modalities and contents to the specific needs of the user.

As a final remark, we highlight that MATs are not useful only for people with VIB: there are many contributions in accessibility research addressing issues of subjects with different disabilities like, for example, reduced mobility [18]. Also, solutions based on MATs can be used to address the challenges posed by Specific Learning Disorders like Dyslexia [71].

Bibliography

[1] Sami Abboud, Shlomi Hanassy, Shelly Levy-Tzedek, Shachar Maidenbaum, and Amir Amedi. Eyemusic: Introducing a "visual" colorful experience for the blind using auditory sensory substitution. *Restorative neurology and neuroscience*, 32(2):247–257, 2014.

[2] John G. Adair. The hawthorne effect: A reconsideration of the methodological artifact. *Journal of applied psychology*, 69(2):334, 1984.

[3] Dragan Ahmetovic, Cristian Bernareggi, Andrea Gerino, and Sergio Mascetti. Zebrarecognizer: Efficient and precise localization of pedestrian crossings. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pages 2566–2571, Aug 2014.

[4] Dragan Ahmetovic, Cristian Bernareggi, and Sergio Mascetti. Zebralocalizer: Identification and localization of pedestrian crossings. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '11, pages 275–284, New York, NY, USA, 2011. ACM.

[5] Dragan Ahmetovic, Cole Gleason, Kris M. Kitani, Hironobu Takagi, and Chieko Asakawa. Navcog: turn-by-turn smartphone navigation assistant for people with visual impairments or blindness. In *Proceedings of the 13th Web for All Conference.* ACM, 2016.

[6] Cuneyt Akinlar and Cihan Topal. Edlines: A real-time line segment detector with a false detection control. *Pattern Recognition Letters*, 32(13):1633 – 1642, 2011.

[7] V. Ralph Algazi, Carlos Avendano, and Richard O. Duda. Elevation localization and head-related transfer function analysis at low frequencies. *The Journal of the Acoustical Society of America*, 109(3):1110–1122, 2001.

[8] Abraham Arcavi. The role of visual representations in the learning of mathematics. *Educational Studies in Mathematics*, 52(3):215–241, 2003.

[9] Aries Arditi, Jeffrey D. Holtzman, and Stephen M. Kosslyn. Mental imagery and sensory experience in congenital blindness. *Neuropsychologia*, 26(1):1–12, 1988.

[10] Jeffrey P. Bigham, Anna C. Cavender, Jeremy T. Brudvik, Jacob O. Wobbrock, and Richard E. Ladner. Webinsitu: a comparative analysis of blind and sighted browsing behavior. In *Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility*, pages 51–58. ACM, 2007.

[11] Jeffrey P. Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C. Miller, Robin Miller, Aubrey Tatarowicz, Brandyn White, Samual White, and Tom Yeh.

Vizwiz: Nearly real-time answers to visual questions. In *Proceedings of the 23Nd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, pages 333–342, New York, NY, USA, 2010. ACM.

[12] Jeffrey P. Bigham, Richard E. Ladner, and Yevgen Borodin. The design of human-powered access technology. In *The Proceedings of the 13th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '11, pages 3–10, New York, NY, USA, 2011. ACM.

[13] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python.* " O'Reilly Media, Inc.", 2009.

[14] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[15] Erin L. Brady, Daisuke Sato, Chengxiong Ruan, Hironobu Takagi, and Chieko Asakawa. Exploring interface design for independent navigation by people with visual impairments. In *Proceedings of the 17th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '15, pages 387–388, New York, NY, USA, 2015. ACM.

[16] Anke Brock, Philippe Truillet, Bernard Oriola, and Christophe Jouffrais. Usage of multimodal maps for blind people: Why and how. In *ACM International Conference on Interactive Tabletops and Surfaces*, ITS '10, pages 247–248, New York, NY, USA, 2010. ACM.

[17] John Brooke. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[18] Patrick Carrington, Amy Hurst, and Shaun K. Kane. Wearables and chairables: Inclusive design of mobile input and output techniques for power wheelchair users. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 3103–3112, New York, NY, USA, 2014. ACM.

[19] Stephen H. Choi and Bruce N. Walker. Digitizer auditory graph: Making graphs accessible to the visually impaired. In *CHI '10 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '10, pages 3445–3450, New York, NY, USA, 2010. ACM.

[20] James Coughlan and Roberto Manduchi. *Camera-Based Access to Visual Information*, pages 219–246. CRC Press, 2016/09/20 2012.

[21] Mihaly Csikszentmihalyi. *Flow.* Harper Perennial Modern Classics. HarperCollins, 2009.

[22] Stanley A. Dallas Jr and Aubrey J. Erickson. Sound pattern generator, March 29 1983. US Patent 4,378,569.

[23] Sebastian Deterding, Miguel Sicart, Lennart Nacke, Kenton O'Hara, and Dan Dixon. Gamification. Using game-design elements in non-gaming contexts. In *CHI'11 Extended Abstracts on Human Factors in Computing Systems.* ACM, 2011.

[24] Moises Diaz-Cabrera, Pietro Cerri, and Paolo Medici. Robust real-time traffic light detection and distance estimation using a single camera. *Expert Syst. Appl.*, 42(8):3911–3923, May 2015.

[25] Moises Diaz-Cabrera, Pietro Cerri, and Javier Sanchez-Medina. Suspended traffic lights detection and distance estimation using color features. In *2012 15th International IEEE Conference on Intelligent Transportation Systems*, pages 1315–1320, Sept 2012.

[26] Pedro Felzenszwalb and Daniel Huttenlocher. Distance transforms of sampled functions. Technical report, Cornell University, 2004.

[27] John A. Gardner. *Access by Blind Students and Professionals to Mainstream Math and Science*, pages 502–507. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002.

[28] Andrea Gerino, Nicolò Alabastro, Cristian Bernareggi, Dragan Ahmetovic, and Sergio Mascetti. Mathmelodies: Inclusive design of a didactic game to practice mathematics. In *Computers Helping People with Special Needs: 14th International Conference, ICCHP 2014, Paris, France, July 9-11, 2014, Proceedings*, pages 564–571, Cham, 2014. Springer International Publishing.

[29] Andrea Gerino, Lorenzo Picinali, Cristian Bernareggi, Nicolò Alabastro, and Sergio Mascetti. Towards large scale evaluation of novel sonification techniques for non visual shape exploration. In *Proceedings of the 17th International ACM SIGACCESS Conference on Computers & Accessibility*, ASSETS '15, pages 13–21, New York, NY, USA, 2015. ACM.

[30] Andrea Gerino, Lorenzo Picinali, Cristian Bernareggi, and Sergio Mascetti. Eyes-free exploration of shapes with invisible puzzle. In *Proceedings of the 17th International ACM SIGACCESS Conference on Computers & Accessibility*, ASSETS '15, pages 425–426, New York, NY, USA, 2015. ACM.

[31] Nicholas A. Giudice and Gordon E. Legge. Blind navigation and the role of technology. *Engineering handbook of smart technology for aging, disability, and independence*, pages 479–500, 2008.

[32] Dorte Hammershøi and Henrik Møller. Methods for binaural recording and reproduction. *Acta Acustica united with Acustica*, 88(3):303–311, 2002.

[33] H. Rex Hartson, José C. Castillo, John Kelso, and Wayne C. Neale. Remote evaluation: The network as an extension of the usability laboratory. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '96, pages 228–235, New York, NY, USA, 1996. ACM.

[34] Thomas Hermann and Helge Ritter. Listen to your data: Model-based sonification for data analysis. *Advances in intelligent computing and multimedia systems*, 8:189–194, 1999.

[35] Amy Hurst, Scott E. Hudson, Jennifer Mankoff, and Shari Trewin. Distinguishing users by pointing performance in laboratory and real-world tasks. *ACM Transactions on Accessible Computing (TACCESS)*, 5(2):5, 2013.

[36] Huston, R. *Principles of biomechanics.* CRC press, 2008.

[37] Volodymyr Ivanchenko, James Coughlan, and Huiying Shen. Detecting and locating crosswalks using a camera phone. *Proceedings / CVPR, IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008:4563143–4563143, 2008.

[38] Volodymyr Ivanchenko, James Coughlan, and Huiying Shen. Staying in the crosswalk: A system for guiding visually impaired pedestrians at traffic intersections. *Assistive technology research series*, 25(2009):69–73, 2009.

[39] Volodymyr Ivanchenko, James Coughlan, and Huiying Shen. Real-time walk light detection with a mobile phone. *Computers Helping People with Special Needs: 12th International Conference, ICCHP 2010, Vienna, Austria, July14-16, 2010, Proceedings, Part II*, 6180:229–234, 07 2010.

[40] R. Dan Jacobson. Navigating maps with little or no sight: An audio-tactile approach. In *Proceedings of the workshop on Content Visualization and Intermedia Representations (CVIR)*, 1998.

[41] Bonnie E. John, Konstantine Prevas, Dario D. Salvucci, and Ken Koedinger. Predictive human performance modeling made easy. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 455–462, New York, NY, USA, 2004. ACM.

[42] Hernisa Kacorri, Sergio Mascetti, Andrea Gerino, Dragan Ahmetovic, Hironobu Takagi, and Chieko Asakawa. Supporting orientation of people with visual impairment: Analysis of large scale usage data. In *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '16, page to appear, New York, NY, USA, 2016. ACM.

[43] Arthur I. Karshmer and Chris Bledsoe. *Access to Mathematics by Blind Students*, pages 471–476. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002.

[44] Brian F. G. Katz and Lorenzo Picinali. *Spatial Audio Applied to Research with the Blind*, volume Advances in Sound Localization, chapter 13, pages 225–250. InTech, 2011.

[45] Brian F. G. Katz, Emmanuel Rio, Lorenzo Picinali, and Olivier Warusfel. The effect of spatialization in a data sonification exploration. *Proceedings of the International Conference of Auditory Display (ICAD08)*, June 24-27 2008, Paris, France, 2008.

[46] Steven Komarov, Katharina Reinecke, and Krzysztof Z. Gajos. Crowdsourcing performance evaluations of user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 207–216, New York, NY, USA, 2013. ACM.

[47] Benjamin Kuipers. Modeling spatial knowledge*. *Cognitive Science*, 2(2):129–153, 1978.

[48] Lefler, M., Hel-Or, H., and Hel-Or, Y. Metric plane rectification using symmetric vanishing points. In *Proc. of the 20th International Conference on Image Processing*. IEEE Comp. Soc., 2013.

[49] Harry Levitt. Adaptive testing in audiology. *Scandinavian audiology. Supplementum*, 6:241–291, 1977.

[50] James R. Lewis. Ibm computer usability satisfaction questionnaires: Psychometric evaluation and instructions for use. *Int. J. Hum.-Comput. Interact.*, 7(1):57–78, January 1995.

[51] John P. Lewis. Fast normalized cross-correlation. *International Journal on Vision interface*, 10(1):120–123, 1995.

[52] Liebowitz, D. and Zisserman, A. Metric rectification for perspective images of planes. In *Proc. of Computer Vision and Pattern Recognition*. IEEE, 1998.

[53] Shachar Maidenbaum. Sensory substitution training for users who are blind with dynamic stimuli, games and virtual environments. In *Proceedings of the 17th International ACM SIGACCESS Conference on Computers & Accessibility*, ASSETS '15, pages 355–356, New York, NY, USA, 2015. ACM.

[54] Roberto Manduchi and James Coughlan. *Portable and Mobile Systems in Assistive Technology*, pages 1078–1080. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[55] Sergio Mascetti, Dragan Ahmetovic, Andrea Gerino, and Cristian Bernareggi. Zebrarecognizer: Pedestrian crossing recognition for people with visual impairment or blindness. *Pattern Recognition*, 60:405 – 419, 2016.

[56] Sergio Mascetti, Dragan Ahmetovic, Andrea Gerino, Cristian Bernareggi, Mario Busso, and Alessandro Rizzi. Robust traffic lights detection on mobile devices for pedestrians with visual impairment. *Computer Vision and Image Understanding*, 148:123 – 135, 2016. Special issue on Assistive Computer Vision and Robotics - 'Assistive Solutions for Mobility, Communication and HMI'.

[57] Sergio Mascetti, Dragan Ahmetovic, Andrea Gerino, Cristian Bernareggi, Mario Busso, and Alessandro Rizzi. Supporting pedestrians with visual impairment during road crossing: a mobile application for traffic lights detection. In *Computers Helping People with Special Needs: 15th International Conference, ICCHP 2016, Linz, Austria, July 13-15, 2016, Proceedings*, page to appear. Springer International Publishing, 2016.

[58] Sergio Mascetti, Cristian Bernareggi, and Matteo Belotti. *TypeInBraille: quick eyes-free typing on smartphones.* Springer, 2012.

[59] Sergio Mascetti, Lorenzo Picinali, Andrea Gerino, Dragan Ahmetovic, and Cristian Bernareggi. Sonification of guidance data during road crossing for people with visual impairments or blindness. *International Journal of Human-Computer Studies*, 85:16 – 26, 2016.

[60] Peter B. L. Meijer. An experimental system for auditory image representations. *Biomedical Engineering, IEEE Transactions on*, 39(2), 1992.

[61] Brian C. J. Moore. *An introduction to the psychology of hearing.* Brill, 2012.

[62] Madoka Nakajima and Shinichiro Haruyama. New indoor navigation system for visually impaired people using visible light communication. *EURASIP Journal on Wireless Communications and Networking*, 2013(1):1–10, 2013.

[63] Mark E. J. Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[64] Liam O'Sullivan, Lorenzo Picinali, Christopher Feakes, and Douglas Cawthorne. Audio tactile maps (atm) system for the exploration of digital heritage buildings by visually-impaired individuals-first prototype and preliminary evaluation. *Forum Acousticum*, 2014.

[65] Liam O'Sullivan, Lorenzo Picinali, Andrea Gerino, and Douglas Cawthorne. A Prototype Audio-Tactile Map System with an Advanced Auditory Display. *International Journal of Mobile Human Computer Interaction*, 7(4):53–75, 2015.

[66] Don Parkes. Nomad": An audio-tactile tool for the acquisition, use and management of spatially distributed information by partially sighted and blind persons. In *Proceedings of Second International Conference on Maps and Graphics for Visually Disabled People*, pages 24–29, 1988.

[67] Romedi Passini and Guyltne Proulx. Wayfinding without vision: An experiment with congenitally totally blind people. *Environment and Behavior*, 20(2):227–252, 1988.

[68] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[69] Helen Petrie, Fraser Hamilton, Neil King, and Pete Pavan. Remote usability evaluations with disabled people. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 1133–1141, New York, NY, USA, 2006. ACM.

[70] Helen Petrie, Valerie Johnson, Thomas Strothotte, Andreas Raab, Steffi Fritz, and Rainer Michel. Mobic: Designing a travel aid for blind and elderly people. *The Journal of Navigation*, 49:45–52, 1 1996.

[71] Luz Rello, Kristin Williams, Abdullah Ali, Nancy Cushen White, and Jeffrey P. Bigham. Dytective: Towards detecting dyslexia across languages using an online game. In *Proceedings of the 13th Web for All Conference*, W4A '16, pages 29:1–29:4, New York, NY, USA, 2016. ACM.

[72] Daniele Riboni, Claudio Bettini, Gabriele Civitarese, Zaffar Haider Janjua, and Rim Helaoui. Smartfaber: Recognizing fine-grained abnormal behaviors for early detection of mild cognitive impairment. *Artificial Intelligence in Medicine*, 2016.

[73] Jan Roters, Xiaoyi Jiang, and Kai Rothaus. Recognition of traffic lights in live video streams on mobile devices. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(10):1497–1511, Oct 2011.

[74] Pablo Revuelta Sanz, Belén Ruiz Mezcua, José M. Sánchez Pena, and Bruce N. Walker. Scenes and images into sounds: A taxonomy of image sonification methods for mobility applications. *J. Audio Eng. Soc*, 62(3):161–171, 2014.

[75] Rajib Sarkar, Sambit Bakshi, and Pankaj K. Sa. Review on image sonification: A non-visual scene representation. In *Recent Advances in Information Technology (RAIT), 2012 1st International Conference on*, pages 86–90, March 2012.

[76] Jochen Schneider and Thomas Strothotte. Constructive exploration of spatial information by blind users. In *Proceedings of the Fourth International ACM Conference on Assistive Technologies*, Assets '00, pages 188–192, New York, NY, USA, 2000. ACM.

[77] Stephen Se. Zebra-crossing detection for the partially sighted. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 211–217 vol.2, 2000.

[78] Andrew Sears and Vicki Hanson. Representing users in accessibility research. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 2235–2238, New York, NY, USA, 2011. ACM.

[79] Ella Striem-Amit, Miriam Guendelman, and Amir Amedi. 'visual' acuity of the congenitally blind using visual-to-auditory sensory substitution. *PLoS ONE*, 7(3):1–6, 03 2012.

[80] Satoshi Suzuki and Keiichi Abe. Topological structural analysis of digitized binary images by border following. *International Journal on Computer Vision, Graphics, and Image Processing*, 30(1):32–46, 1985.

[81] Marzia Taibbi, Cristian Bernareggi, Andrea Gerino, Dragan Ahmetovic, and Sergio Mascetti. Audiofunctions: Eyes-free exploration of mathematical functions on tablets. In *Computers Helping People with Special Needs: 14th International Conference, ICCHP 2014, Paris, France, July 9-11, 2014, Proceedings*, pages 537–544, Cham, 2014. Springer International Publishing.

[82] Andrew F. Tatham. The design of tactile maps: theoretical and practical considerations. *Proceedings of international cartographic association: mapping the nations*, pages 157–166, 1991.

[83] Technical Committee CEN/TC 226 "Road equipment". *European Standard EN 12368:2006 on "traffic control equipment - signal head"*, 2006.

[84] Evelyn Kubiak Thomas Dick. Issues and aids for teaching mathematics to the blind. *The Mathematics Teacher*, 90(5):344–349, 1997.

[85] Mohammad S. Uddin and Tadayoshi Shioyama. Measurement of the length of pedestrian crossings - a navigational aid for blind people. In *Intelligent Transportation Systems, 2004. Proceedings. The 7th International IEEE Conference on*, pages 690–695, Oct 2004.

[86] Von Gioi, Rafael G., Jakubowicz, Jeremie, Morel, Jean-Michel, and Randall, Gregory. Lsd: A fast line segment detector with a false detection control. *Trans. on Pattern Analysis and Machine Intelligence*, 2010.

[87] Bruce N. Walker and Joshua T. Cothran. Sonification Sandbox: A graphical toolkit for auditory graphs. *9th International Conference on Auditory Display*, 2003.

[88] Gang Wang, Xinyi Zhang, Shiliang Tang, Haitao Zheng, and Ben Y. Zhao. Unsupervised clickstream clustering for user behavior analysis. In *SIGCHI Conference on Human Factors in Computing Systems*, 2016.

[89] Elizabeth M. Wenzel, Marianne Arruda, Doris J. Kistler, and Frederic L. Wightman. Localization using nonindividualized head-related transfer functions. *The Journal of the Acoustical Society of America*, 94(1):111–123, 1993.

[90] Michele A. Williams, Amy Hurst, and Shaun K. Kane. Pray before you step out: describing personal and situational blind navigation behaviors. In *Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility*, page 28. ACM, 2013.

[91] Matthew Wright. Open sound control: an enabling technology for musical networking. *Organised Sound*, 10:193–200, 12 2005.

[92] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *ICML*, volume 97, pages 412–420, 1997.

[93] Woon Seung Yeo and Jonathan Berger. A framework for designing image sonification methods. In *Proceedings of the 11th International Conference on Auditory Display (ICAD 2005)*, pages 323–327. Georgia Institute of Technology, 2005.

[94] Tsubasa Yoshida, Kris M. Kitani, Hideki Koike, Serge Belongie, and Kevin Schlei. Edgesonic: Image feature sonification for the visually impaired. In *Proceedings of the 2Nd Augmented Human International Conference*, AH '11, pages 11:1–11:4, New York, NY, USA, 2011. ACM.

[95] Eberhard Zwicker. Subdivision of the audible frequency range into critical bands (frequenzgruppen). *The Journal of the Acoustical Society of America*, 2(33), 1961.

[96] Kathryn Zyskowski, Meredith Ringel Morris, Jeffrey P. Bigham, Mary L. Gray, and Shaun K. Kane. Accessible crowdwork?: Understanding the value in and challenge of microtask employment for people with disabilities. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, CSCW '15, pages 1682–1693, New York, NY, USA, 2015. ACM.