Extending Digital Whiteboards with Mobile Devices using Focus-Aware Visual Cues

Katrin Freihofner

MASTERARBEIT

eingereicht am
Fachhochschul-Masterstudiengang

Interactive Media

in Hagenberg

im Juni 2014

Declaration

I hereby declare and confirm that this thesis is entirely the result of my own original work. Where other sources of information have been used, they have been indicated as such and properly acknowledged. I further declare that this or similar work has not been submitted for credit elsewhere.

Hagenberg, June 30, 2014

Katrin Freihofner

Contents

Kurzfassung

Digitale Whiteboards bieten eine große Arbeitsfläche und unterstützen damit die Zusammenarbeit mehrerer Benutzer. Im praktischen Einsatz konnten jedoch verschiedene Schwachpunkte festgestellt werden. Die Projektoren bieten eine geringe Auflösung. Außerdem erfordern Zoomen und Navigation Übung. Der Vorteil, dass jede Information sofort für alle sichtbar ist, geht einher mit der Gefahr, dass sich Benutzer unwohl fühlen während sie vor Publikum arbeiten. Die Erweiterung dieses Setups mit zusätzlichen mobilen Endgeräten, wie zum Beispiel Tablet-Computer, bringt die Vorteile privater Interaktion mittels Touch-Gesten und die hohe Auflösung dieses Gerätes ermöglicht eine smarte, schnelle und präzise Interaktion. Andererseits führt die Kombination von Tablets mit digitalen Whiteboards zu ständigen Blickwechseln zwischen den Geräten. Benutzer suchen nach auffälligen Punkten um sich auf dem großen Display zu orientieren.

Wir präsentieren einen Lösungsansatz um Benutzer in diesem Orientierungsprozess auf großen Displays zu unterstützen. Dafür ermitteln wir den Benutzerfokus und zeigen die aktuelle Position des Tablet-Inhalts am Whiteboard an. Ist der am Tablet-Computer dargestellte Inhalt außerhalb des Whiteboard-Displays, werden visuelle Indikatoren am Rand des Whiteboards abgebildet. Diese zeigen die Distanz und die Richtung des Tablet Inhalts über ihre Position und die Deckkraft an. Zusätzlich zu einer gerenderten Lösung stellen wir eine Darstellung über Umgebungslicht vor. Diese Lösungsansätze haben wir mittels eines explorativen Experten-Interviews evaluiert. Dieses Interview umfasste auch den Vergleich der beiden Ansätze, Rendering und Umgebungslicht. Die Ergebnisse für den gerenderten Lösungsansatz sind, verglichen mit dem Umgebungslichtkonzept, generell besser.

Abstract

Digital whiteboards offer a large screen space and therefore support multi-user collaboration. While working on this large display several drawbacks could be detected. The projectors have a low resolution. Besides, zooming and navigation need some practice. The benefit of public presentation of information accompanies with the drawback of users feeling awkward while working in front of an audience. Extending the setup with additional mobile devices, such as tablet computers, adds several benefits. Private interaction via touch gestures and the high-resolution of this device enable a smart, fast and precise interaction. On the other hand, the combination of tablet computers with digital whiteboards causes a lot of gaze shifts between these devices. When the same data is presented in different ways, users search for prominent points to use them as orientation points.

In this thesis we present an approach to support users' orientation on whiteboards, by detecting their focus and showing the position of the tablet's current view at the whiteboard screen. When the tablet's view is outside the currently shown view on the whiteboard display, visual cues appear as borderlines on the edges of the whiteboard. These lines indicate the distance and direction of the tablet's view via opacity and position. Additionally to a rendered solution, we investigated in an ambient light approach, which indicates the distances of the tablet's view through the lights color. We evaluated our solution by an explorative interview with two experts. This also included the comparison of rendered- and ambient light visual cues. The results show, overall the rendered approach performs better, compared to ambient light.

Chapter 1

Introduction

With the introduction of digital whiteboards, collaboration is getting enriched and the direct interaction-way of this device simplifies work in many ways (Figure 1.1). Especially the work with extensive datasets, as for example map interaction, benefits from the large screen space. A digital whiteboard can display all this data and therefore provide an overview, observable for everybody.

Figure 1.1: A digital whiteboard supports the simultaneous work of multiple users. Moreover, it provides an overview of large datasets, observable by everybody.

At the same time, mobile devices are becoming increasingly ubiquitous and are thus widely available. They are well integrated in our daily life and support us constantly. We are used to the interaction with our mobile devices and know their features well. Although their great opportunities, mobile devices are mostly used for private interactions and to substitute features like checking emails.

With benefits like direct manipulation via touch gestures, mobile devices have a great potential as an additional interaction space and to support users in interacting with digital whiteboards. Compared to mobile phones, tablet computers are especially suitable for this setup because of their form factor. They provide enough screen space for working in a comfortable way.

1.1 Distributed User Interfaces and Cross-Device Interaction

By adding an additional tablet computer to the digital whiteboard to further enhance this setup, Distributed User Interfaces (DUIs) and the interaction of those devices become a concern. First, we want to refer to the definition of DUIs by Elmqvist [9, page 2]:

> "A distributed user interface is a user interface whose components are distributed across one or more of the dimensions input, output, platform, space, and time."

An issue in DUIs is the management of human-computer-interaction. Input and output as well as data can be redirected to the different devices [9, page 1]. Furthermore, Rashid et al. [30, 31] detected, the major known drawback of this setup is constant gaze shifts between the devices. The results of their study show, a gap between the displays as it appears on a whiteboard plus mobile device-setup slows down the movement of visual objects across these displays. Additionally, they found out that coordinated visuals could cause attention switches. These attention switches are related to performance overhead. Tested within this study were text, image and map search tasks. While working with maps, searching for eye-catching points and reorientate oneself on the other display are issues in particular (Figure 1.2).

Therefore, we will address these concerns by visual cues that support users in cross-device interaction. The orientation support we are presenting in this thesis needs to attract attention, while on the other hand, be suitable for collaboration and thus, is not distracting co-workers who are currently working on the whiteboard.

Another goal for our solution is to provide a system, which automatically responses to user's attention, or more particularly, to a lack of attention. Therefore, visual cues that indicate the tablet view's position, are shown on

Figure 1.2: This sketch illustrates the problem with DUIs. Constant gaze shifts between digital whiteboard and tablet computer are the result of this setup, as described by Rashid et al. [30, 31].

the whiteboard display as soon as a user's focus is no longer on the tablet's screen.

Focus indicates interest [42, 43] and addressing attention before requesting is a common behavior [23], as it is natural to focus who we speak to or who we expect answer from. This approach is also shown in attentive user interfaces, for example by Smith [36]. An example for a commercial product that is sensitive to focus is Samsung's Galaxy S4[1] with its *smart pause* and *smart stay* functions. It automatically pauses videos when the focus is not on the phone anymore. *Smart stay* can prevent the mobile phones display from going to standby mode as long somebody is watching.

Interestingly, tracking head positions can reveal a lot about current focus. Stiefelhagen et al. [39] presented in their work, that head position is a great indicator for their focus. In 74% of their studied meeting scenarios, the focus could be predicted. Accordingly, our approach is to detect user-focus via face-detection and the built-in front camera of the tablet device (Figure 1.3).

[1] http://www.samsung.com/at/promotions/galaxys4/

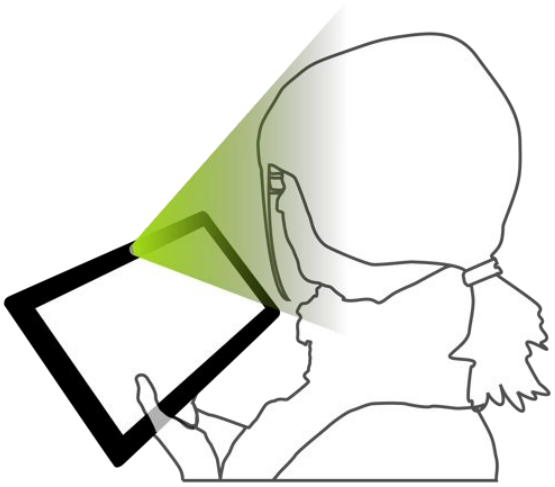


Figure 1.3: The focus-detection is done via the tablet's built-in front camera and a face-detection algorithm.

1.2 Supporting Interaction by using Visual Cues

We implemented different forms of visual cues that indicate the current position of the tablet's view on the digital whiteboard for each user. These visual representations can be displayed on demand, by button or by focus-detection. We present two options:

- the visual cues for tablet views that are currently visible on the whiteboard display—referred to as *on-screen visual cues* (Figure 1.4),
- visual cues for tablet views that are outside of the whiteboard display—referred to as *off-screen visual cues*. These off-screen visual cues can further be classified in:
  - borderlines, rendered on the whiteboard—referred to as *off-screen rendering* (Figure 1.5), and
  - visual cues by ambient light—referred to as *off-screen ambient light* (Figure 1.6).

The on-screen visual cues are additionally supported by motion to attract attention, implemented as a magnification animation. This is caused by visual attention, which is concentrated around the fovea and therefore dependent on our current focus. The field of view where information can be processed fast varies from task to task [44]. When it comes to objects in motion, this field of view can increase largely [27]. Accordingly, animated objects can be detected easily, also when they are not in the current field of view. When off-screen visual cues are appearing on the edges around the digital whiteboard, the same effect attracts attention.

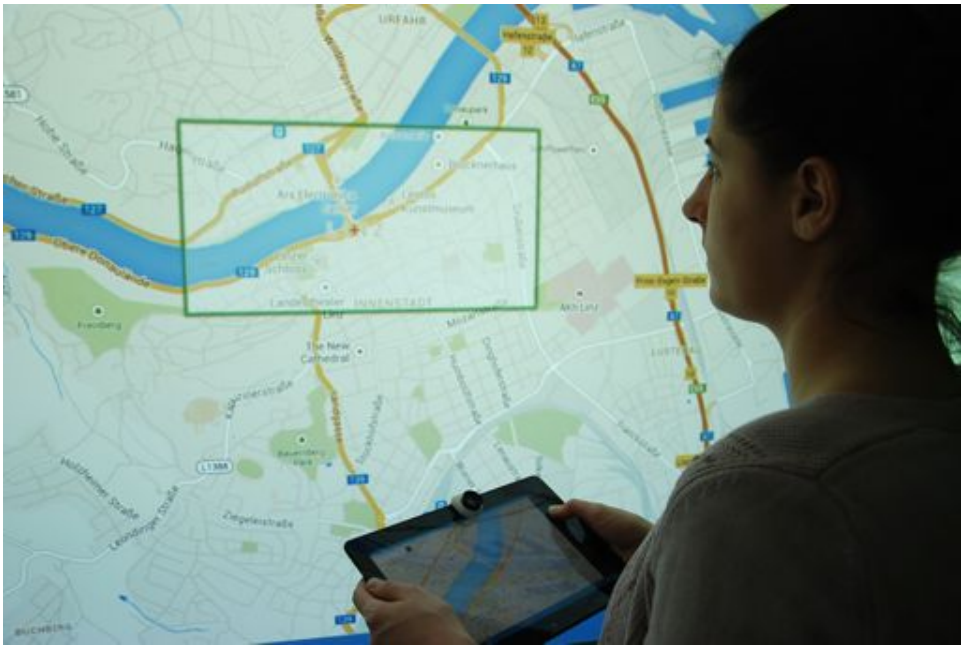


Figure 1.4: An example for an on-screen representation of the tablet's view.

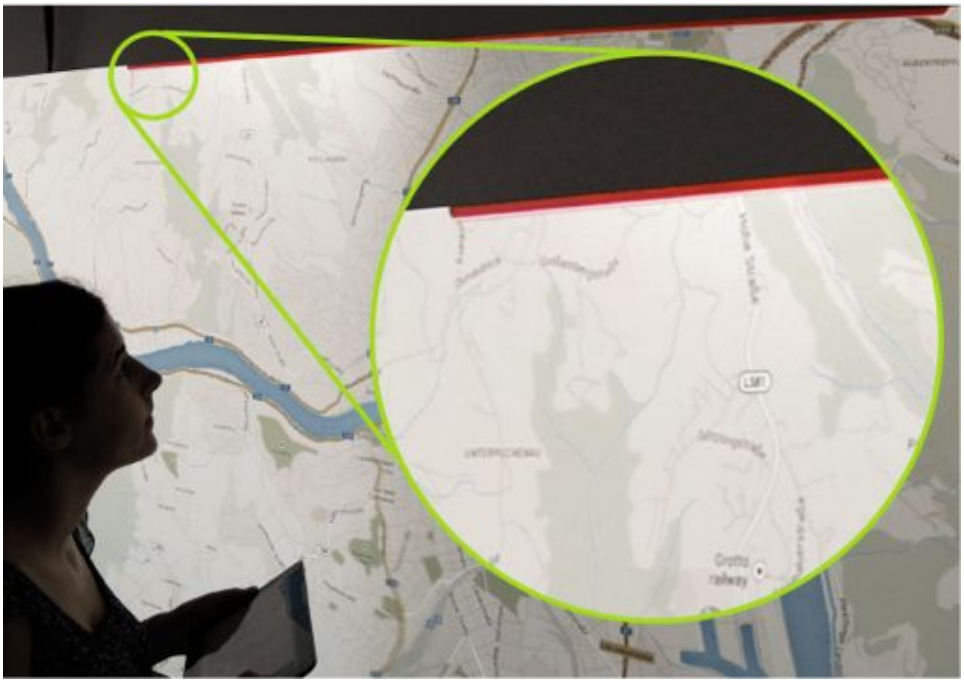


Figure 1.5: An example for an off-screen rendering. The borderline appears on the top-edge of the whiteboard's display, to indicate the direction of the tablet's view. The opacity of the line shows the distance to the view.

The first intent was to render borderlines directly on the whiteboard display. These lines should indicate distance and direction of the tablet's view. After several design iterations the borderlines appear as shown in figure 1.5.

In our second approach of off-screen visual cues, we present an implementation via ambient light (Figure 1.6). This solution adds the benefit that it does not need any screen-space at all. Besides, ambient light is known to be less distracting [1]. We assume that co-workers, who are directly working on the whiteboard get less distracted by the ambient light, compared to the rendering of borderlines directly on the whiteboard. To support multi-user capability, all types of visual cues appear for each user in an individual color.

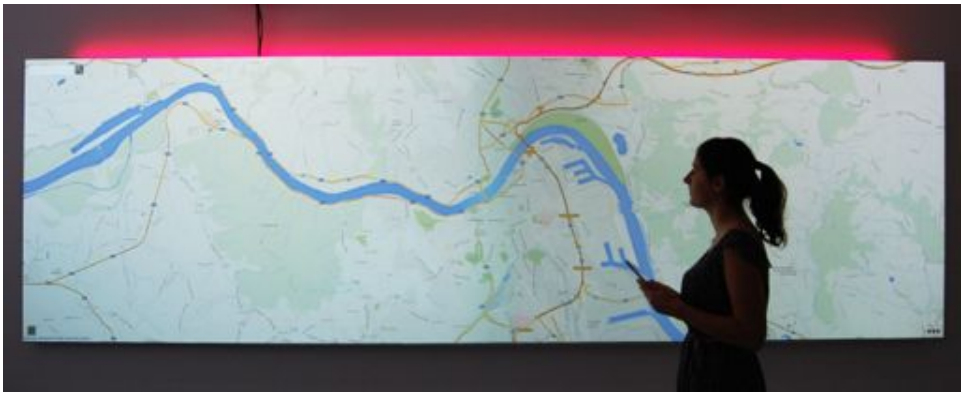


Figure 1.6: An example for an off-screen visual cue implemented with ambient light.

1.3 Application Scope

We show the extension of digital whiteboards with mobile devices, which brings additional interaction benefits, like a fast and easy way of direct manipulation. The system we present in this thesis is suitable for a large variety of applications. While a whiteboard is suitable for displaying an overview, each mobile device provides a detailed view and represents the individual focus (Figure 1.7). Especially for map applications, *overview + detail* is a common and effective visualization technique [4, 15, 28, 33]. The solution we are presenting supports the combination of *overview + detail*, distributed on multiple devices. As *overview + detail* is especially interesting in map applications, consequently this is the main application scope for our solution. Map views can be zoomed individually on each device and are able to show different types of data. More specifically, different types of data are diverse map views (hybrid and satellite) in our scenario.

Figure 1.7: The basic concept supports collaboration as well as individual work on the large surface.

1.4 Contribution

This section covers a listing of the contribution we are making to this research area. In this thesis, we present a focus-aware technique for supporting users' orientation on DUIs. The implemented solution supports:

- on-screen representations, and
- off-screen representations of the tablet view,
  - in the form of rendered visual cues, and
  - via ambient light.

Furthermore, these visual cues can be triggered by button click, but our approach also works:

- focus-aware,
- calibration-free, and
- without dwell time.

While related work is mainly focusing on one small device, our presented solution is designed for the combination of large digital whiteboards with multiple tablet computers, and therefore, supports multi-user collaboration. Additionally, we have investigated in an explorative evaluation and a comparison of the implemented solution. This evaluation was done in the form of an interview. The results give insights in benefits and drawbacks of this system.

1.5 Outline

The structure of the thesis at hand is as follows: after this introduction, Chapter 2 discusses previous and related work, inspirations, existing projects with gaze interaction and ambient light. Based on these findings, the third chapter shows the application design. First, the setup and the approach of an extension with a tablet computer is presented. That chapter also includes the whole design process and the final on- and off-screen visual cues. The implementation is explained in detail in Chapter 4. That section covers the workflow, calculations and implementation details for the different types of visual cues, ambient light and multi-user handling. Finally, face tracking—the basis for the automatic focus-detection via the tablet's built-in camera—and the underlying framework are discussed briefly. In the Discussion, design and results of an explorative interview about the comparison of the off-screen rendering and the off-screen borderlines via ambient light are described. Additionally, the limitations of our solution are described. The last chapter includes the conclusion and possibilities for future directions of the presented work.

Chapter 2

Related Work

In this chapter we discuss publications related to the thesis at hand. This knowledge is the basis for our solution. There exist numerous interesting publications in this research area, however, only the most important will be discussed briefly. These are structured in: similar setups, comparable visual cues, gaze and ambient light as interaction techniques.

2.1 Similar Setups

Several projects use a mobile device as private interaction space in combination with large public displays. A recent study concerning distributed user interfaces for emergency planning [8] shows deep insights in this research area. The presented system supports collaboration with a large, multi-touch, shared display and multiple private interaction spaces in the form of mobile devices. This solution is suggested, as it reduces information overload and improves individual information processing. The shared display showed multiple interaction points placed on a map, which were related to different topics. Each interaction point had a QR-Code, which enabled exploring information about the related topic via the individual mobile device (Figure 2.1). This is especially interesting, because of the similarity to our application scope.

Others, like Myers et al. [25], pointed out different applications for a similar setup. Examples are the *slideshow commander* for controlling presentations remotely, a remote clipboard and shortcuts for the extension of a workstation with an additional Personal Digital Assistant (PDA). A similar publication, by Greenberg et al. [13], presented a system to create and share notes. Multiple users worked with a PDA, where they could create notes and publish all or parts of them on a large public wall display. This system is mainly designed for collaborative situations, but benefits from the possibility of a private interaction space on the PDA. Rekimoto [32] showed in his work how a PDA can be used as a tool palette for a wall display. As

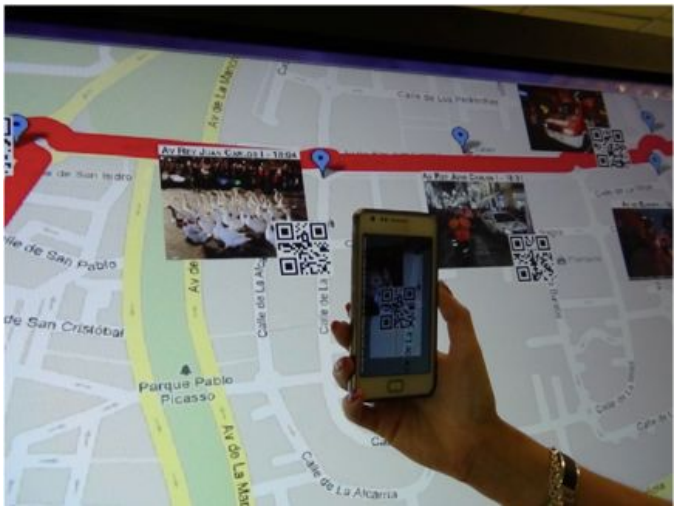


Figure 2.1: The setup in [8] consisted of a shared display and multiple mobile devices. Additional information to the interaction points can be gained by scanning QR-Codes.

the large surface of a whiteboard display makes traditional interfaces with menu bars ineffective, the PDAs could be used to display interaction possibilities. Additionally, private information could be hidden on the PDA and interfaces could be personalized.

Besides the mentioned project, several publications [34, 45] show the benefit of high-resolution mobile devices, displaying a detailed focused view, while a large display presents the context. This is especially interesting for map applications. Sanneblad and Holmquist [34] used a tablet computer as a magic lens for details and a low-resolution wall display for the context view. The detailed view is displayed on the tablet computer, as soon as the device is placed in front of the current area of interest on the wall display. The views got automatically aligned to each other. However, this approach makes private interaction impossible. Sanneblad and Holmquist [34] reported that participants of a preliminary user study had concerns about the fragility of the setup. They feared to drop the device, which indicates that this is not a convenient way of interacting with this setup.

Similary, Olwal [26] used mobile phones for alternative views of large surfaces, like a magic lens. Pointing directly on the surface controlled the content, displayed on the mobile phone. In the other direction, the large surface's zoom level could be controlled by the mobile phone. Furthermore, annotations in the form of text input and free-hand annotations could be added via mobile phones. Emphasized was the fact that these mobile phones provided tactile input opportunities. Multiple users could use the setup simultaneously. Users were separated by color, equally to the work at hand.

In contrast, Weigel et al. [45] presented a combination of low-resolution wall displays with handheld projectors as mobile devices. The view of these handheld projectors could change in size and displayed both, focus or context of the current area of interest. According to the actual requirements, users could control the transition between focus and context via the position of the projector. As users (and therefore projectors) moved closer to the wall, more detail was presented in higher resolution and a smaller projection. The transition between focus and context worked continuously and was suitable for collaboration of multiple users. Although the presented work used mobile devices, the benefit of a private interaction space was not supported.

Compared to the mentioned publications, our approach can be used to display different views on the tablet screen and on the whiteboard display. Particularly, this means different types of maps as well as overviews and detailed views.

As already discussed in Chapter 1, although the combination of a digital whiteboard and multiple mobile devices seams a suitable approach, this combination will bring some drawbacks. Constant gaze shifts between these devices [30, 31], the visual search for orientation points and the reorientation, accompany with this configuration. Therefore, this area needs further research.

2.2 Visual Attention and Perception

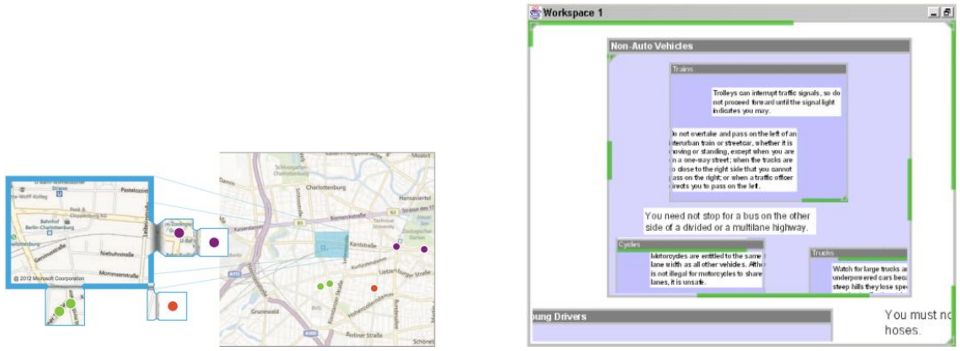
In the area of visual attention and perception, several interesting investigations have been done so far. As this is an important part of this work, the following section will consider visual cues for on- and off-screen objects, with a focus on the presented setup. While showing off-screen objects means perceiving elements on the whiteboard, which are not visible in the actual view, on-screen visual cues are more about guiding attention to a specific screen area.

2.2.1 Attention on On-Screen Elements

Although, research has been done to attract attention on large displays, especially for advertisement, only little research so far has a focus on guiding attention on whiteboards for orientation purpose. One approach is the use of a spotlight on the projection [21]. Additionally, to exposing an area with a spotlight, the surrounding gets darkened. This method could guide user's attention successfully. However it makes independent work of multiple users on one whiteboard very hard, as this visualization technique is rather distracting for co-workers.

Ion et al. [17, 18] presented a method to display "off-view" moving objects in an *overview + detail*-setup for maps. Depending on these dynamic objects, the detailed views automatically adapted to keep them visible, also outside

Figure 2.2: Spotlight [21]: attracting attention on on-screen elements, by a spotlight and darkening of the surrounding area.



(a) (b)

Figure 2.3: Canyon [17, 18]: a technique for uncovering off-view elements (a). City Lights [46] shows off-screen elements in the form of borders. The direction and size of these off-screen objects is presented by the length and position of these borders (b).

the actual view. A small additional view was attached in the direction the object was moving. Distance was indicated by a virtual fold (Figure 2.3 (a)). However, this technique needed a lot of screen space and does not meet our requirements, like for example the possibility of private interaction.

2.2.2 Perception of Off-Screen Elements

Different ways of making off-screen objects visible on-screen are already published [3, 14, 16, 46]. Most of them deal successfully with the problem of devices with a small screen [3, 14, 16]. Existing research is comparing these techniques for mobile devices [6, 12]. All these methods have in common that they are designed to uncover multiple off-screen objects in a certain area. In contrast, the work at hand requires displaying a few elements (dependent

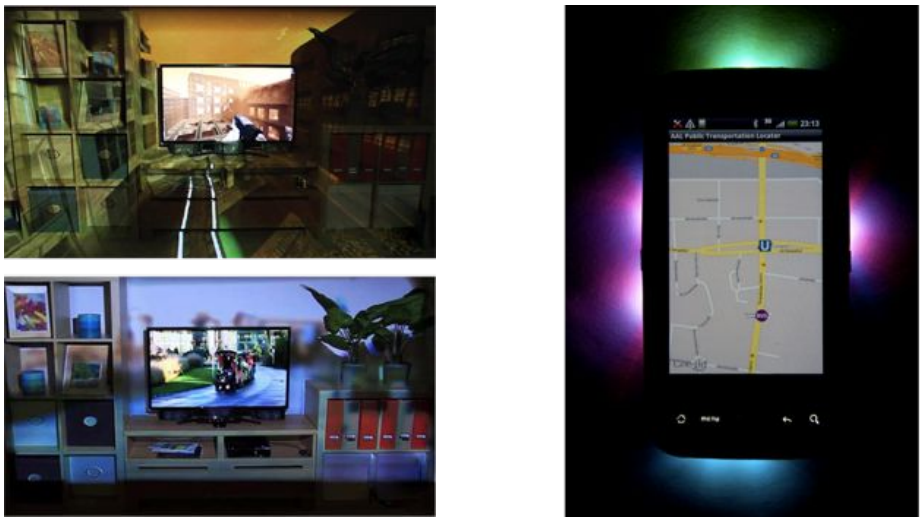
on the number of users), on a large display. Furthermore, we expect views at different distances, which would be difficult to show with the mentioned techniques.

An example for displaying off-screen elements on desktop workstations is City Lights. City Lights [46] is a fisheye visualization technique. It presented contextual views in the form of borders along the window to show off-screen objects (Figure 2.3 (b)). The length and position of the borders indicated the direction and size of these off-screen objects. These borders were also used for navigation. By selecting a border, the view got automatically shifted to the closest object indicated by that border element. This technique is quite similar to the visual cues we present in the thesis at hand. Nonetheless, City Lights is designed for single user and single display interaction.

Other interesting approaches in this context exist, for example, Frisch and Dachselt [10] showed a technique for uncovering off-screen elements of class-diagrams. In this project, border elements showed the direction, number and type of off-screen objects. However, this technique is only partly suitable for our use case, as there is no way of indicating size and distance. Context aware graph navigation that can also be used for map applications has been presented by Ghani et al. [11]. Off-screen nodes and their surrounding area were shifted to the borders of the screen. This is a very space-efficient rendering technique. Which nodes are shown was, amongst others, dependent on the distance of the nodes to the edge of the screen.

2.3 Ambient Light

Jones et al. [20] presented, IllumiRoom, an ambient light solution for augmenting the area around the screen with a projection. Their solution was mainly designed for gaming. The projection extended the view and let users immerge in a new gaming experience (Figure 2.4 (a)). An example for a commercial product solution is Philips' Ambilight television. The light around the television's edges matches and adapts according to the currently shown content. While these solutions are designed for large monitors, also mobile phones can be extended with ambient light (Figure 2.4 (b)). One of the demonstrated use cases [24, 29] is to visualize off-screen objects of maps, similar to the approach we present in this thesis. Equal to our solution, distance was indicated by light intensity [29]. However, one light represents one off-screen element. A similar approach, but for dynamic objects, will be presented [24] in the near future. During two studies with prototypes, they analyzed user interaction with ambient light, added to a tablet computer. Amongst others, the idea of mapping distance to the light's brightness was tested. Unfortunately, they found no clear preference. Qualitative results show, ambient light does not distract at all and benefits from its intuitiveness.



(a) (b)

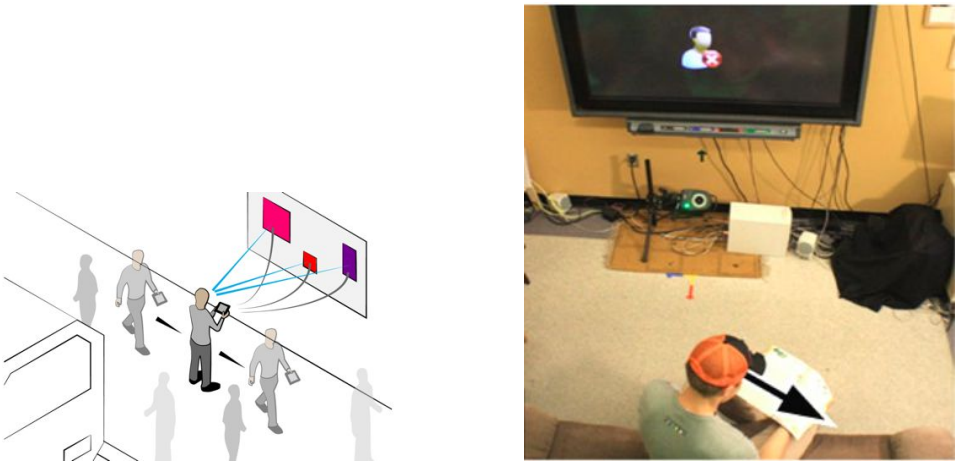
Figure 2.4: Two versions of the IllumiRoom [20] (a). Ambient Light for mobile phones [29] (b).

2.4 Gaze Supported Interaction

The positive effects of using gaze to support natural user interaction are proved by multiple publications [7, 22, 37, 38]. However, the problem of unintentionally triggering an action (also known as the Midas touch problem [19]) as well as long dwell times are known issues in gaze interaction.

Certain projects used attention in the form of eye gaze to decide which device a user wants to communicate with. Gaze could activate appliances and interaction was done by speech [35]. Additionally, devices could react on a lack of user attention as it is implemented in the work at hand. Similarly, Ballendat et al. [2] and Figure 2.5 (b), presented a system that consisted of several devices, which are reacting to position, identity, movement and orientation. In one of the presented use cases, a television is showing a video. As the attention of the person in front of the television switches to another device or person, the television reacts with pausing the video. In contrast, Dickie et al. suggested a gaze-based, single-user technique for switching between multiple desktop computer setups [7]. The results of an experiment show that the gaze-based technique performs best compared to switching by mouse, function keys and working with multiple keyboards.

There is extensive research about using gaze interaction for selection tasks on distant displays. Examples are, Turner et al. [40, 41] and Figure 2.5 (a), who introduced multiple interaction techniques, combining eye-gaze and touch gestures on a mobile device. By combining gaze interaction with



(a) (b)

Figure 2.5: Interaction via gaze [40] (a). The video gets paused when a user's attention is shifted to something else [2] (b).

additional touch gestures, no dwell time is needed and the Midas touch problem can be minimized. Besides, Stellmach and Dachselt [37, 38], extended the combination of gaze and touch gestures with for example, an additional magnification lens. Others [22], combined eye-gaze with keyboard input.

Compared to the mentioned research, our approach uses a lack of attention to trigger an action. Second, as the focus-detection must be enabled via a button, unintentional activation is not possible. Furthermore, there is no dwell time needed in our solution.

Chapter 3

Application Design

In the following chapter, the design decisions concerning the implemented solution are presented. Therefore, on- and off-screen visual cues are described in detail. These visual cues are classified in digital renderings on the whiteboard and visual cues via ambient light around the whiteboard. Furthermore, the used setup is introduced and general requirements for this system are discussed.

3.1 System

The solution, we are presenting supports multiple users who can either work directly on the digital whiteboard or with additional personal devices, such as tablet computers (Figure 3.1).

Figure 3.1: The basic setup consists of a digital whiteboard and gets extended with an additional tablet computer for every user.

3.1.1 Digital Whiteboard

The digital whiteboard used for this work is a 4 × 1.25 m projection. Interaction with the device is handled via pen and keyboard. Pen interaction enables direct manipulation and fast interaction. The large surface of the device has several benefits. It can act as an overview display and supports multiple users working in parallel. On the other hand, the projectors have a low resolution and the benefit of public presentation of information accompanies with the drawback of users feeling awkward while working in front of an audience. Zooming and navigation play a key role in map interaction. These features can be controlled via pen input. Compared to well-established multi-touch interaction, this interaction technique needs some practice.

3.1.2 Tablet Computer

Extending the whiteboard setup with additional tablet computers adds a lot of benefits. First of all, the device is mobile, there is no need to stand and work directly on the whiteboard. This means, it is easier to interact with the system while keeping an overview from a distance, without constantly moving to and away from the whiteboard. Large user groups can distribute their work on several devices and are no longer standing in the way and blocking co-workers. The benefits of the interaction via touch gestures and the high-resolution of this device, enable a smart, fast and precise interaction. Furthermore, the built-in front camera of the device makes focus-detection possible. An additional tablet computer adds a private interaction space, where information can be hidden from others. Moreover, it enables authentication and personalization.

3.2 Uncovering the Tablet's View

The approach of adding a tablet computer to this setup solves several of the mentioned drawbacks of a whiteboard-only system. In the following we discuss the requirements for the proposed setup.

3.2.1 Design Goals

We designed this system to support the extension of digital whiteboards with tablet computers. It enhances, finding the spatial orientation of the tablet view's position on the large whiteboard display. Most importantly, visual cues need to be detected very fast and attract the user's attention. On the other hand, the visual cues should be as little distracting as possible, for co-workers that are currently working on the whiteboard.

A more general requirement for this system is an easy and intuitive interaction. This system should be self-explanatory and therefore useable without a need for any practice.

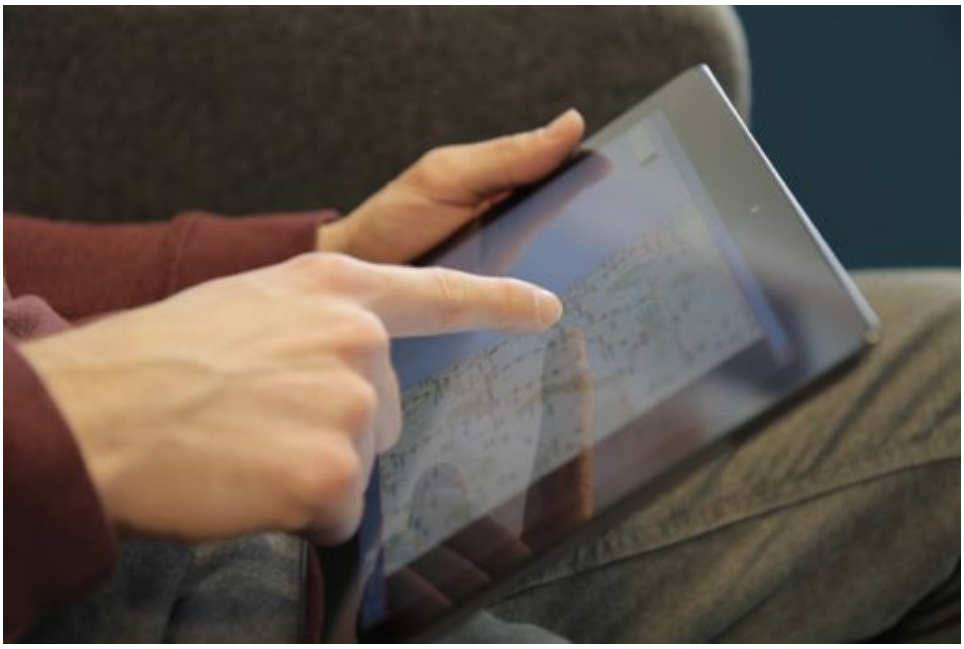


Figure 3.2: An additional tablet computer enables a private interaction space, personalization as well as authentication. Another benefit is the possibility of direct manipulation via multi-touch interaction.

Visual Cues

In a typical use case, the whiteboard shows an overview map of an area, while the tablet devices show different detailed views. These detailed views can be part of the whiteboard's currently shown content or not. The implemented setup supports map views on every level of detail.

The system needs to react immediately and the visual cues should appear on demand. Therefore, the goal for focus-aware orientation via visual cues is to work without dwell time. Unintended actions must not be triggered. Enabling and disabling the focus-detection on demand can counteract this. Both, on- and off-screen visual cues need to be straightforward to interpret and easy to spot. It is crucial for a tablet view's representation to be easy to detect at all zoom levels. Furthermore, off-screen visual cues need to provide a spatial hint about distance and direction of the tablet's view position.

Private Interaction and Collaboration

Adding a tablet computer to the whiteboard setup enables a private interaction space (Figure 3.2). Information is no longer immediately visible for everybody. As the visual cue on the whiteboard is only shown on demand, users can hide their current view on their personal device.

On the other hand, the system should be able to share and present the tablet's view of a single user for collaboration on the whiteboard display. The possibility to synchronize the current personal view with the whiteboard's content is provided via a button on the tablet application's interface.

Direct Manipulation and Automatic Response

On the digital whiteboard, direct manipulation is possible via pen input. This benefit can be used on the tablets via touch interaction (Figure 3.2). Navigation is done by dragging.

The system is designed to support focus-aware visual cues. This focus-detection is working without any calibration. Depending on a user's current focus, the on/off-screen visual cue is shown. Once enabled, the application automatically detects whether the current focus is on the tablet device or not. If the application cannot detect a person, it assumes that the focus is not on the tablet anymore and shows the appropriate visual cue.

Supporting Multiple Users

The visual cues enable assuming where co-workers are currently working. These representations of the work area can be compared and overlapping views can be detected.

The described setup supports individual work as well as asynchronous collaboration. The personal tablet device serves as individual workspace. It is completely independent and zooming and navigation does not affect the whiteboard application. Detailed views and overview can be displayed on the personal device on demand without distracting co-workers. Different types of map views (hybrid and satellite view) are adjustable for each device.

3.2.2 Map Application

The map application designed for the whiteboard has only a few interaction possibilities. Besides navigation via panning and zooming, locations can be searched on the map and the map view's type can be changed. Initially, the start location is set to "Linz, Austria" and an appropriate zoom level for getting an overview of the city. The application designed for the tablet devices has also "Linz" as start location, but shows a more detailed view. The zoom level can vary and its minimum and maximum values can be configured for each map. A settings button is placed in the right bottom corner of the whiteboard's map application. In the settings dialog the maximum distance for the off-screen visual cues can be entered in kilometers. Furthermore, users' colors can be changed and the ambient light can be turned on and off.

The map application we implemented for the tablet devices (Figure 3.3) has a button bar with five buttons on the right border. As the application is designed to be used in landscape mode, the button bar is easy to reach for

Figure 3.3: The tablet application. On the right border of the display is the button bar with the *Show View*- (4), the *Sync*- (3), the *Focus Detection*- (2) and the *Zoom*-buttons (1). In the top and bottom corners on the left, are a search box (5) and a button to change the map view's type from hybrid to satellite (6).

both, right as left handed people. On the very bottom, two zoom buttons (Figure 3.3, No. 1) adjust the zoom level of the map. The *Focus Detection*-button (Figure 3.3, No. 2) is responsible for enabling and disabling the focus-detection via camera. As previous work shows (i.e. [38]), it is desired by users to be able to turn focus-detection on and off on demand. The *Sync*-button (Figure 3.3, No. 3) is to synchronize the whiteboard's view with the view currently shown on the tablet screen. The topmost *Show View*-button (Figure 3.3, No. 4) is for showing on- and off-screen visual cues of the current personal view on the whiteboard. The application also has a search functionality (Figure 3.3, No. 5) and the map view's type can be switched to hybrid/satellite view (Figure 3.3, No. 6).

3.2.3 Visual Cues

Both, the individual tablet computers as well as the collaborative whiteboard can show different parts of maps. Also, the type of view (hybrid and satellite) and the zoom level can differ. As long as a tablet's view is overlapping with the view shown on the whiteboard it is referred to as on-screen visual cue (Figure 3.4). All other cases are off-screen representations. To detect the personal view in a collaboration setup, each user has an individual predefined color, in which visual cues appear.

Figure 3.4: Examples for tablet views that are handled as on-screen visual cues.

3.2.4 On-Screen Visual Cue

The following section will describe the appearance and design of on-screen visual cues in detail.

Representation of the Tablet's View

The on-screen visual cue is a rectangle, which indicates the current tablet view's position. This rectangle has the exact dimensions of the tablet's current map view. If the map application's window gets resized on the tablet computer, the tablet's view representation also appears in the appropriate size. The rectangle's colored border attracts attention and is responsible for user identification. To make sure, the rectangle emphasizes from the background, independent of its color, we added a white background. Depending on the zoom levels of the devices, the tablet's view can be rather small and therefore hard to detect (Figure 3.5) on the large whiteboard display. The application handles this problem with an additional magnification and animation.

Magnification and Animation

To ensure a zoomed-in view can be detected on a large overview, the application provides rendering multiple magnified versions of the on-screen representation. These enlarged rectangles surround the actual tablet's view

Figure 3.5: If the tablet's view is zoomed-in and the whiteboards current zoom level is very low, a rectangle representing the current tablet's view is hard to detect. Therefore, rendering multiple, magnified rectangles is important.

and are attracting attention. An animation shows the rectangles, beginning with the largest and ending with the actual tablet's view. We adjust the size of the enlarged rectangles dependent on the current zoom level of the whiteboard application's map. Accordingly, the visual representation's size is appropriate and visible on the screen for all zoom levels. As the *Show View*-button of the application is hit, the animation is played and the visual cue of the actual view stays until the button is released. The smallest rectangle, representing the actual view of the tablet, has a white, semi-transparent filling. In figure 3.6 this is shown for multiple users.

3.2.5 Off-Screen Visual Cue

As the tablet screen can display maps that are completely outside of the view currently shown on the digital whiteboard screen, it comes to off-screen visual cues. The basic idea is, to enable estimating how far the tablet's view is away from the whiteboard's view and in which direction it is located. We have implemented borderline representations in two versions, a rendering,

Figure 3.6: The smallest rectangle with a semi-transparent filling shows the actual view of the tablet. The representation's color is unique for every user. Multiple visual cues can be shown simultaneously and can overlap.

as shown in figure 3.10 and lines in the form of ambient light around the whiteboard (Figure 3.12). Both options show the tablet's view, shifted to the border of the whiteboard's view. While the borderline's position is indicating the tablet view's location, the opacity is indicating the distance. We decided for light opacity to show high distances and fully opaque lines to show close views.

Edge-Aligned Borderlines

For situations where the tablet's view is orthogonally shifted from the whiteboard's view, the borderline appears on the closest aligned border (Figure 3.7). The length of the border shows the actual view of the tablet, which means the higher the zoom level on the tablet screen, the shorter is the borderline.

Corner Visual Cue

If the tablet's view is diagonal shifted to the whiteboard's view, the visual cue appears in the corner of the whiteboard screen. This includes cases where

1. the tablet's view is completely in the corner (Figure 3.8 (a)),
2. the tablet's view is partly in the corner,
   (a) top/bottom of the view is in the corner (Figure 3.8 (b)),
   (b) left/right part of the view is in the corner (Figure 3.8 (c)).

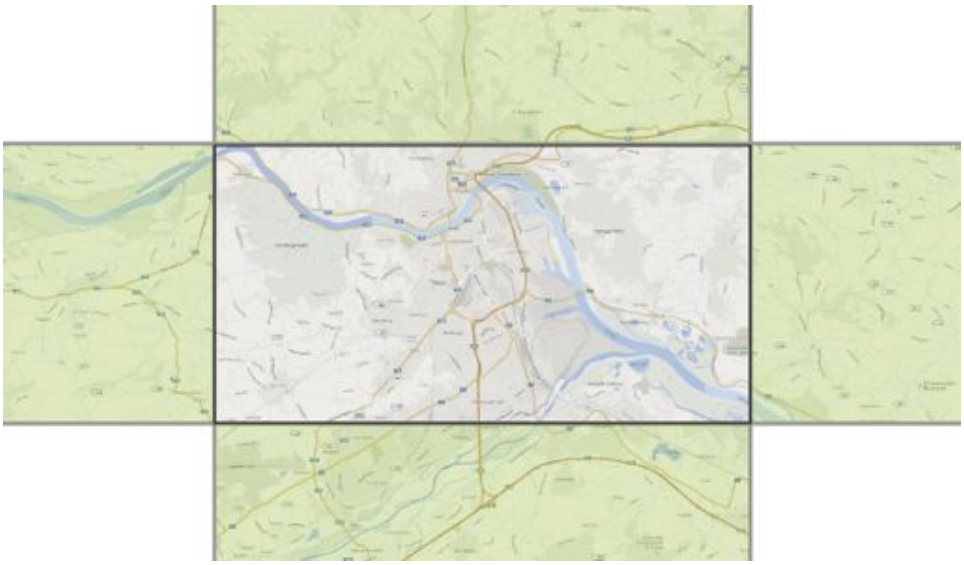


Figure 3.7: If the tablet's view is completely in one of the green areas, it is an orthogonal shift and the borderline therefore appears only on one of the edges.

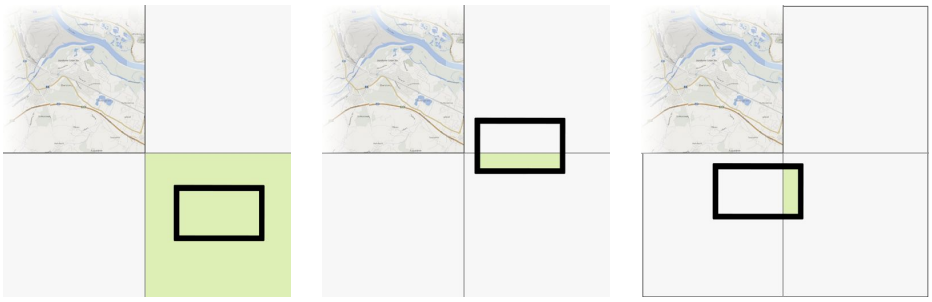
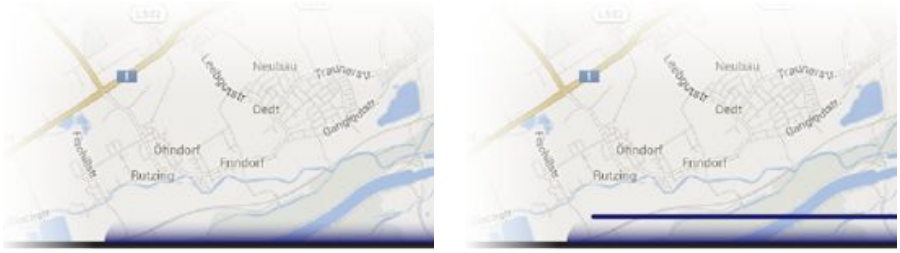


Figure 3.8: Examples for diagonal shifts represented in the corner of the whiteboard screen. The tablet's view is completely shifted to the corner (a). The tablet's bottom of the view is partly shifted in the corner (b). The tablet's right side of the view is partly shifted in the corner (c).

Distance and Opacity

In order to indicate distance, we adjust the borderlines opacity. The closer the tablet's view is to the whiteboard's view, the more opaque it becomes. As the distances can highly differ between use cases for the application, the maximum distance can be adjusted in the settings of the whiteboard. Henceforward the maximum distance, the opacity of the borderline is zero.

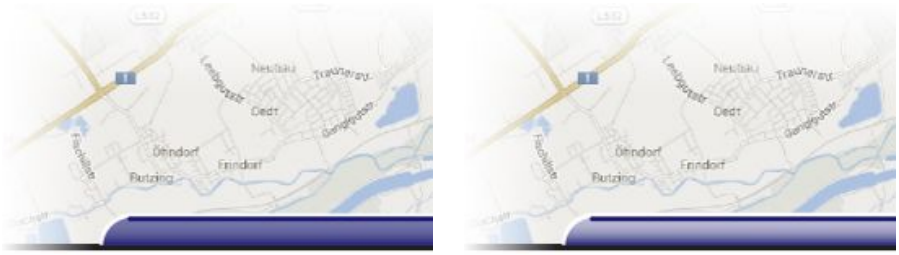


(a) (b)

Figure 3.9: The rendered borderlines show the direction and the distance of the tablet's view to the whiteboard's view. As the opacity of the lines indicates the distance, big distances can be hard to detect (a). Therefore an additional thin, fully opaque line attracts attention. Furthermore, the final version of the borderline has a white background (Figure 3.10) to emphasize from the map (b).

Figure 3.10: The final, rendered borderline of a tablet's view, that is orthogonally shifted to the south of the whiteboard's map view.

Rendered Borderline To make sure the rendered border can always be detected, although the distance and therefore the transparency is high (Figure 3.9 (a)), a thin colored line on the inside edge of the border attracts additional attention (Figure 3.9 (b)). For further visual support, a slightly bigger, white background line is rendered, so it becomes easier to interpret the opacity level, independent of the background's color (Figure 3.10). The opacity calculation will be discussed in the implementation chapter.



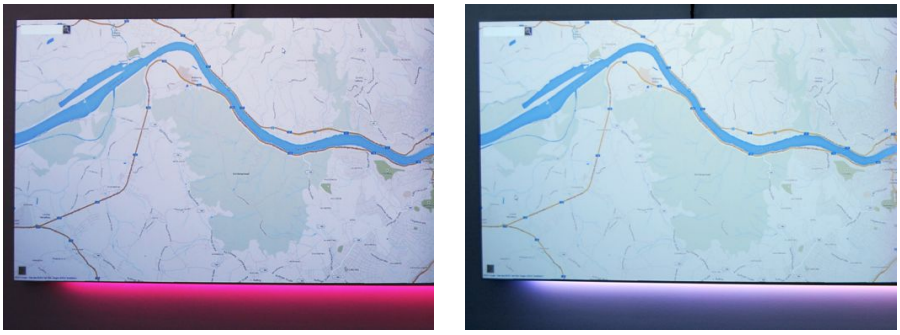
(a) (b)

Figure 3.11: Examples for rendered borderlines: with 60% opacity, which indicates a distance of about 36 km (a) and with 30% opacity, which is a distance of about 18 km (b).

Figures 3.11 (a) and (b) show empirically measured example renderings of how the distance influences the opacity of the borderline: the first example (Figure 3.11 (a)) has an alpha value of 60%, which is a distance of about 36 km (as the maximum distance is set to 60 km). In contrast, the second example shows an alpha value of only 30% (Figure 3.11 (b)), which indicates a distance of about 18 km.

Ambient Light Similar to the rendered borderlines, the off-screen indicators via ambient light appear around the edges of the whiteboard in a user's individual color (Figure 3.12). The ambient light can show direction and distance like the rendered borderlines, but need no screen space. Thus, this technique does not hide any information on the map. For indicating distance, the borderline color and the white background are blended. Figures 3.13 (a) and (b) show examples of different distances.

Figure 3.12: The ambient light can show off-screen visual cues for multiple users simultaneously.



(a) (b)

Figure 3.13: Examples for borderlines via ambient light: with a distance of 47 km (a) and with a distance of 14.5 km (b).

Chapter 4

Implementation

The following chapter includes the implementation details of the described application. After a short introduction to the map application, we define calculations for the different visual cues. The implementation of on- and off-screen visual cues will be discussed precisely, followed by the implementation of the ambient light and multi-user handling. The last two parts of this chapter cover the focus-detection and describe the underlying framework briefly. Source code parts, shown in this chapter, are simplified and adapted to increase readability.

As the application works via wireless network connection, all devices should be part of the same wireless network. The application is designed, implemented and tested for Windows 8. The digital whiteboard we used is a projection that is also running on a Windows 8 machine.

4.1 Map App

This map application is built-up on the GMap.NET[1] framework, which will be discussed further in the underlying framework (Section 4.5). Basis for positioning visual cues on these maps are latitude and longitude values. The basic setup consists of three applications:

1. host,
2. whiteboard application, and
3. application for tablets.

[1] http://greatmaps.codeplex.com/

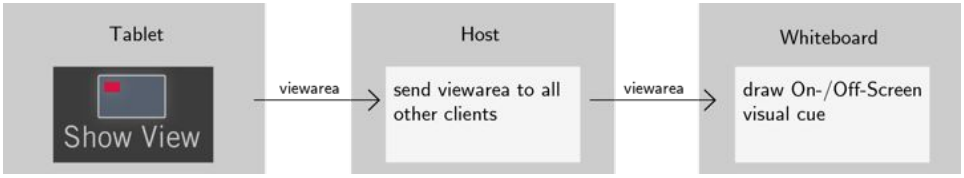


Figure 4.1: If one of the clients hits the *Show View*-button, the current `viewarea` is sent to the host application and distributed to the other clients. The receiving whiteboard application handles the appropriate representation of the tablet view.

The `viewarea` is the most important data sent between these applications as shown in Figure 4.1. It includes the following attributes

- the view's top latitude,
- the view's bottom latitude,
- the view's left longitude,
- the view's right longitude,
- the unique client ID.

Additionally to the view's bounds, the unique client ID is sent to identify the requesting user.

Responsible for handling the connection between the devices, sending and receiving messages, is the *host* application. This application can either run on the whiteboard or on one of the tablet devices in the network. But depending on the use case, participants and their tablet devices will come and go. Therefore, it is recommended to start the host application on the computer for the whiteboard. All devices are basically clients. Clients can subscribe themself to the connection. Subscribed clients get a unique client ID, which enables to identify and distinguish between clients.

Whiteboard The map application running on the whiteboard is responsible for showing the visual cues of the different tablet views. It also works independently of the tablet application. As the whiteboard application receives the appropriate `viewarea` from the host application, it interprets it and decides which visual cue should appear on which location of the current map view.

Tablet The map application for the tablet computers is designed for tablets and touch interaction, but works on every windows machine. Therefore, it can also be used on notebooks. It sends the current `viewarea` when the appropriate event is triggered. This can happen via button click or as a result of the focus-detection.

4.2 The Workflow

Figure 4.2 shows the workflow in detail to clarify it step-by-step. Basically, if the *Show View*-button is hit on one of the tablet computers, or the event is raised by the focus-detection, the `viewarea` is sent from the tablet to the host. Then the host broadcasts it to the whiteboard application. First, the whiteboard application checks if the user ID, which raised the event, is already known. Is an ID unknown, a new user is created, saved to the user-list and assigned to a color.

The first step for the actual visual representation of the view is to decide whether or not the received `viewarea` is on-screen of the currently shown map on the whiteboard. Therefore, a boundingbox-check is implemented. If the tablet's view is on-screen, the coordinates for the rectangles need to be calculated, the rectangles are styled appropriately and drawn to the map. Is the tablet's view off-screen, the relation of the location of the tablet's view, to the whiteboard, needs to be detected. Thus, the four corners get checked first. If no corner can be identified, the application tests for orthogonal shifts. Once the appropriate border is detected, the coordinates and distances are calculated, the borderlines are styled according to the user ID and the distance. The last step is again, to draw the borderline on the map. If it comes to ambient light borderlines, the color values are getting sent to the ambient light controller.

4.3 Visual Cue

In the following paragraphs the implementation for showing the discussed visual cues will be described in detail.

4.3.1 On-Screen Visual Cue

Is the view of the requesting tablet overlapping with the whiteboard's view, an on-screen visual cue in the form of the mentioned animated rectangles is displayed.

Visualize Tablet View

The size of the tablet's view visual representation corresponds to the exact dimensions of the tablet's current view. As the `viewarea` of the tablet is sent to the whiteboard, the rectangle with these receiving values is drawn. However the order of the coordinates need to be changed to draw a rectangle. The third and fourth coordinate need to be switched, otherwise the displayed polygon has the shape of an hourglass. All colored rectangles have a stroke width of 4 pt, to emphasize them from the background, white rectangles with a stroke width of 5 pt are rendered additionally in advance.

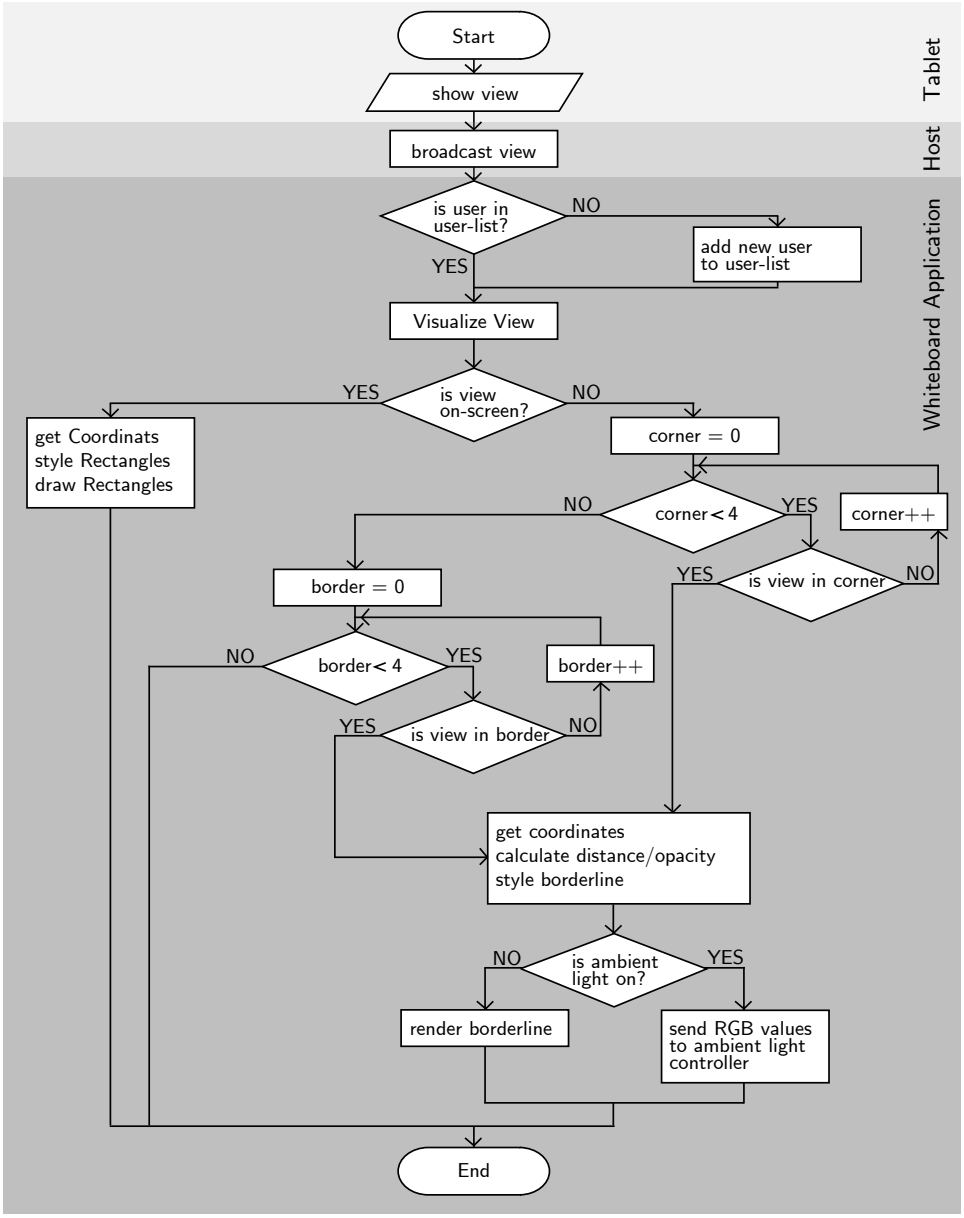


Figure 4.2: This flowchart shows the basic workflow when a *Show View*-event is raised by one of the tablet devices. If the event is triggered by the appropriate button or by the focus-detection makes no difference in the workflow. The three gray scales in this figure indicate, that these steps happen in different applications. The first part in the tablet application, the second is done by the host and the whiteboard application is responsible for the last part.

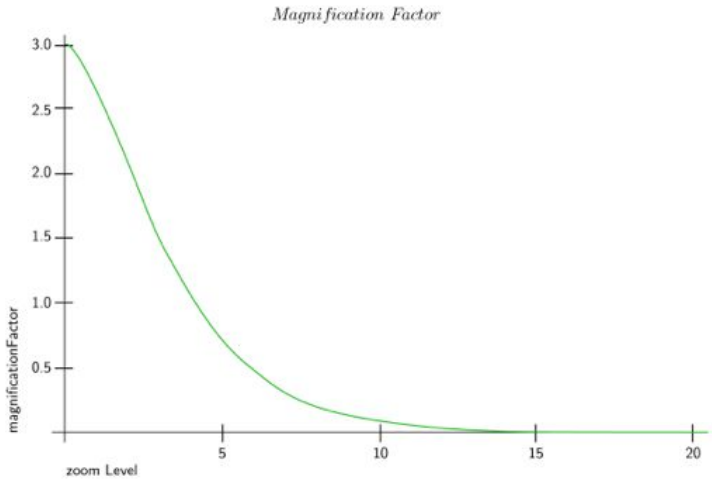


Figure 4.3: This diagram shows the resulting values of our `magnificationFactor`-calculation. It results in low values for highly zoomed in tablet views and high values for low zoom levels.

Animation and Semi-Transparent Overlay

For the animation of the on-screen visual cue multiple rectangles need to be calculated and displayed. The original coordinates are magnified by a `magnificationFactor`, which is dependent on the current zoom level of the map, displayed on the whiteboard. Additionally, the `widthHeightRatio` of the view needs to be considered in the calculation. As there are longitude values from 180 to −180 and latitude values from 90 to −90, either the longitude or the latitude values need to be adjusted to keep the rectangles in the same shape (Program 4.1 Line 4).

Scaling of the magnified rectangles should be very low for highly zoomed in maps and rather large for low zoom levels. Therefore, we empirically evaluated and decided for a function with an exponential growth for the `magnificationFactor` (Figure 4.3).

Program 4.1 Lines 5-7 show the actual values to calculate the appropriate `magnificationFactor` for the coordinates. In our solution, this zoom value can vary from 2 to a maximum of 20. This `magnificationFactor` needs to be added or subtracted, depending on the currently calculated coordinate (Figure 4.4 and Program 4.1 Lines 12-26).

This function results in very low values for high zoom levels. Until a zoom level of 17 the `magnificationFactor` is zero, which means that the appearing rectangles are the same size. The maximum `magnificationFactor` is 3, for a zooming level of zero, but as the minimum zoom level of the current implementation is 2, the rectangle is maximum scaled by a value of 2.05 (Figure 4.4).

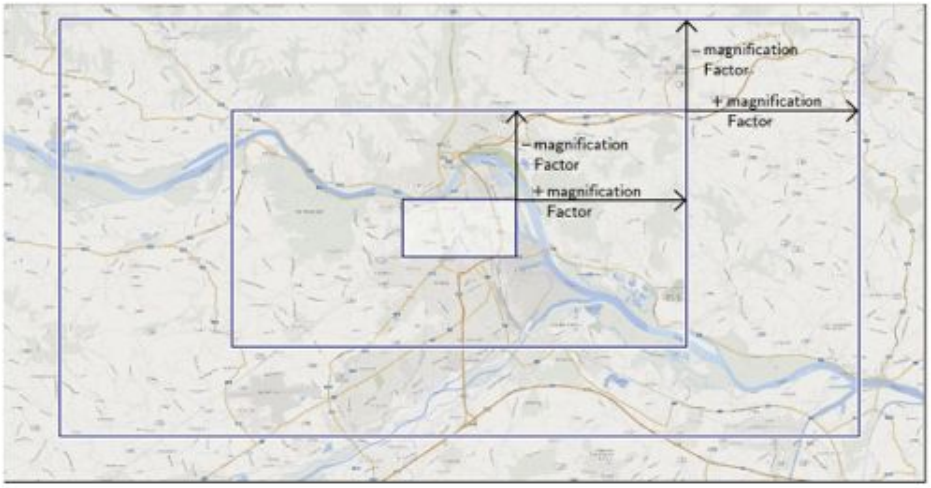


Figure 4.4: The `magnificationFactor` is the value that indicates how much the original rectangle is scaled to get the magnified rectangles for the animation. Depending on the actual coordinate, the `magnificationFactor` needs to be added or subtracted.

The program `zoomingAnimation` (Program 4.1) shows how the coordinates for the magnified rectangles are calculated. The *zoomingAnimation* method is called asynchronously and let the rectangles appear and disappear with a pause of 250 milliseconds.

If the map on the whiteboard has a higher zoom level, than the map of the tablet device, the actual view would be to big to visualize. Consequently, the whole whiteboard view gets an semi-transparent overlay.

4.3.2 Off-Screen Visual Cue

An off-screen visual cue appears, if the `viewarea` of the tablet is completely outside the whiteboard's view. First, the application checks if the tablet's view is in one of the four corners or orthogonally shifted. An orthogonal shift is displayed by a visual cue of one borderline, while a visual cue in one of the corners is achieved by generating two borderlines.

Orthogonal Shift

An orthogonally shifted view of the tablet is visualized by a single line along the edge of the whiteboard's display (Figure 3.7) or the appropriate ambient light border. The line's boundaries are depending on the tablet's view and are aligned to the edges of the whiteboard's view. Therefore, vertically aligned borderlines have the latitude value of the whiteboard's edge and

```
1  //calculate coordinates for magnified rectangles and add to map
2  private async void startZoomingAnimation(int uniqueClientId)
3  {
4      double whRatio = viewarea.getWidth() / viewarea.getHeight();
5      magnificationFactor = Math.Pow(5, (MainMap.Zoom / (-3))) *
6          (MainMap.Zoom + 3);
7      magnificationFactor *= 1.5;
8      ZoomedCoords.Clear();
9
10     for (int i = 2; i >= 0; i--)
11     {
12         ZoomedCoords.Add(
13             new PointLatLng(Coords[0].Lat + (magnificationFactor * i),
14             Coords[0].Lng - (magnificationFactor * i * whRatio)));
15
16         ZoomedCoords.Add(
17             new PointLatLng(Coords[1].Lat + (magnificationFactor * i),
18             Coords[1].Lng + (magnificationFactor * i * whRatio)));
19
20         ZoomedCoords.Add(
21             new PointLatLng(Coords[2].Lat - (magnificationFactor * i),
22             Coords[2].Lng + (magnificationFactor * i * whRatio)));
23
24         ZoomedCoords.Add(
25             new PointLatLng(Coords[3].Lat - (magnificationFactor * i),
26             Coords[3].Lng - (magnificationFactor * i * whRatio)));
27
28         Rectangle = new GMapPolygon(ZoomedCoords);
29         ...
30         ZoomedCoords.Clear();
31     }
32 }
```

Program 4.1: This code part shows how the coordinates for the magnified rectangles are calculated. The `widthHeightRatio` is important for remaining the correct ratio of longitude and latitude values. The `magnificationFactor` is dependent on the current zoom level of the map, displayed on the whiteboard.

the longitude top- and bottom values of the tablet's view. Consequently, for horizontally aligned borderlines, the longitude value is taken from the whiteboard top/bottom border and the latitude values are the left and right values of the tablet's `viewarea`. Additionally to longitude and latitude, the orientation of the borderline is important for the correct presentation of the gradient line. These calculations are necessary for both options, the rendered, as well as the borderlines via ambient light. The orientation is therefore saved, and later used to define the gradient values for the rendered borderline or rather used set the correct RGB values for the ambient light.

```
1 // calculating alpha value
2 float alpha = (float) (1 - (calcDistance(uniqueClientId) /
      maxDistanceToGoal));
3
4 if (alpha > 1) alpha = 1;
5 if (alpha < 0) alpha = 0;
6 ...
7
8 // adding gradient stops for borderlines
9 gradientBrush.GradientStops.Add(new GradientStop(Color.FromScRgb(alpha,
      userColor.ScR, userColor.ScG, userColor.ScB), 0.0));
10 gradientBrush.GradientStops.Add(new GradientStop(Color.FromScRgb(alpha,
      userColor.ScR, userColor.ScG, userColor.ScB), 0.8));
11 gradientBrush.GradientStops.Add(new GradientStop(userColor, 1.0));
```

Program 4.2: The alpha value for the borderlines is calculated by the distance between the center of the tablet's view, the center of the whiteboard's view and the maximum distance. Alpha cannot be greater than one, or less than zero.

Distance and Opacity

Every rendered borderline is split in two parts, a gradient and a fully opaque part. For the correct orientation the end and start point of the x and y-values of the line need to be adjusted to the orientation value of the line. The distance between the tablet's view and the whiteboard needs to be calculated for the gradient part. The alpha value is the distance divided by the maximum distance, defined in the settings of the whiteboard application (Program 4.2, Line 2). The alpha value for the borderlines needs to be between zero and one. For the borderlines via ambient light, this distance is blended with the user's color.

Corners

The application needs to check each corner separately. Per corner, there are three possible cases for the tablet's view in relation to the whiteboard's view. The tablet's view can be exactly shifted diagonally and therefore be completely in the corner (Figure 3.8 (a)), but also if only a part of the tablet's view is in the corner, borderlines appear in the corner (Figure 3.8 (b) and (c)). Consequently it is more likely that the view of the tablet is at least partly in a corner, therefore corners are checked before orthogonal shifts.

Every corner is made of two borderlines, a horizontal and a vertical line. These borderlines are generated exactly as the borderlines for orthogonally shifted views. But the lines need an additional shift to appear in the correct position. Therefore, the distance between the boundaries of the tablet's view

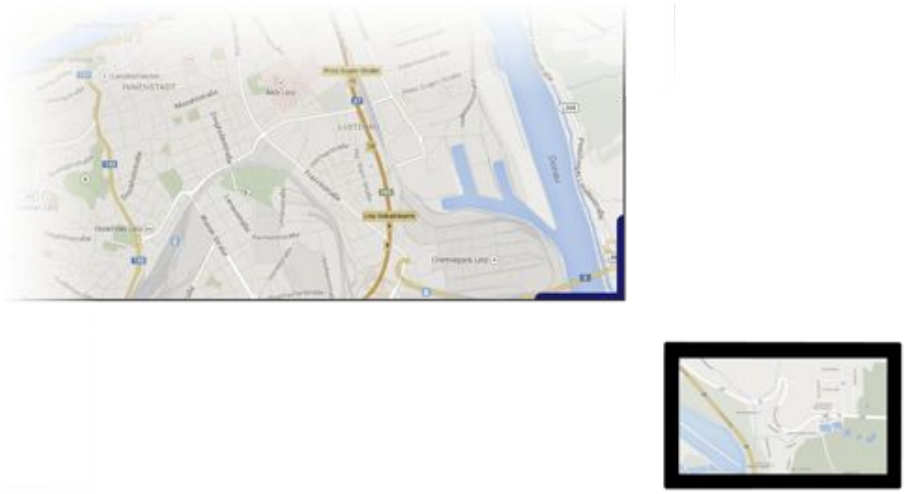


Figure 4.5: Case 1: If the tablet's view is diagonally shifted from the whiteboard's view, the resulting borderlines are each half the length of their original size.

and the whiteboard's view is calculated. In the program 4.3 (Lines 8 and 20) it is shown that the distance, in this case for the right bottom corner, is calculated by: the difference between the right longitude values of the views, for the horizontal line and the bottom latitude values for the vertical line.

Corner Shift In general, the shift for the lines in the corner is dependent on the distance between the two views. It is calculated by

$$shift = distance - \frac{lengthAlignedSide}{2}. \tag{4.1}$$

The `lengthAlignedSide` is the width of the view, for the calculation of horizontal lines, or the height for the calculation of vertical lines. By shifting the line accordingly, half of the view's width/height is visible in the corner. Besides the distance, the direction of the tablet's view is important. For a more accurate representation of the view's direction an additional calculation step is needed, for views that are partly shifted to the corner. Only views that are completely in the corner appear with borderlines with each a length of half the width/height of the original view (Figure 3.8 (a)). Otherwise, depending on the case, the latitude/longitude values from the original view are taken for the vertical/horizontal visual cue of the borderline, the second borderline is shown only to 25% percent (Program 4.4). As the figures 4.5, 4.6, 4.7 show, this visual cue makes it easier to assume the spatial location of the view.

Figure 4.6: Case 2: Is the tablet's view is only partly in the corner, as in this case shifted to the south of the of the whiteboard's view, the horizontal line has the same longitude values as the original view. Only a quarter of the vertical line is shown.

Figure 4.7: Case 3: Like in the former figure (4.6), the tablet's view is only partly in the corner, in this case on the right side of the whiteboard's view. Here the vertical line has the same latitude values as the original view. Only a quarter of the horizontal line is shown.

There is a special case that can cause an incorrect presentation of the corners. This case is dependent on the ratio of the tablets view and the distance between the two views (Figure 4.8).

```
1  // right bottom corner
2  if (userViewarea.Right > MainMap.ViewArea.Right &&
3      userViewarea.Top < MainMap.ViewArea.Bottom ||
4      userViewarea.Left > MainMap.ViewArea.Right &&
5      userViewarea.Bottom < MainMap.ViewArea.Bottom)
6  {
7      // horizontal line
8      distance = calcDistanceToWBView(userViewarea.Right,
9          MainMap.ViewArea.Right);
10     calcShift(distanceToBorder, userViewarea, userViewarea.getWidth());
11     ...
12     addPointsForBorder(uniqueClientId, MainMap.ViewArea.Bottom,
13         userViewarea.Left - shift);
14     addPointsForBorder(uniqueClientId, MainMap.ViewArea.Bottom,
15         userViewarea.Right - shift);
16     orientation = User.Orientations.bottom;
17     addLineToMainMap(userViewarea,0);
18
19     // vertical line
20     distance = calcDistanceToWBView(Viewarea.Bottom,
21         MainMap.ViewArea.Bottom);
22     calcShift(distanceToBorder, userViewarea, Viewarea.getHeight());
23     ...
24     addPointsForBorder(uniqueClientId, (userViewarea.Top + shift),
25         MainMap.ViewArea.Right);
26     addPointsForBorder(uniqueClientId, (userViewarea.Bottom + shift),
27         MainMap.ViewArea.Right);
28     orientation = User.Orientations.right;
29     addLineToMainMap(userViewarea, 2);
30 }
```

Program 4.3: This program shows how lines for the right bottom corner are created. When the rendered borderlines are added to the map (Lines 17 and 29), an index handles the order of the lines (index 0 and 1 are horizontal lines (white background and colored line), index 2 and 3 are the vertical lines). For the ambient light these calculations are also necessary and followed by converting the coordinates to pixel values.

If

$$distance \geq \frac{width}{2}, \tag{4.2}$$

the resulting `shift` is ≤ 0. In this case, `shift` is set again to make 25% of the borderline visible (Program 4.4). We are aware, this causes an incorrect result. However, the corresponding borderline would not be visible otherwise or too short and therefore hidden by the second borderline of this corner.

Figure 4.8: `shift` for borderlines in the corner is dependent on the distance between the two views and the length of the aligned side. The length of the aligned side is the width of the tablet's view for horizontal lines and the height for vertical lines.

```
1 // calculating shift for borderlines
2 private void calcShift(double distance, double lengthAlignedSide)
3 {
4     shift = (distance - (lengthAlignedSide / 2));
5
6     if (shift <= 0)
7     {
8         shift += lengthAlignedSide;
9     }
10 }
```

Program 4.4: Calculating `shift` for the borderlines of the corner. If `shift` is less or equal 0, then it is increased to make a quarter of the original length visible. This makes sure, every line is shifted in the correct direction and therefore is visible.

Rendered Borderline The rendered lines have a stroke width of 16 pt, but are only shown half, as they are aligned to the border of the screen. To emphasize no matter what the background map currently looks like, each line has a slightly larger white background line with a stroke width of 20 pt.

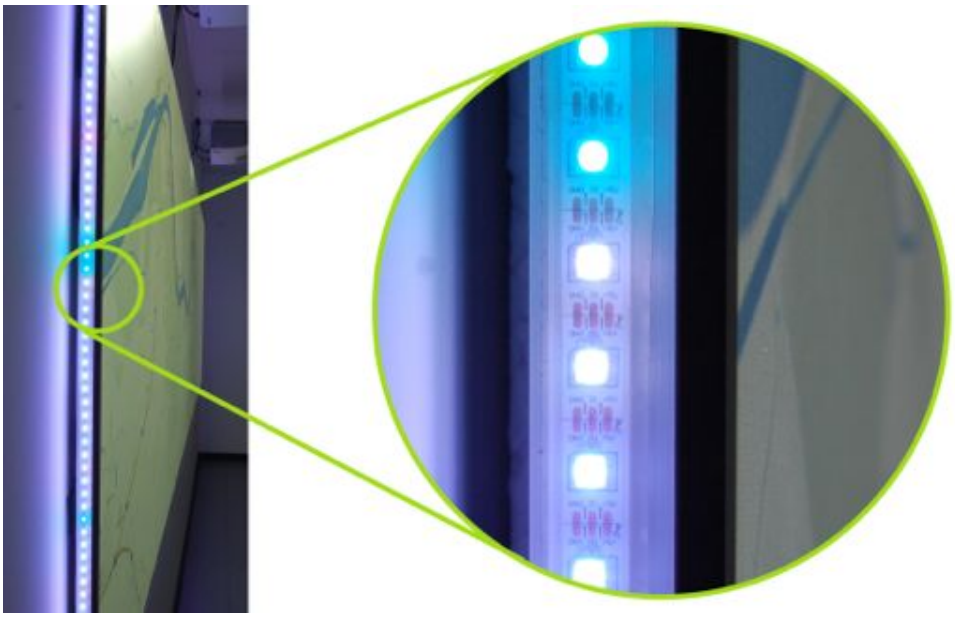


Figure 4.9: Three LED strips around the edges of the digital whiteboard are responsible for the ambient light. Via an ambient light controller software, each LEDs can be addressed individually.

Ambient Light The ambient light is implemented using three addressable LED strips, with 60 LEDs per meter, which are arranged around the edges of the whiteboard (Figure 4.9). Altogether we have 624 LED sources. These strips are connected and controlled by a Leonardo Arduino microcontroller (Figure 4.10). An ambient light controller runs on the whiteboard computer and enables managing each LED source individually. The whiteboard application is connected to this controller software by a TCP connection. Via that connection, it sends a `pixelValues`-array with the appropriate RGB values. The ambient light controller can access the individual LED sources and maps the received data accordingly. The LEDs RGB values are reached via PWM.[2] This technique uses a rectangular pulse wave. The pulse width is modulated to send the appropriate values to the LEDs. Each LED source takes the first value and cuts it from the stream. The rest of the stream gets forwarded to the following LEDs.

The `pixelValues`-array gets updated for each *Show View*-event, a tablet sends to the whiteboard application. This `pixelValues`-array is filled according the former described calculations for borderlines. Therefore, it is necessary to compute the rendered borderlines and convert the latitude and longitude values to the according pixels. These pixel values are the start and end pixels of the borderline. The color and the opacity of the line are

[2] http://arduino.cc/en/Tutorial/PWM/

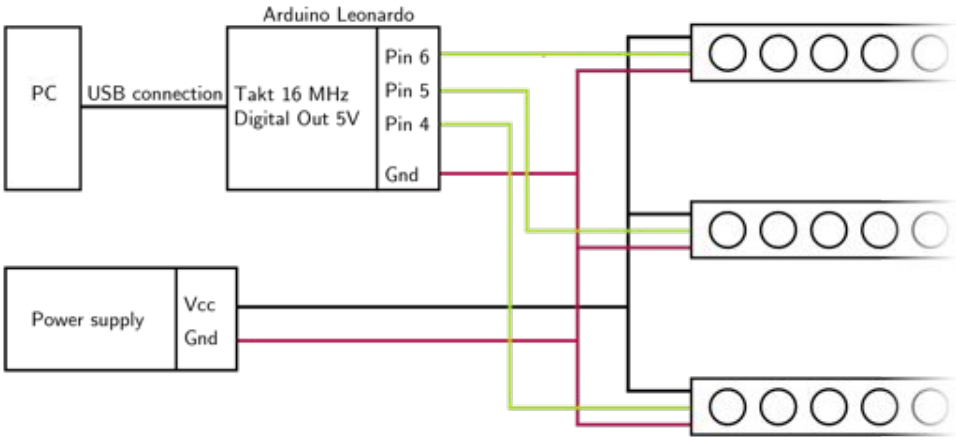


Figure 4.10: This sketch shows the wiring for the three LED strips, connected to the Arduino board (Pin 4, 5 and 6). The data supports a computer, running the ambient light controller and connected via USB to the Arduino board.

```
1  // filling the pixelArray sent to the ambient light controller
2  public Color[] fillPixelValues(int uniqueClientId, int index)
3  {
4      // calculate width/height of borderline
5      ...
6      // top border
7      if(orientation == Orientations.top)
8      {
9          start = MainMap.FromLatLngToLocal(borderLine.Points[0]).X;
10         end = MainMap.FromLatLngToLocal(borderLine.Points[0]).X + width;
11         checkBounds(uniqueClientId, width, index);
12     }
13     // other orientations
14      ...
15     // blending
16     for (int i = start; i <= end; i++)
17     {
18         pixelValues[i].R = (byte)((alpha * userColor.R)
19             + ((1 - alpha) * 255));
20         pixelValues[i].G = (byte)((alpha * userColor.G)
21             + ((1 - alpha) * 255));
22         pixelValues[i].B = (byte)((alpha * userColor.B)
23             + ((1 - alpha) * 255));
24     }
25     return pixelValues;
26 }
```

Program 4.5: Computing the pixel array for the ambient light.

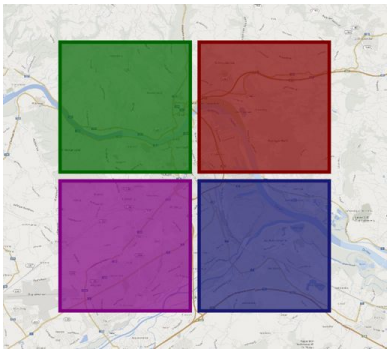


Figure 4.11: These four colors are predefined for the first users. Via the settings dialog, these colors can be adjusted.

blended with the white background by alpha blending (Program 4.5, Lines 18-23). This technique enables to show different distances for tablet views (Figure 5.10). The `pixelValues`-array is filled with the resulting color values (Program 4.5).

4.3.3 Multi-User Handling

As already mentioned, the shown setup implicates the simultaneous handling of multiple users. Even if visual cues of multiple users appear at the same time on the whiteboard screen, it needs to be possible to distinguish between them and identify them. We can identify devices by their unique client ID and distinguish their visual cues by color-coding. The `user`-class of the map application is a part of the whiteboard application and is responsible for user handling. The whiteboard application handles the initialization of the user colors. Four pre-defined colors (DarkGreen, MidnightBlue, Purple and Maroon) guarantee a high contrast to the map in the background (Figure 4.11). These colors can be modified via a color picker in the settings of the whiteboard application. Therefore, the current version of the application is designed for up to four tablet devices. Theoretically, the application can handle more users.

The appearance of each on- or off-screen visual cue and particularly, the disappearance of these visual cues need to be independent of each other. The solution for removing only the visual cue of a single user, as desired, is to tag all visual cue elements with the unique client ID of a user (Program 4.6). This approach makes sure, elements can be identified with a user and therefore, appear and disappear on demand. These visual elements are implemented as `Markers`. We will further describe these markers in section Underlying Framework. For the ambient light, the start and end pixel values are saved for each user. Thus, the appropriate pixels can be reset to black to let an ambient borderline disappear.

```
1  // clear all rendered visual elements of a specific user
2  private void clearMainMap(int uniqueClientId)
3  {
4      if (MainMap.Markers.Count() > 0)
5      {
6          for (int i = (MainMap.Markers.Count() - 1); i >= 0; i--)
7          {
8              if (MainMap.Markers[i].Tag != null &&
9                  MainMap.Markers[i].Tag.Equals(uniqueClientId))
10             {
11                 removeMarker(i);
12             }
13         }
14     }
15 }
16
17 // clear ambilight of a specific user
18 public void clearPixelValues(int uniqueClientId)
19 {
20     for (int i = userStart; i <= userEnd; i++)
21     {
22         pixelValues[i].R = 0;
23         pixelValues[i].G = 0;
24         pixelValues[i].B = 0;
25     }
26     Client.SendObject(new AmbientLightTcpMessage(
27         MessageType.RgbPerPixel,
28         new AmbientLightValues(){
29         Values = ByteConversion.Convert(pixelValues)}));
30
31     userStart = 0;
32     userEnd = 0;
33 }
```

Program 4.6: All visual elements (markers) on the map are tagged with the unique user ID. This enables deleting elements by user or respectively, setting the appropriate LEDs to black.

4.3.4 Synchronize View

The *Sync*-Button enables users to show their current personal view on the whiteboard. When the *Sync*-Button on one of the connected tablet computers is hit, the position of the view and the zoom level get sent to the host application. The host application is responsible for distributing the information to the whiteboard, which will navigate to the received position and adjust the zoom level according to the request. Each connected tablet computer can synchronize its view with the digital whiteboard and can therefore overwrite the last view.

Figure 4.12: The used tablet computer with the map application.

4.4 Head Tracking

Focus-detection can be enabled and disabled for each tablet device individually via a button. One major benefit of this implementation is, that it works calibration-free. When the focus-detection is enabled, the front camera of the tablet is activated, which is noticeable via a notification LED source next to the camera. If the application can't detect a user's face, the application supposes that their focus is not on the tablet anymore and therefore renders the visual cue as described before. The algorithm for the face-detection is a project of the University of Debrecen, Hungary and was developed, by a team of students, during the 17th Summer School on Image Processing.[3]

Within the head tracking algorithm, the camera captures and these frames are analyzed. First, these frames are converted to gray for further processing. The converted frames are checked for any detectable faces. As the program 4.7 (Lines 2-5) shows, the method `DetectHaarCascade` analyzes every `grayframe` for rectangular regions that may contain the haar-like features, the cascade has been trained for. Each image is scanned multiple times at several scales. `HAAR_DETECTION_TYPE.DO_CANNY_PRUNING` is a heuristic to reduce the number of analyzed areas. If this flag is set, an Canny edge detector [5] is used to exclude image areas where the searched elements

[3] http://www.inf.unideb.hu/~ssip/teams/team4/index.html

(a) (b)

Figure 4.13: The front camera's view (a) without any adjustment and (b) with additional wideangle lens. As shown in (b), with the additional lens, the whole head is in the focus of the camera and can easier be detected.

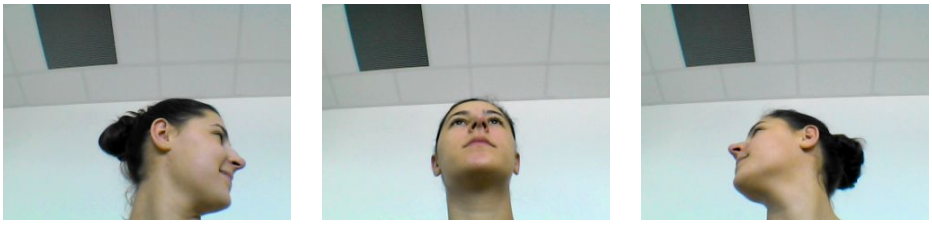


Figure 4.14: These pictures were taken as soon as the head-detection recognizes the focus is off-screen.

cannot be because of too much or too few edges. This pruning-procedure speeds up the detection of faces. It's threshold values are specially tuned for face-detection. If the algorithm can find suitable image areas, they get grouped together. The result are rectangular regions, which get saved in the `faces`-array. As long as this array is not empty, the program could detect a face and therefore, the user's focus is not off-screen.

The built-in front camera of the Acer tablet (Figure 4.12) is designed for video-telephony. Head tracking was inaccurate and not reliable while working with the tablet in a convenient way, as the person to track was very close to the camera. Therefore, a lens was added, originally designed as wide-angle extension for the Microsoft Kinect. To make the lens removable, the additional lens was attached to the tablet computer with a clip, printed with a 3D-printer. Figure 4.13 (a) shows the cameras picture without any adjustment, figure 4.13 (b) with the additional wideangle lens. Figure 4.14 shows at which point the algorithm recognizes the focus to be off-screen.

```
1  // detecting the face
2  MCvAvgComp[] faces = grayframe.DetectHaarCascade(
3      haarfaceFeature, scaleFactor, minNeighbors,
4      HAAR_DETECTION_TYPE.DO_CANNY_PRUNING,
5      minWindowSize)[0];
6
7  // if a face can be detected, the focus is on-screen
8  if (faces.Length != 0)
9  {
10     offscreen = false;
11 }
12 ...
```

Program 4.7: If the `faces`-array is empty, no face could be detected and a *focus off-screen*-event is sent to the whiteboard.

4.5 Underlying Framework

The described setup is written in C# and uses Windows Presentation Foundation.[4] For displaying and handling map data GMap.NET[5] has been used. This open source framework enables loading maps, searching through them and placing elements on the maps. These elements are implemented as markers, which are updated on any navigation or zooming. Both applications, designed for the whiteboard and for the tablet computers are basically the same. But they differ in interaction possibilities, as well as, in handling sending and receiving information, from and to the host application.

The underlying framework enables adding markers to the map. Markers can have different shapes and design. In this application the markers are `GMapPolygon` objects. These marker objects are saved in a `GMapMarker`-generic `ObservableCollection`. Every marker has a tag attribute, which is used to identify the marker with the appropriate user. As a corner visual cue is made of four borderlines (two colored borderlines and two white background lines), the marker collection can store up to four markers per user. For every request of a user, their individual markers are removed from the collection and refreshed (Program 4.6). Responsible for the connection between the devices is the Windows Communication Foundation.[6]

The basis of the head-detection is the OpenCV[7] library. It is published under the BSD license and can be used with different programming languages. However for .NET applications an additional wrapper is needed to call OpenCV functions. Therefore EmguCV[8] was used.

[4] http://msdn.microsoft.com/en-us/library/ms754130(v=vs.100)/
[5] http://greatmaps.codeplex.com/
[6] http://msdn.microsoft.com/en-us/library/ms731082(v=vs.110)/
[7] http://opencv.org/
[8] http://emgu.com/

To connect the whiteboard application with the ambient light, we used an ambient light controller, written in C#. This controller can address each of the LED lights individually. Therefore, the Adafruit Neopixel library[9,10] is used. For the hardware part, we used an Arduino Leonardo[11] microcontroller with the current 1.0.5 software.

[9] https://learn.adafruit.com/adafruit-neopixel-uberguide/overview/
[10] https://github.com/adafruit/Adafruit_NeoPixel/
[11] http://arduino.cc/en/Main/ArduinoBoardLeonardo/

Chapter 5

Discussion

To get deeper insights we conducted an interview and compared the two approaches for off-screen visual cues. We are presenting the results of this interview, ideas for further extensions and new approaches. Furthermore, this chapter discusses the limitations of the implemented solution. More specifically, issues concerning the presented off-screen visual cues and the head-detection.

5.1 Interviews

We have investigated in an explorative evaluation and a comparison of the implemented solution. This evaluation was done in the form of an interview with two experts. The collected results give us insights in benefits and drawbacks of this solution from a user's perspective. Additionally, these results indicated points of improvement and proposed directions for future work.

5.1.1 Design

For our interview, we invited two interaction design and usability experts to explore our implemented solution. After a short introduction in our test room, they got a tablet computer with the running application, while the connected digital whiteboard was showing a map of "Linz, Austria". After some time to freely test the system and it's possibilities, we conducted an interview. This interview and the following discussion were done with both interviewees together and one interviewer.

The focus of this interview was to find differences between the two off-screen visual cue approaches. To detect whether users are able to estimate distances, shown by the visual cues, we prompted the participants with examples of eight different distances in each case via ambient light or by rendering (Figure 5.1 and 5.2). We decided for four distances per approach, distributed from a minimum value of 13.98 to a maximum value of 60 km

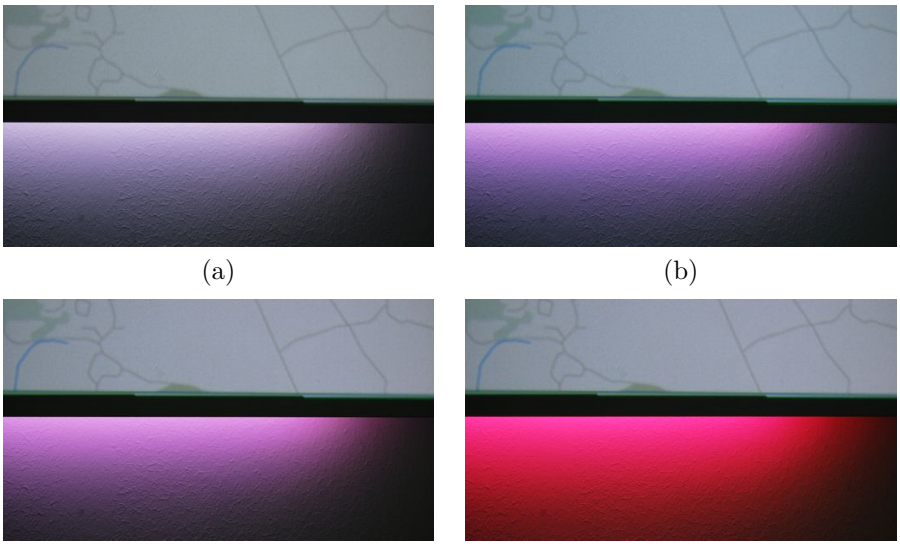


Figure 5.1: This figures show the ambient light indicators with different distances that we have shown to our participants. Here in ascending order: (a) with a distance of 13.98 km, (b) with a distance of 27.32 km, (c) with a distance of 41.78 km and (d) with a distance of 60 km. During the interview, these examples where shown in random order.

(Tables 5.1 and 5.2, Column 1). These visual cues were shown in random order to be independent on former assumptions. First, four distance cases with ambient light and second, the four rendered borderlines. As each participant had to estimate all eight cases, distances were different for the ambient light and the rendered approach. Interviewees were requested to write down their assumption of the shown distance on a piece of paper.

After this task, pros and cons, as well as new ideas, extensions and suggestions for improvement points were discussed extensively. Additionally, we collected ideas for the *overlap*-problem (Figure 5.8 and 5.9).

5.1.2 Results

The results of the conducted explorative interview show that participants' feedback about our solution in general, was positive. Our system supported them in orientation on the large whiteboard display. The easy and direct way of manipulation via the tablet's touch display was mentioned by the participants to be a benefit. Participants were sitting in front of the digital whiteboard, where they explored all functions from, and checked the triggered results immediately on the digital whiteboard.

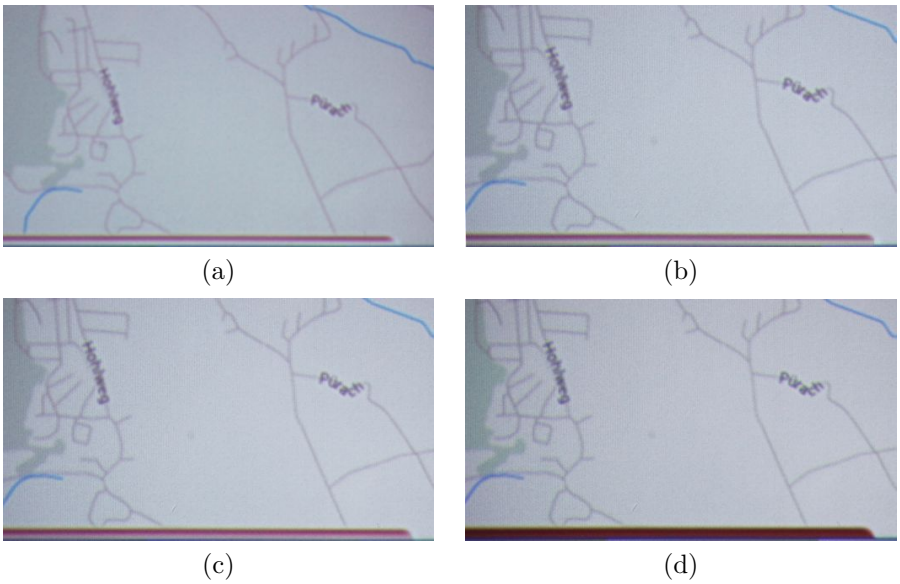


Figure 5.2: This figures show the rendered borderlines with different distances that we have shown to our participants during the interview. Here in ascending order: (a) with a distance of 15.64 km, (b) with a distance of 22.32 km, (c) with a distance of 47.34 km and (d) with a distance of 58.45 km. During the interview, these examples where shown in random order.

Comparing Off-Screen Visual Cues Asked for pros and cons of the two solutions, participants stated saving display space as the main benefit of the ambient light approach. However, they recognized a lower resolution compared to the rendered borderlines. Generally, ambient light impressed by its beauty but the rendering was declared to be the more suitable solution. Participants mentioned that borderlines, indicating high distances are hard to interpret, especially for visual cues by ambient light.

When the participants were prompted with example renderings and ambient light borderlines, they had to estimate the distances. The distances could be up to 60 km. Overall, participants performed better in the rendering task. While the ambient light guesses varied on average 12.85 km (Table 5.1), the rendered lines varied only 8.05 km (Table 5.2). Therefore, the resulting assumptions for the rendered solution were overall examples, 4.8 km more precise to the ambient light conditions. Eye-catching was one outlier in the rendering task, the participant answered 60 km, but the correct solution was 22.32 km (Table 5.2, Line 4, participant 2). The participant mentioned to have difficulties remembering the mapping of opacity to distance and mixed it up. We eliminated this distance case from the calculation and got a result of 3.63 km deviation from the actual solution, for the rendering task. This means, after the correction of the results, the participants'

Ambient Light			
shown distance	*participant 1*	*participant 2*	*deviation from correct solution*
27.32	35	40	10.18
60	45	50	16.51
41.78	55	60	15.72
13.98	5	5	8.98
			12.85

Table 5.1: The results of the distance guessing-task for the ambient light scenario, in tested order.

Rendering			
shown distance	*participant 1*	*participant 2*	*deviation from correct solution*
58.45	60	60	1.55
15.64	10	0	10.64
47.34	40	50	2.34
22.32	20	60	17.68
			8.05

Table 5.2: The results of the distance guessing-task for the rendered borderlines, in tested order.

estimations for the rendered approach were 9.22 km closer to the actual shown distances, compared to the ambient light solution. Besides the described outlier, all assumed values for the rendered solution were closer to the actual distance than within the ambient light conditions. Interestingly, for the rendered borderlines, results indicate that estimating high distances is much easier, compared to low distances. This is not true for the ambient light cases. The interviewees noted the mapping of opacity to distance on their sheet of paper for checking back during the distance-guessing task.

Interestingly, for the rendering cases, participants were able to rank the four shown distances in the correct order by their km assumptions. Again, this was more difficult for the ambient light cases. The two furthest away distances (41.78 and 60 km) got mixed up.

Overlapping Lines Another issue of the interview was to find a solution for overlapping borderlines, as it's the case when multiple visual cues are in the same position.

First, we asked the interviewees for their solution approaches. Participants stated, they would suppose the overlapping lines get stacked on each other. While discussing the inappropriateness of this solution for the ambient light, they came up with the idea of blending the lines' colors. Another idea for the ambient light scenario was to shift lines appropriately to coexist.

Next, we prompted participants with three solution approaches for the overlapping problem. These solutions where

- one line covers the other completely or partly (Figure 5.3 (a) and (b)),
- the lines' colors are blended (Figure 5.3 (c) and (d)), and
- the lines are stacked on each other (Figure 5.3 (e) and (f)).

For better imagination and to get a good impression for multiple scenarios, we showed all three options for two scenarios. One where the lines are side by side (Figures 5.3 (a), (c) and (e)) and one where the second line is fully included in the first one (Figures 5.3 (b), (d) and (f)).

For one participant, the length of the lines was confusing, he thought, that they are also indicating distance. The participant noted

> "...here (pointing to figure 5.3 (b)), I would guess that the blue line is further away, compared to the red one..."

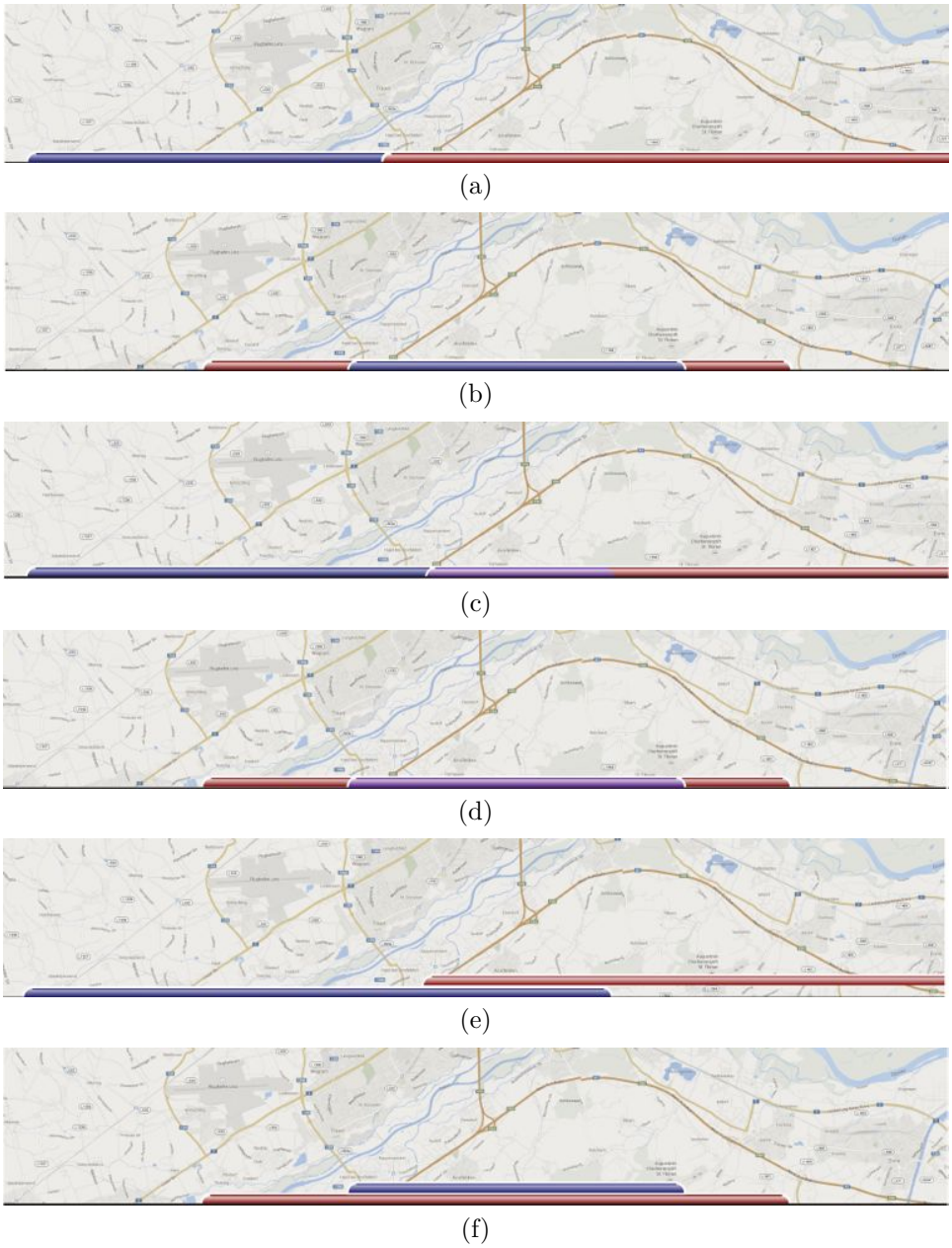
After discussing, the idea of blending borderlines (Figure 5.3 (a) and (b)) was rejected because the third color, developed by blending the two lines, was mentioned to be irritating.
One participant commented

> "...when my color is blue, in this case (pointing to figure 5.3 (b)), additionally to my color, I need to know and remember, that it can become purple..."

The only discovered drawback of the stacked solution was the potential of a space problem, depending on the number of stacked lines.

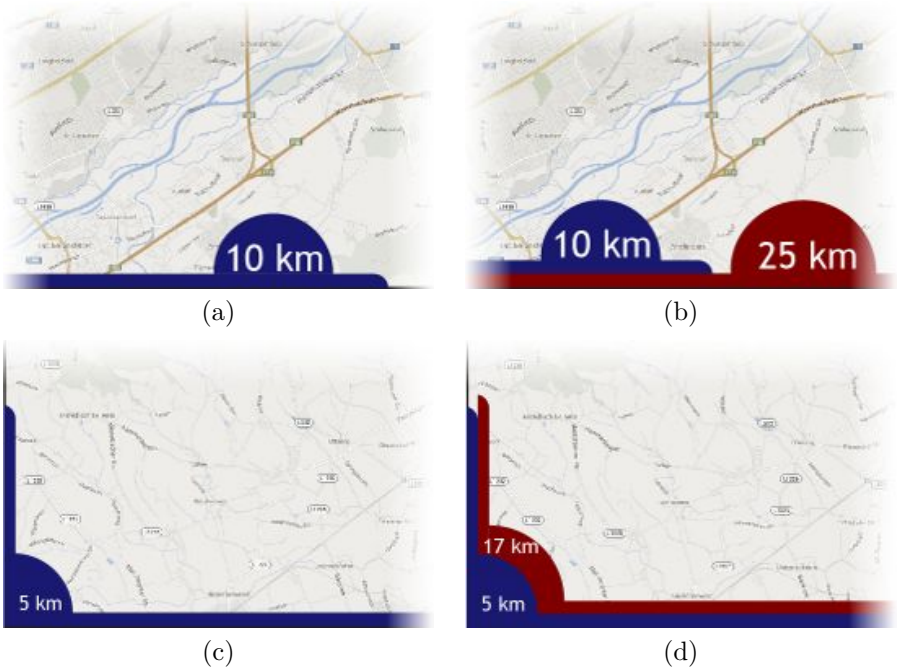
Extensions, Improvements and New Approaches Additionally, during the interview, multiple new ideas for visual cues came up. Participants began to draw and sketch several drafts. While the first sketches where more focused on improvements of our implemented borderline idea (Figure 5.4), also drafts of completely new approaches were discussed. Figure 5.6 shows illustrations for some of these ideas.

A completely different approach for displaying lines of multiple users in the same location is shown in figure 5.5, the symbol plus the number two should indicate, more then one users' view is displayed here. Furthermore, an arrow supports direction identification. However they decided, that this



(a) (b) (c) (d) (e) (f)

Figure 5.3: The currently implemented solution, where one line covers the other (a) and (b). A solution via blending the lines' colors (c) and (d). Stacked borderlines (e) and (f).



(a) (b)

(c) (d)

Figure 5.4: Illustrations of improvement ideas for the rendered borderlines. Lines can be extended with bubbles and a distance description. Figures (a) and (c) show this idea for one user. Figures (b) and (d) illustrate multi-user scenarios.

Figure 5.5: Multiple users in the same direction could also be indicated by a symbol and the number of users. An additional arrow should show the direction of the tablet's view in this case.

approach is rather confusing when two users are not in the exact same position, in this case, multiple arrows would be needed.

As shown in figure 5.4 (a) and (b), participants desired to just write down the distances and be therefore without the need of further interpretation. Other ideas were to extend the borderlines with additional patterns or icons,

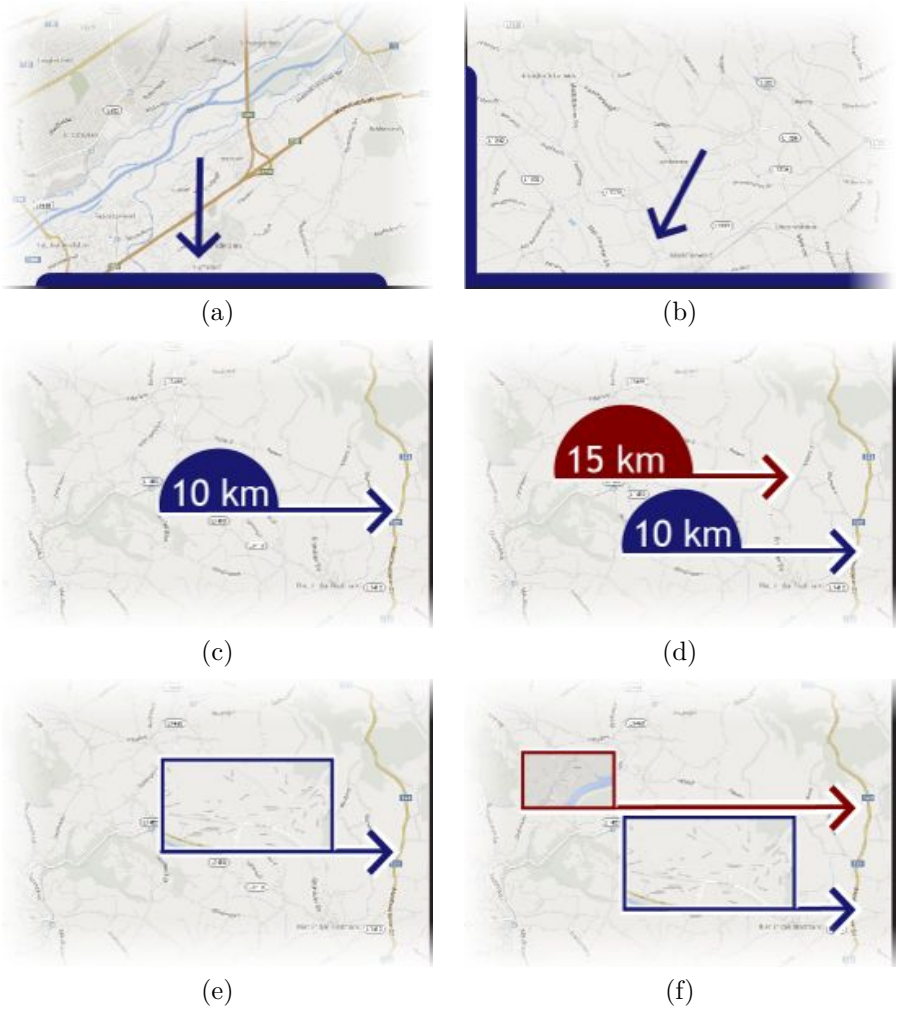


Figure 5.6: Illustrations for two different ideas based on direction arrows. Figures (a) and (b) extend the borderline with an additional arrow. While (a) shows this for an orthogonal shifted line, (b) shows it for a corner shift. Figures (c) and (d) show arrows plus distance indication, (e) and (f) show arrows plus a small window with the tablet's view. Figures (d) and (f) again, show these ideas for multi-user scenarios.

individual for every user. These solutions would be better to recognize, for all distances, compared to the current opacity solution.

Furthermore, the idea to use arrows, showing the direction of the tablet's view was discussed (Figure 5.6). Participants stated, that borderlines could be unfamiliar and therefore difficult to interpret. These lines and their translation need to be learned. While indicating directions via arrows is a natural and well-known process from devices like navigation systems and compasses.

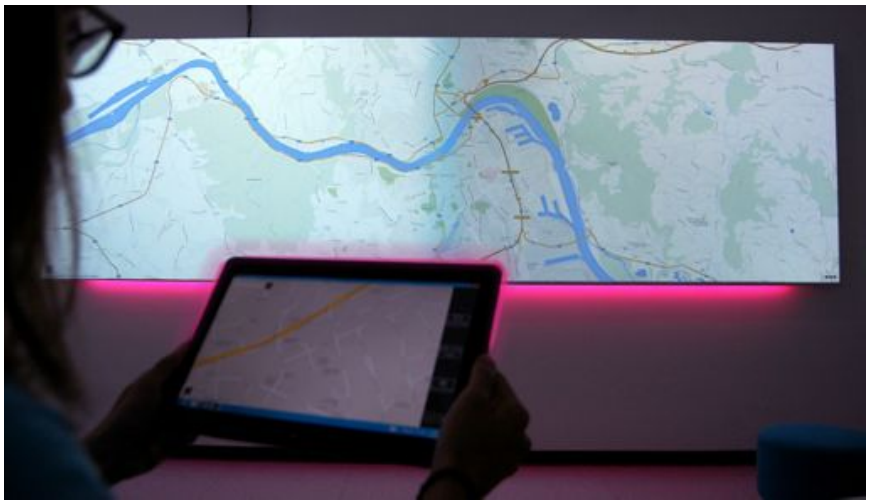


Figure 5.7: This figure illustrates the idea of indicating a user's color also on the tablet to easily assign users to devices.

Qualitative results also show that rendered borderlines should be thicker, one suggestion was double the current size. Concerning mapping of users to colors, it was mentioned that it would help to indicate a tablet's color with a colored rubber 'bumper' for the tablet or a LED on the device in the appropriate color (this idea is shown in figure 5.7). Moreover, a caption on the whiteboard display could show the mapping of users to colors. Participants stated, this does not need to be always visible but should be possible to show. Further ideas concerned additional information displays, like activity indicators (which user is the most active, who where last active in which area, etc.).

To sum our results up, participants agreed that a numerical distance indicator would be the better solution compared to the currently implemented approach, where distance is mapped to opacity. During the interview we also detected, that the mapping of light opacity to high distances and fully opaque appearance to low distances is not natural. Also if declared at first, participants mixed these indications up. As our participants just used the system for the first time, the question if mapping distances to opacity gets better by gaining some experience is still open.

5.2 Limitations

Limitations of the current approach are for example showing the visual cues of multiple off-screen devices on the same border and the mapping of distance to opacity. This affects both, the rendered, as well as the off-screen visual cues via ambient light.

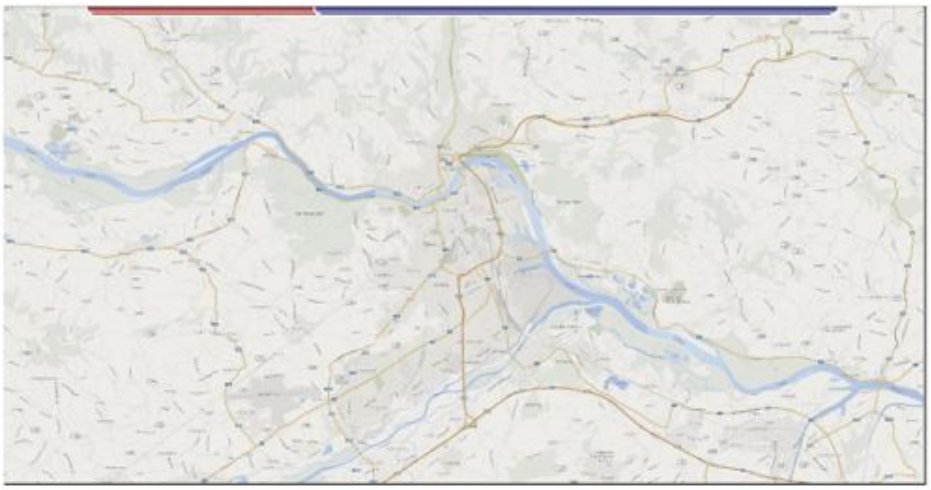


Figure 5.8: Multiple tablet views can be displayed simultaneously. However, they can overlap or even hide each other.

Basically, we designed this solution to support collaboration. Therefore, the application can handle simultaneous *Show View*-events of multiple devices. But when more than one borderline is shown, they can overlap or even hide another line completely (Figures 5.8 and 5.9). The overlap is dependent of:

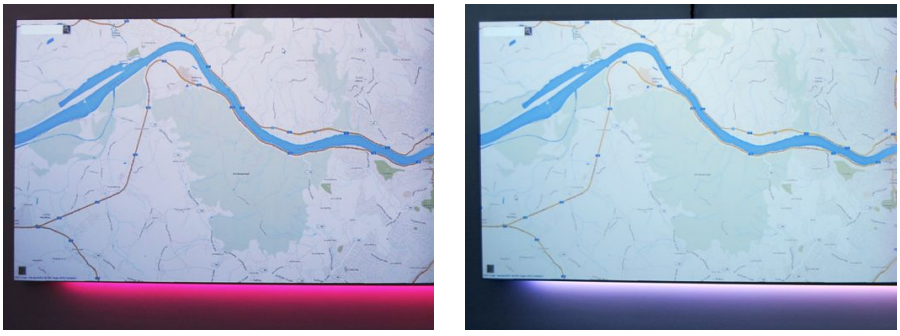
- the positions of the tablets' views,
- the current zoom levels of the tablets' views,
- the calculated distances and therefore the opacity of the borderlines.

The arrangement is dependent on the sequence of incoming events; the later event is on top. We also discussed this issue in the section above, during the interview. The results indicate that a solution where the rendered borderlines are stacked on each other would fit users expectations best.

As the distance of off-screen tablet views is indicated by opacity, this can lead to bright, white appearing ambient light. If multiple lights show high distances, it can become hard to differentiate and assign them to an individual user (Figure 5.10). Additionally, we discovered, the mapping of low distance to high opacity is not straightforward for everybody. This was also one of the discussed points during the evaluation of our approach 5.1.

Head-Detection The dependence on the current light condition of the room respectively, the contrast between head and background is one of the limitations of the head-detection.

Figure 5.9: When two borderlines overlap, the RGB values for the appropriate leds are overwritten by the device, which sends the *Show View*-event later.



(a) (b)

(c) (d)

Figure 5.10: The borderline in form of ambient light with a distance of about 14.5 km and the color red (a), with a distance of about 47 km and the color red (b), with a distance of 14.5 km and the color blue (c), with a distance of 47 km and the color blue (d). The figures (b) and (c) show, the further away the tablet's view gets, the less colored the light is. Therefore, it becomes hard to distinguish between the lines and assign them correctly to a user. This is especially true when two visual cues are indicating a similar distance and are positioned nearby or are overlapping.

Visual cues are always displayed when no head can be detected via the built-in front camera of the tablet device. Therefore, no matter where a user's focus currently is, as long as it cannot be detected on the tablet device, the visual cue appears on the whiteboard. This means, once the focus-detection is enabled, visual cues are displayed also if the user is not looking on the whiteboard display at all. Thus, visual cues can appear unintentionally.

Evaluation During the interview we got some interesting insights in the usability of the implemented solution. We could detect drawbacks and points for improvement. Additionally, also new approaches and extensions of the current solution were discussed. However, this interview was conducted with only two interviewees and on an explorative approach. There is still a lack of a full and structured evaluation of the system with a broad variety of potential users. As this system is designed for collaboration, a group evaluation with a specific use case could be interesting as well.

Chapter 6

Conclusion

In this thesis we presented an approach to support users' orientation on DUIs. The system is designed for a digital whiteboard and multiple tablet computers. This setup supports collaboration as well as individual work. Visual cues, displayed on a digital whiteboard enable users to easily detect where their current tablet view is located on this large display. We designed, implemented and evaluated this approach.

6.1 Contribution

A tablet's view can be on- or outside of the whiteboards view. Therefore, we implemented a solution for on- and off-screen visual cues. Both work independent of zoom level and map type. For supporting multi-user capabilities, we designed these visual cues to be distinguished by appearing in user-individual colors. The on-screen visual cues are animated and magnified for attention purpose. For the representation of off-screen tablet views, we investigated in two different approaches. First, a rendered borderline visual cue and second, an ambient light solution. Both options indicate distance via opacity and direction via their position along the edge of the whiteboard display.

Besides the possibility of displaying this orientation support by button click on the tablet device, the system can react on users' focus. When the focus shifts from the tablet, the appropriate visual cue appears on the digital whiteboard. The presented head-detection works calibration-free and without dwell time.

We conducted an interview to evaluate our solutions. The participants' general feedback was positive. While comparing the two approaches for off-screen visual cues, we detected benefits and drawbacks of both solutions. The rendered borderline benefits from easier detection of the indicated distance. Ambient light needs no screen space at all and is definitely the more eye-catching solution. Especially when it comes to the case where multi-

ple borderlines should appear on the same position, the rendered borderline benefits from it's changeability. During these interviews we additionally collected ideas to further improve and extend the system.

6.2 Future Work

This work is concentrated on supporting users' orientation on large displays, the presented visual cues are an approach. Finding a good balance between easy to spot visual cues and therefore attracting attention on the one hand but not distracting other users on the other hand is difficult. We presented two different approaches for off-screen representations, in the form of a rendering and via ambient light.

An explorative interview was designed and conducted as an initial study. This interview's main focus was to get an impression how rendered and ambient light off-screen visual cues are accepted by users. Furthermore, interview results were expected to point out areas requiring further investigations.

However, as the explorative interview, covered by this work can be seen as a preliminary study, there is definitely room for structured studies with users. Especially interesting to evaluate would be the collaboration of a group. This could also detect how good the balance between attracting attention and not distracting co-workers of our solution works.

As an answer to the problem when multiple borderlines are overlapping, we suggest the implementation of stacked lines. As the results of our interview detected, the distance indication by opacity could be extended with a numerical and text-based solution. Other approaches, like the suggested direction arrows could be suitable as well. Further research should be focused on advancement and comparison of these ideas.

As already mentioned in the limitations, focus-detection is based on the detection of user's face via the tablet's built-in front camera. A possible solution for this problem would be focus tracking and user identification from the whiteboard. With this solution, visual cues can be shown only when the focus is detected to be on the whiteboard display.

Appendix A

Content of CD-ROM/DVD

Format: CD-ROM, Single Layer, ISO9660-Format

A.1 Thesis

Path: /

Katrin Freihofner_Thesis.pdf Masterthesis

A.2 Miscellaneous

Path: /documents

images/ . . . . . images and illustrations

References

Literature

[1] Mark Altosaar et al. "AuraOrb: Using Social Awareness Cues in the Design of Progressive Notification Appliances". In: *Proceedings of the 18th Australia Conference on Computer-Human Interaction: Design: Activities, Artefacts and Environments*. OZCHI '06. Sydney, Australia: ACM, 2006, pp. 159–166. URL: http://doi.acm.org/10.1145/1228175.1228204.

[2] Till Ballendat, Nicolai Marquardt, and Saul Greenberg. "Proxemic Interaction: Designing for a Proximity and Orientation-aware Environment". In: *ACM International Conference on Interactive Tabletops and Surfaces*. ITS '10. Saarbrücken, Germany: ACM, 2010, pp. 121–130. URL: http://doi.acm.org/10.1145/1936652.1936676.

[3] Patrick Baudisch and Ruth Rosenholtz. "Halo: A Technique for Visualizing Off-screen Objects". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '03. Ft. Lauderdale, Florida, USA: ACM, 2003, pp. 481–488. URL: http://doi.acm.org/10.1145/642611.642695.

[4] Patrick Baudisch et al. "Keeping Things in Context: A Comparative Evaluation of Focus Plus Context Screens, Overviews, and Zooming". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '02. Minneapolis, Minnesota, USA: ACM, 2002, pp. 259–266. URL: http://doi.acm.org/10.1145/503376.503423.

[5] Wilhelm Burger and Mark J Burge. *Digital Image Processing: An Algorithmic Approach Using Java*. 1st ed. New York: Springer, 2008, p. 127.

[6] Stefano Burigat, Luca Chittaro, and Andrea Vianello. "Dynamic Visualization of Large Numbers of Off-screen Objects on Mobile Devices: An Experimental Comparison of Wedge and Overview+Detail". In: *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services*. MobileHCI '12. San

Francisco, California, USA: ACM, 2012, pp. 93–102. URL: http://doi.acm.org/10.1145/2371574.2371590.

[7] Connor Dickie et al. "LookPoint: An Evaluation of Eye Input for Hands-free Switching of Input Devices Between Multiple Computers". In: *Proceedings of the 18th Australia Conference on Computer-Human Interaction: Design: Activities, Artefacts and Environments*. OZCHI '06. Sydney, Australia: ACM, 2006, pp. 119–126. URL: http://doi.acm.org/10.1145/1228175.1228198.

[8] David Díez et al. "Sharing your view: A distributed user interface approach for reviewing emergency plans". In: *International Journal of Human-Computer Studies* 72.1 (2014), pp. 126–139. URL: http://www.sciencedirect.com/science/article/pii/S107158191300061X.

[9] Niklas Elmqvist. "Distributed User Interfaces: State of the Art". English. In: *Distributed User Interfaces*. Ed. by José A. Gallud, Ricardo Tesoriero, and Victor M.R. Penichet. Human-Computer Interaction Series. Springer London, 2011, pp. 1–12. URL: http://dx.doi.org/10.1007/978-1-4471-2271-5_1.

[10] Mathias Frisch and Raimund Dachselt. "Off-screen Visualization Techniques for Class Diagrams". In: *Proceedings of the 5th International Symposium on Software Visualization*. SOFTVIS '10. Salt Lake City, Utah, USA: ACM, 2010, pp. 163–172. URL: http://doi.acm.org/10.1145/1879211.1879236.

[11] Sohaib Ghani, Nathalie H. Riche, and Niklas Elmqvist. "Dynamic Insets for Context-aware Graph Navigation". In: *Proceedings of the 13th Eurographics / IEEE - VGTC Conference on Visualization*. EuroVis'11. Bergen, Norway: Eurographics Association, 2011, pp. 861–870. URL: http://dx.doi.org/10.1111/j.1467-8659.2011.01935.x.

[12] Tiago Gonçalves et al. "Comparison of Off-screen Visualization Techniques with Representation of Relevance on Mobile Devices". In: *Proceedings of the 27th International BCS Human Computer Interaction Conference*. BCS-HCI '13. London, UK: British Computer Society, 2013, 9:1–9:10. URL: http://dl.acm.org/citation.cfm?id=2578048.2578061.

[13] Saul Greenberg, Michael Boyle, and Jason Laberge. "PDAs and shared public displays: Making personal information public, and public information personal". In: *Personal Technologies* 3.1-2 (1999), pp. 54–64. URL: http://dx.doi.org/10.1007/BF01305320.

[14] Sean Gustafson et al. "Wedge: Clutter-free Visualization of Off-screen Locations". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '08. Florence, Italy: ACM, 2008, pp. 787–796. URL: http://doi.acm.org/10.1145/1357054.1357179.

[15] Kasper Hornbæk, Benjamin B. Bederson, and Catherine Plaisant. "Navigation Patterns and Usability of Zoomable User Interfaces with and Without an Overview". In: *ACM Transactions on Computer-Human Interaction* 9.4 (Dec. 2002), pp. 362–389. URL: http://doi.acm.org/10.1145/586081.586086.

[16] Zahid Hossain et al. "EdgeSplit: Facilitating the Selection of Off-screen Objects". In: *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services*. MobileHCI '12. San Francisco, California, USA: ACM, 2012, pp. 79–82. URL: http://doi.acm.org/10.1145/2371574.2371588.

[17] Alexandra Ion. "Uncovering Moving Off-View Objects on Large Interface Displays". Hagenberg, Austria: Interactive Media; FH Oberösterreich – Fakultät für Informatik, Kommunikation und Medien, 2012.

[18] Alexandra Ion et al. "Canyon: Providing Location Awareness of Multiple Moving Objects in a Detail View on Large Displays". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '13. Paris, France: ACM, 2013, pp. 3149–3158. URL: http://doi.acm.org/10.1145/2470654.2466431.

[19] Robert J. K. Jacob. "The Use of Eye Movements in Human-computer Interaction Techniques: What You Look at is What You Get". In: *ACM Transactions on Information Systems* 9.2 (Apr. 1991), pp. 152–169. URL: http://doi.acm.org/10.1145/123078.128728.

[20] Brett R. Jones et al. "IllumiRoom: Peripheral Projected Illusions for Interactive Experiences". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '13. Paris, France: ACM, 2013, pp. 869–878. URL: http://doi.acm.org/10.1145/2470654.2466112.

[21] Azam Khan et al. "Spotlight: Directing Users' Attention on Large Displays". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '05. Portland, Oregon, USA: ACM, 2005, pp. 791–798. URL: http://doi.acm.org/10.1145/1054972.1055082.

[22] Manu Kumar, Andreas Paepcke, and Terry Winograd. "EyePoint: Practical Pointing and Selection Using Gaze and Keyboard". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '07. San Jose, California, USA: ACM, 2007, pp. 421–430. URL: http://doi.acm.org/10.1145/1240624.1240692.

[23] Paul P. Maglio et al. "Gaze and Speech in Attentive User Interfaces". In: *Proceedings of the Third International Conference on Advances in Multimodal Interfaces*. ICMI '00. Beijing, China: Springer-Verlag, 2000, pp. 1–7. URL: http://dl.acm.org/citation.cfm?id=645524.656806.

[24] Heiko Müller et al. "An Ambient Light Display for Dynamic OffScreen Points of Interest". To be published at NodiCHI 2014.

[25] Brad A. Myers, Herb Stiel, and Robert Gargiulo. "Collaboration Using Multiple PDAs Connected to a PC". In: *Proceedings of the 1998 ACM Conference on Computer Supported Cooperative Work*. CSCW '98. Seattle, Washington, USA: ACM, 1998, pp. 285–294. URL: http://doi.acm.org/10.1145/289444.289503.

[26] Alex Olwal. "Augmenting Surface Interaction through Context-Sensitive Mobile Devices". In: *Human-Computer Interaction – INTERACT 2009*. Ed. by Tom Gross et al. Vol. 5727. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2009, pp. 336–339. URL: http://dx.doi.org/10.1007/978-3-642-03658-3_39.

[27] Harry E. Petersen and Doris J. Dugas. "The Relative Importance of Contrast and Motion in Visual Detection". In: *Human Factors: The Journal of the Human Factors and Ergonomics Society* (1972), pp. 207–216. URL: http://hfs.sagepub.com/content/14/3/207.short.

[28] Emmanuel Pietriga, Caroline Appert, and Michel Beaudouin-Lafon. "Pointing and Beyond: An Operationalization and Preliminary Evaluation of Multi-scale Searching". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '07. San Jose, California, USA: ACM, 2007, pp. 1215–1224. URL: http://doi.acm.org/10.1145/1240624.1240808.

[29] Qian Qin, Michael Rohs, and Sven Kratz. "Dynamic Ambient Lighting for Mobile Devices". In: *Proceedings of the 24th Annual ACM Symposium Adjunct on User Interface Software and Technology*. UIST '11 Adjunct. Santa Barbara, California, USA: ACM, 2011, pp. 51–52. URL: http://doi.acm.org/10.1145/2046396.2046418.

[30] Umar Rashid, Miguel A. Nacenta, and Aaron Quigley. "Factors Influencing Visual Attention Switch in Multi-display User Interfaces: A Survey". In: *Proceedings of the 2012 International Symposium on Pervasive Displays*. PerDis '12. Porto, Portugal: ACM, 2012, 1:1–1:6. URL: http://doi.acm.org/10.1145/2307798.2307799.

[31] Umar Rashid, Miguel A. Nacenta, and Aaron Quigley. "The Cost of Display Switching: A Comparison of Mobile, Large Display and Hybrid UI Configurations". In: *Proceedings of the International Working Conference on Advanced Visual Interfaces*. AVI '12. Capri Island, Italy: ACM, 2012, pp. 99–106. URL: http://doi.acm.org/10.1145/2254556.2254577.

[32] Jun Rekimoto. "A Multiple Device Approach for Supporting Whiteboard-based Interactions". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '98. Los Angeles, California, USA: ACM Press/Addison-Wesley Publishing Co., 1998, pp. 344–351. URL: http://dx.doi.org/10.1145/274644.274692.

[33] Mikkel Rønne Jakobsen and Kasper Hornbæk. "Sizing Up Visualizations: Effects of Display Size in Focus+Context, Overview+Detail, and Zooming Interfaces". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '11. Vancouver, BC, Canada: ACM, 2011, pp. 1451–1460. URL: http://doi.acm.org/10.1145/1978942.1979156.

[34] Johan Sanneblad and Lars Erik Holmquist. "Ubiquitous Graphics: Combining Hand-held and Wall-size Displays to Interact with Large Images". In: *Proceedings of the Working Conference on Advanced Visual Interfaces*. AVI '06. Venezia, Italy: ACM, 2006, pp. 373–377. URL: http://doi.acm.org/10.1145/1133265.1133343.

[35] Jeffrey S. Shell, Roel Vertegaal, and Alexander W. Skaburskis. "EyePliances: Attention-seeking Devices That Respond to Visual Attention". In: *CHI '03 Extended Abstracts on Human Factors in Computing Systems*. CHI EA '03. Ft. Lauderdale, Florida, USA: ACM, 2003, pp. 770–771. URL: http://doi.acm.org/10.1145/765891.765981.

[36] John D. Smith, Roel Vertegaal, and Changuk Sohn. "ViewPointer: Lightweight Calibration-free Eye Tracking for Ubiquitous Handsfree Deixis". In: *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology*. UIST '05. Seattle, WA, USA: ACM, 2005, pp. 53–61. URL: http://doi.acm.org/10.1145/1095034.1095043.

[37] Sophie Stellmach and Raimund Dachselt. "Look & Touch: Gaze-supported Target Acquisition". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '12. Austin, Texas, USA: ACM, 2012, pp. 2981–2990. URL: http://doi.acm.org/10.1145/2207676.2208709.

[38] Sophie Stellmach et al. "Designing Gaze-supported Multimodal Interactions for the Exploration of Large Image Collections". In: *Proceedings of the 1st Conference on Novel Gaze-Controlled Applications*. NGCA '11. Karlskrona, Sweden: ACM, 2011, pp. 1–8. URL: http://doi.acm.org/10.1145/1983302.1983303.

[39] Rainer Stiefelhagen, Jie Yang, and Alex Waibel. "Estimating Focus of Attention Based on Gaze and Sound". In: *Proceedings of the 2001 Workshop on Perceptive User Interfaces*. PUI '01. Orlando, Florida: ACM, 2001, pp. 1–9. URL: http://doi.acm.org/10.1145/971478.971505.

[40] Jayson Turner. "Cross-device Eye-based Interaction". In: *Proceedings of the Adjunct Publication of the 26th Annual ACM Symposium on User Interface Software and Technology*. UIST '13 Adjunct. St. Andrews, Scotland, United Kingdom: ACM, 2013, pp. 37–40. URL: http://doi.acm.org/10.1145/2508468.2508471.

[41] Jayson Turner et al. “Eye Pull, Eye Push: Moving Objects between Large Screens and Personal Devices with Gaze and Touch”. In: *Human-Computer Interaction – INTERACT 2013*. Ed. by Paula Kotzé et al. Vol. 8118. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013, pp. 170–186. URL: http://dx.doi.org/10.1007/978-3-642-40480-1_11.

[42] Roel Vertegaal et al. “Eye Gaze Patterns in Conversations: There is More to Conversational Agents Than Meets the Eyes”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI ’01. Seattle, Washington, USA: ACM, 2001, pp. 301–308. URL: http://doi.acm.org/10.1145/365024.365119.

[43] Roel Vertegaal et al. “Media Eyepliances: Using Eye Tracking for Remote Control Focus Selection of Appliances”. In: *CHI ’05 Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’05. Portland, OR, USA: ACM, 2005, pp. 1861–1864. URL: http://doi.acm.org/10.1145/1056808.1057041.

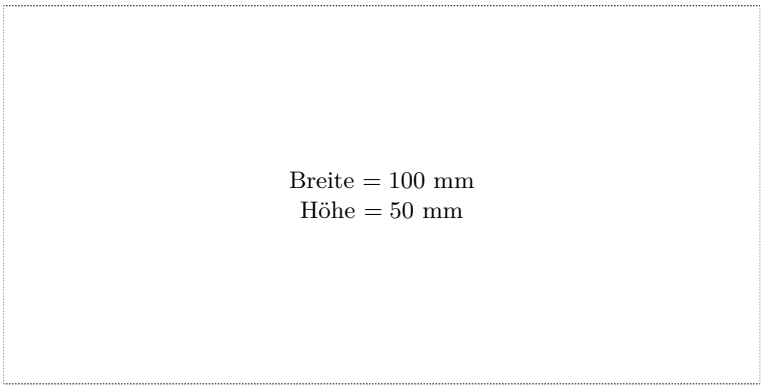
[44] Colin Ware. *Information Visualization*. 2nd ed. San Francisco: Elsevier Inc., 2004, pp. 146–147.

[45] Martin Weigel et al. “From Focus to Context and Back: Combining Mobile Projectors and Stationary Displays”. In: *Proceedings of the 4th annual digital media conference*. Toronto, Ontario, Canada, 2013. URL: http://grouplab.cpsc.ucalgary.ca/grouplab/uploads/Publications/Publications/2012-FocusContextProjectors.Report2012-1031-14.pdf.

[46] Polle T. Zellweger et al. “City Lights: Contextual Views in Minimal Space”. In: *CHI ’03 Extended Abstracts on Human Factors in Computing Systems*. CHI EA ’03. Ft. Lauderdale, Florida, USA: ACM, 2003, pp. 838–839. URL: http://doi.acm.org/10.1145/765891.766022.

Messbox zur Druckkontrolle

— Druckgröße kontrollieren! —



— Diese Seite nach dem Druck entfernen! —