

The S3D-UI Designer – Creating User Interface Prototypes for 3D Displays

Nora Broy^{1,2}, Verena Lindner¹, Florian Alt¹

¹LMU Munich
Media Informatics Group
Amalienstraße 17
80333 Munich, Germany
firstname.lastname@ifi.lmu.de

²BMW Group
Research & Technology
Hanauer Straße 46
80992 Munich, Germany
Nora.NB.Broy@bmw.de

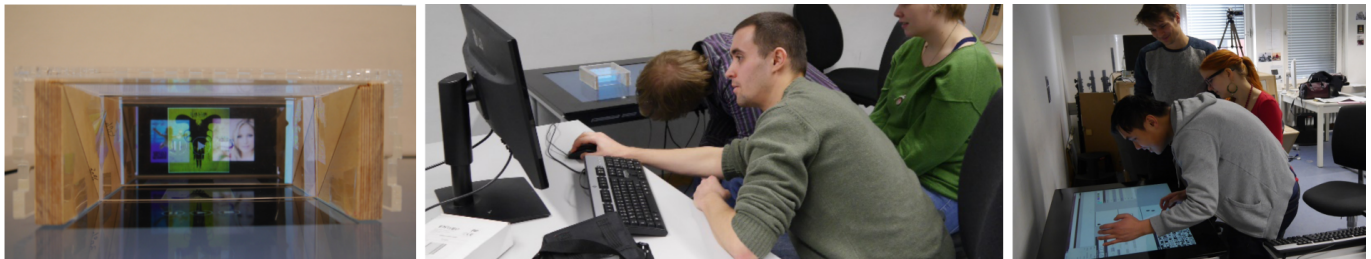


Figure 1: We present the S3D-UI Designer, a tool for stereoscopic 3D (S3D) user interface prototyping. A 2D user interface allows multiple depth layers to be easily and quickly designed. For the visualization we use the MirrorBox, a 3D output device that comprises three virtual depth layers. We evaluated our toolset with 26 participants that built 3D UI mockups (left) using keyboard/mouse (middle) and a touch surface (right) as input devices.

ABSTRACT

In this paper, we present the S3D-UI Designer – a tool to create prototypes for 3D displays. Stereoscopic (S3D) displays are quickly becoming popular beyond cinemas and home entertainments. S3D displays can today already be found in mobile phones, public displays, and car dashboards. A benefit of such UIs is that they can group and structure information in a way that makes them easily perceivable for the user. At the same time, prototyping these UIs is challenging, as with traditional techniques, UI elements cannot easily be rendered and positioned in 3D space. In contrast to professional 3D authoring tools, we present a tool targeted towards non-experts to quickly and easily sketch a S3D UI and instantly render it on a 3D display. We report on the design of the tool by means of a workshop and present an evaluation study with 26 participants assessing its usability.

Author Keywords

Prototyping; Stereoscopic 3D; User Interfaces

ACM Classification Keywords

H.5.2 User Interfaces: Prototyping

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.

MUM '16, December 12 – 15, 2016, Rovaniemi, Finland
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-4860-7/16/12...\$15.00
DOI: <http://dx.doi.org/10.1145/3012709.3012727>

INTRODUCTION

In recent years, stereo-vision (S3D) has become commonplace in entertainment, being mainly used in cinemas and home television. While in those settings content is static, we see large potential in interactive 3D user interfaces (UIs). The trend towards such UIs is supported by autostereoscopic screens rapidly proliferating in consumer devices. Such screens make wearing glasses obsolete for the user, resulting in a significant gain in comfort. State-of-the-art examples include mobile phones, public displays, as well as cars.

The benefit of using stereoscopic UIs compared to traditional 2D UIs lies in the opportunity, to (a) spatially group elements that relate to each other (e.g., apps or items frequently used or launched after each other), and to (b) present important information more prominently (e.g., further at the front). The latter case is particularly interesting for application areas, where distraction should be minimized as, for example, in cars.

Creating 3D UIs is today possible, using authoring tools such as Unity, Maya, Blender, or 3D Studio Max. While these tools offer powerful means for creating, positioning, and animating objects, they at the same time raise a number of challenges. First, they were never designed to create UIs. In particular, they do not offer any support for creating a sophisticated S3D UI, for example through notifying the designer about overlapping elements, too large depth budgets, or elements positioned too close to each other in order to distinguish their depth. Second, such tools are for expert designers and require trainings or prior knowledge for an efficient usage. Hence, they do not support the rapid prototyping of UIs as is often required in early phases of the design process.

At the focus of this paper we tackle these challenges and provide an easy-to-use tool that guides the designer towards prototyping a powerful and usable S3D interface. We draw upon a concept from the 1990s – *layered user interfaces* [?] – that recently received attention in the context of eye tracking in 3D [?]. Such interfaces present information on different depth layers. According to [?], such information structures support the user in focusing on a single UI object and in dividing attention between multiple objects. Prior work has shown that structuring information on layers can significantly improve search times [?]. This is also interesting from a designer perspective, since the use of layers makes it easier to create the UI, i.e., a designer can create the layers one-by-one. At the same time, it is difficult to understand how the layers interact with each other (i.e., obstructions, interplay of colors, etc.).

In this work, we present the S3D-UI Designer, a tool that allows several depth layers to be prototyped while the designer can instantly observe the final output on a 3D display. At the same time, the tool implements a number of features that support the designer already in early stages of the design process.

Our contribution is threefold. First, we report on two workshops, where designers had to create layered 3D UIs with common design tools. Second, based on the findings, we implemented a web-based prototyping tool. Third, we evaluated the tool with 26 participants and show that it strongly supports designers in creating prototypes of layered 3D UIs.

RELATED WORK

Prototyping graphical UI is important in the design process of building usable systems. It is applicable at different stages, ranging from early sketching on paper to an almost implemented product [?]. When sketching on paper, prototyping can even be done without any programming knowledge. It is used during brainstorming sessions or for designing, creating, and testing UI [?] and offers a way of rapidly presenting and communicating ideas [?]. But people not used to sketching may be afraid of drawing ugly interfaces and, hence, be reluctant to express themselves [?]. Sefelin et al. [?] show that test participants prefer digital prototypes over paper prototyping for early evaluations. Walker et al. [?] found that the participants commented more frequently on computer than paper prototypes while the paper as well as the digital prototypes identified the same number of usability issues.

However, creating digital prototypes typically requires programming skills and technological expertise when novel input or output technologies are intended. Possible solutions are tools that allow digital prototypes to be rapidly created without the need of expert knowledge. Such tools have been developed for a variety of application areas. Weigel et al. [?] developed a tool which allows creating prototypes for mobile projection applications. Moreover, there are tools that enable the creation of context aware applications. For example, *iCAP* [?] is a tool for prototyping context-aware applications based on if-then rules without writing any code. *Topiary* [?] allows for designing applications taking the location of people or places into account. Additionally, there are tools which address the use of hardware elements. For example, the *CalderToolkit* developed by Lee et al. [?] is a hardware toolkit

to rapidly prototype functional interactive devices for tangible UI. With regard to the automotive context, Schneegass et al. [?] presented *MI-AUI* that allows the simple creation of in-car UI using tangible controls. Broy et al. presented 3D-HUDD, a prototyping tool for 3D head-up displays [?]. Moreover, prototyping tools can also integrate several fidelity levels in one interface [?]. De Sá [?] developed a prototyping tool for mobile devices which supports the creation of prototypes ranging from sketch-based to functional interactive software prototypes.

There are several commercial tools, such as Balsamiq¹ and Axure² which support the creation of detailed UI prototypes for websites and mobile applications. In addition, a lot of developers use tools such as Microsoft PowerPoint³ and Adobe Photoshop⁴ to create simple wireframes but also sophisticated UI designs [?]. Presente3D⁵ is an extension for PowerPoint that enables the creation of stereoscopic presentation slides. However, to our knowledge there is no tool that supports prototyping S3D UI per se.

Prototyping S3D UIs has so far only been explored by Broy et al. using two physical tools that allow arranging content painted on foil in 3D space [?]. As [?] focus on paper prototyping, it does not address the integration of digital content. With our work we bridge this gap, i.e., allowing digital S3D prototypes to be created, while maintaining the ease of use of techniques that are well-known for 2D UIs.

PROTOTYPING USER INTERFACES FOR 3D DISPLAYS

As shown in related work, few solutions exist that optimally support prototyping for S3D displays. We focus on layered UIs as one particular use case. Thus, we can apply well-known 2D interaction techniques and avoid complex 3D interaction which are considered to decrease the usability [?] and pragmatic quality of the system [?]. We envision our research to provide fundamental knowledge upon which also non-layered UIs can draw in the future.

The basic idea of the S3D-UI Designer is a 2D UI that allows several depth layers to be designed and the depth layout to be instantly shown on a 3D output device. This interface provides input controls as well as a 2D display, which is referred to as the *input display* in the remainder of this paper. The 3D display is referred to as *output display* and represents the stereoscopic prototype. While in theory the number of possible layers is only limited by the employed technology – solutions range from autostereoscopic displays [?] to multilayer displays with several layers of transparent screens [?] – prior work recommends a maximum of 6 layers when structuring information on different layers [?]. We chose a technology supporting a limited number of layers only. To render the 3D output, we used a MirrorBox⁶. Tilted mirrors allow UI element projected from below to be seen. We positioned the

¹<https://balsamiq.com/>

²<http://www.axure.com/>

³<https://products.office.com/en-US/powerpoint?omkt=en-US>

⁴<http://www.adobe.com/products/photoshop.html>

⁵<http://www.presente3d.com/>

⁶We designed the MirrorBox following the instructions in [?].



Figure 2: Design Workshop – After the two groups of participants received a brief introduction to 3D prototyping, they had to design a hierarchical menu respectively an email application for a mobile device with a S3D display. After each task, groups presented their results. The workshop concluded with the groups filling in a questionnaire and engaging into a discussion.

MirrorBox on the display of a tabletop to reflect three virtual layers (Figure 2). We deliberately opted for this setup for several reasons. Besides offering a seamless process from paper-based to computer-based prototyping and even mixing both mediums, we reduce the complexity of layered UI by limiting it to three layers. Nevertheless, we envision a dynamic layout of several layers as a further refinement of the tool. Moreover, the MirrorBox neither requires glasses nor has technical shortcomings as autostereoscopic technologies (e.g., crosstalk, maintaining a defined sweet spot). It clearly differentiates the layers and, thus, eases the understanding of layered 3D UI and lowers the entry level for designers.

DESIGN WORKSHOPS

To identify core requirements for a 3D prototyping tool we ran two design workshops. The objective of the workshops was to identify important features and functionalities that would best support designers during their work. In these workshops, participants solved different design tasks with different objectives and by means of different 2D tools. The MirrorBox was positioned on top of a Microsoft PixelSense as 3D output display (Figure 2). For creating the UI, we attached an external monitor with keyboard and mouse.

To cover a wide range of features, we selected Microsoft Powerpoint and Adobe Photoshop for creating the UIs. Whereas Powerpoint allows items to be arranged quickly and predefined elements to be used, Photoshop provides a higher degree of artistic freedom when it comes to creating custom elements. In both programs we provided a template consisting of three distinct areas placed on top of each other. However, both tools required further interaction steps to display the content with the MirrorBox. The Photoshop version required to save the file, while the Powerpoint solution needed to mirror the layers and aligning the mirrored presentation.

Prototyping Tasks

The workshop was framed around creating a layered UI for an autostereoscopic smartphone. The participants had to solve two tasks. In the *menu task*, they had to design an hierarchical menu consisting of different applications (call, sms, email, etc.). In the *email app task*, they should create a layered UI for an email app. In addition, we let the participants design the UIs with two objectives. In the *free design condition*, we provided the task in text form, leaving participants the freedom to come up with an own solution. In the *mockup condition*, participants were presented a 2D mockup of the UI we expected them to rebuild for a 3D UI.

Setup, Participants, and Procedure

We recruited 7 participants (4 *f*, 3 *m*) with backgrounds in design and computer science. Participants were undergraduates and postgraduates. We divided the participants into two groups. The first group started with freely designing the menu (Powerpoint) before designing the email client based on the provided mockup (Photoshop). The second group was asked to first freely design the email client (Photoshop) before continuing with realising the mockup menu (Powerpoint).

Upon arriving at the lab, participants received a short introduction to the main principles of designing layered UIs [?]. Then, we explained the setup and how to use the prototyping tools. They started with the *free design task*. After 20 minutes they briefly presented their results before continuing with the *mockup task*. This time they used the program they had not yet worked with and had 10 additional minutes to finish the task. Again they gave a short presentation of their results. The workshop concluded with a 30-minute *discussion* phase, which allowed participants to come up with further ideas for the prototyping tool and to discuss them with the group.

Findings

Observations from the workshop and qualitative feedback from the discussions lead to four major findings.

Maintain opportunity to collaboratively prototype the user interface. Participants liked to collaboratively work on the user interface and instantly discussed the results, rather than designing elements separately. Our setup allowed only one person to create and arrange elements at a time while the others observed, commented, and provided suggestions. Being asked later whether this was an issue and whether it would have been better to provide every participant a device to work in parallel, all participants agreed that this would have destroyed the collaboration resulting in weaker outcomes.

Realtime feedback on spatial arrangements. Participants criticized the interaction steps necessary to view the 3D result inside the MirrorBox. In consequence, the instant synchronization of the input and output display is essential. A further weakness identified by the participants was that the used tools did not provide feedback on whether their choice of arranging UI elements would lead to a satisfactory result. In many cases, participants found placing elements to result in overlaps with objects already placed on other layers. In these cases, they needed to rework the interface. Hence, they requested features such as reference lines to align content across layers and automated feedback on overlapping objects.

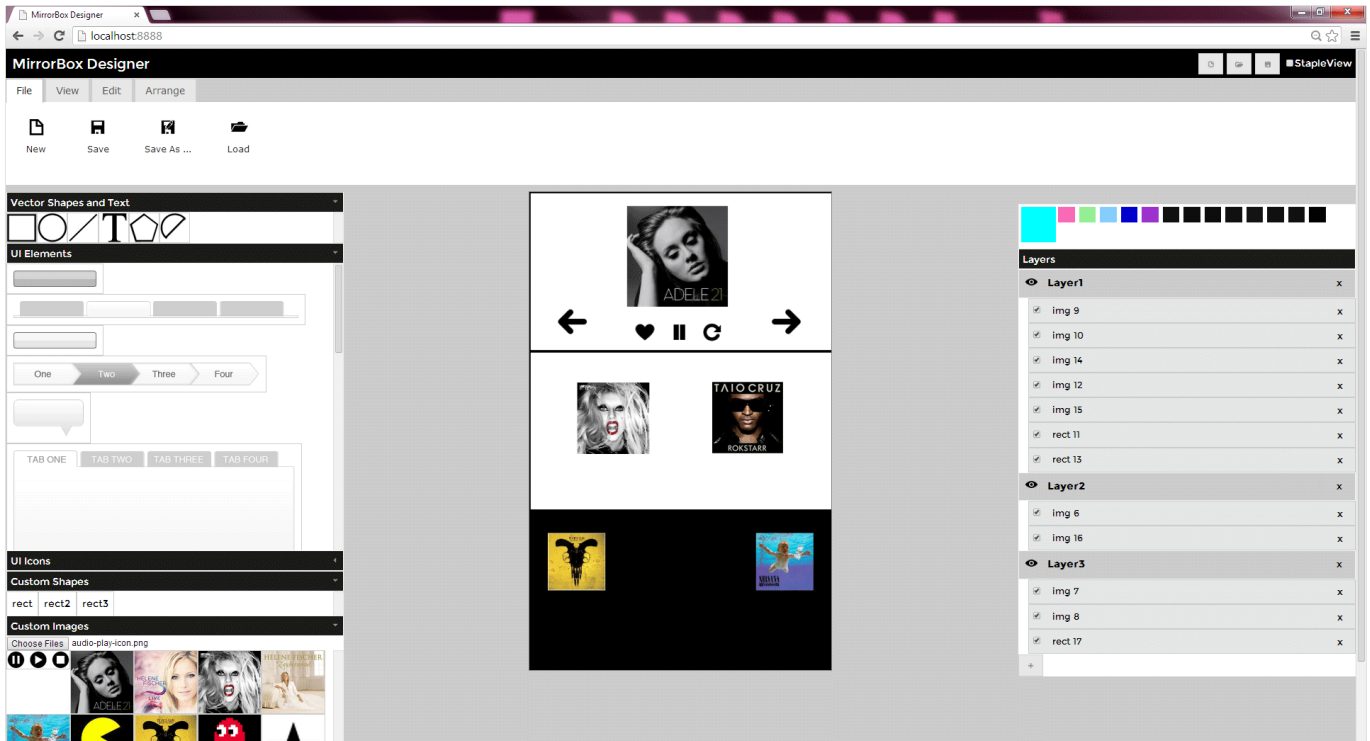


Figure 3: The S3D-UI Designer allows the visual layout of the three virtual layers to be created and provides different 2D views on these layers.

Enabling several viewing options for layers. The participants envisioned to switch the arrangement of the layers visualized by the input display. In our setup, the layers were stacked on top of each other. Beside displaying the layers individually and changing their order, the tool should provide a combined view. Rather than rendering each layer on separate display spaces, the combined view should render all layers on one display space to facilitate the transition for the users between the input display and the 3D output display.

Focus on depth layout features. Participants were sometimes overwhelmed and frustrated by the number of features and controls – in particular those provided by Photoshop. This suggests that the most important design functions should be offered instead of including complex drawing and shader methods. As a conclusion, the tool should focus on arranging GUI items on the layers while their detailed design can be achieved with other programs such as Photoshop.

THE S3D USER INTERFACE DESIGNER

The findings from the workshops served as a basis to inform the design of the prototyping tool (cf., Figure 3). We implemented the tool as a client-server based web application for use on different input devices, such as desktop environments, tablets, or tabletops. Data is stored on a server to be accessible everywhere. We also implemented an XML-based data format to exchange the generated 3D layouts and to extend the tool for other output devices.

Implementation

The S3D-UI Designer consists of a client-server architecture and implements one server and several clients (cf., Figure 4).

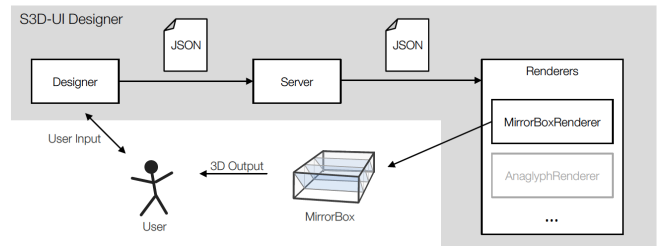


Figure 4: The S3D-UI Designer is based on a client-server architecture.

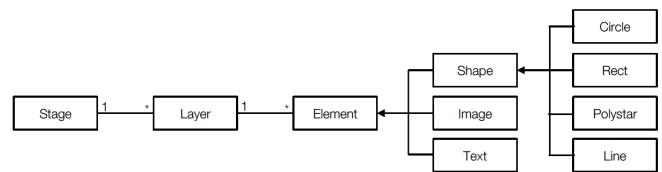


Figure 5: The data model represents a layered UI.

One client represents the *Designer* and the other clients are the *Renderers* enabling the output for different 3D display devices. As a first step, we implemented the *MirrorBoxRenderer*. The *Designer* implements the 2D GUI that allows for designing several depth layers.

The depth layers that are created using the *Designer* are parsed into a JSON model representing the layered UI. The data model is outlined in Figure 5. The so-called stage can contain multiple layers. Each layer consists of multiple children. Those children can be vector shapes, images, and text.

The JSON format allows the data to be exchanged between clients and server, the permanent storage of created UI, and the development of different *Renderers* in the future (e.g., for an anaglyph stereoscopic output). We implemented the data model in JavaScript to display it with the *Designer* and the *MirrorBoxRenderer*.

Since our setting just needs a one-way communication, the *Designer* sends the JSON file to the server which pushes the data to all *Renderers* via UDP. The server is based on *Node.js*. To enable the Node server to send the data to the *MirrorBoxRenderer* we use socket.io. The *Renderer* parses the data and renders the layers in accordance with the 3D output device. In our case, the *MirrorBoxRenderer* scales and arranges the three layers by means of the layout of the MirrorBox.

Graphical User Interface

The *Designer* (cf., Figure 3) consists of four areas. On top, a tab sheet allows toggling between four tabs (i.e., file, edit, view, and arrange). The file tab provides means to load and save files. From the content area placed on the left side, UI elements can be chosen. In particular, we offer, vector-shapes (UI elements, icons), images, and text. Images can also be loaded from the local file system which allows the integration of elements that were created with other tools. The edit tab allows the user to modify the properties (e.g., position, scale, rotation, color) of the UI elements. In the center, the depth layers are depicted. Note that in theory an unlimited number of layers could be created. Based on the findings from the workshop, we provide means to switch between two views with the view tab: one shows the layers stacked on top of each other and one depicts a combined view. In addition, the tool checks for overlapping UI elements and highlights colliding items. The user can add horizontal and vertical reference lines that are displayed across all layers. Finally, the right area shows the different layers and their elements. Using the arrange tab, elements can be grouped and ungrouped. The entire UI is drag and drop enabled, hence allowing for an easy placement and arrangement of the different elements.

EVALUATION

To evaluate the usability of the tool, we conducted a user study. In particular, we were interested how the tool as well as the working environment support collaboration in exploring UI ideas. Hence, the participants designed the layers on a touch surface of a Microsoft PixelSense as well as using a classical desktop setting with a desktop monitor, a keyboard, and a mouse. The study was designed as a repeated measures experiment with the working environment as independent variable (touch vs. desktop).

Setup, Participants, and Procedure

As in the previous workshops, participants received two tasks. The tasks were formulated in an open manner, leaving participants more room to explore the features of the tool. In both setups, participants designed prototypes for a multilayer mobile phone. In the *application task* participants created the UI for a music application. In the *game task* their assignment was to adapt the popular arcade game *PacMan* to 3D.

We recruited 26 people (14 female, 12 male, P1–P26), aged 22 – 56 years ($M = 26.1$, $SD = 6.5$), with backgrounds in sociology, management, and computer science.

The participants were separated into groups of three participants for each session. As in the previously described workshops, the study started with a short introduction about S3D and we briefly explained the tool's main functions. Then we presented the tasks and the working environments. For each task, the participants had 25 minutes to design the required UI. Thereby the participants used the desktop setting for one task and the touch surface for the other task. After each task, the participants rated statements about collaboration, creativity, and usability on five-point Likert scales (Figure 7). To minimize sequence effects, we counterbalanced the order of the conditions (touch / desktop) across groups. After the hands-on phase, the participants filled in a final questionnaire. They rated statements about the system's functionality and its usability on five-point Likert scales (Figure 6). The study concluded with a 15 minute discussion about the tool and its usage in the two working environments. One workshop lasted about 90 minutes.

Findings

The final questionnaire (Figure 6) and the discussion show that the participants were overall satisfied with the usability and the functionality of the tool. One experienced designer said: “With the easy usage of the S3D-UI Designer, people with no experience have the opportunity to quickly try out design ideas. [...] This makes designing accessible for everyone.” (P16). This was validated by an inexperienced participant from another group. He admitted “I cannot do anything in Photoshop. Here I understood everything immediately” (P10). Participants stated they like the minimalist, clear design of the tool and the simplicity of its UI, which contributed to its intuitiveness. In general, the final questionnaire (Figure 6) and the discussion reveal that the major benefit of the tool is the interplay of the MirrorBox and the S3D-UI Designer since it supported the “finding of a feeling for 3D” and “created a relaxed atmosphere”. One participant outlined this advantage: “It was very cool to see the output on the MirrorBox, as it differs completely from the result on the display.” (P5). Nevertheless, the arrangement of the input devices and the 3D output device is crucial as one participant noted the effort it took to look into the MirrorBox was “inconvenient” (P4). It should be positioned prominently near the eye level.

Figure 7 presents the means of the ratings for the statements concerning collaboration, creativity, and usability. We aggregated the data by these dimensions. The data shows that touch ($M = 4.106$, $SD = .807$) outperforms the desktop setting ($M = 3.942$, $SD = .782$) for collaboration. A t-test reveals that the difference is not statistically significant, $T(25) = 1.842$, $p = .077$. While the rating of creativity is rather similar for touch ($M = 3.858$, $SD = .849$) and the desktop environment ($M = 3.923$, $SD = .656$), $T(25) = -.502$, $p = .620$, the usability is rated significantly higher for the desktop setting ($M = 3.058$, $SD = .942$) than for the PixelSense ($M = 2.000$, $SD = .980$), $T(25) = -5.543$, $p < .001$.

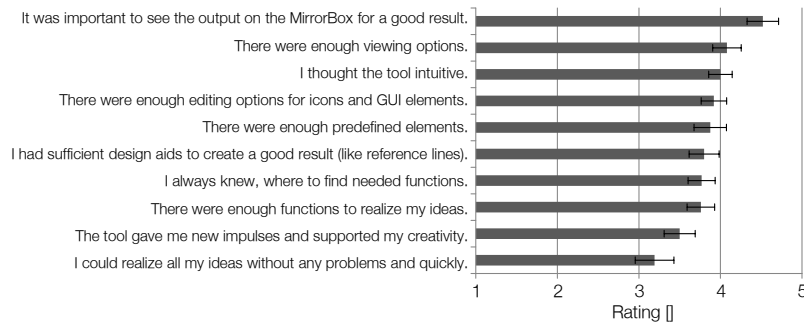


Figure 6: Means and standard errors of the ratings for the final questionnaire's items (5-point Likert scales; 1=strongly disagree to 5=strongly agree).

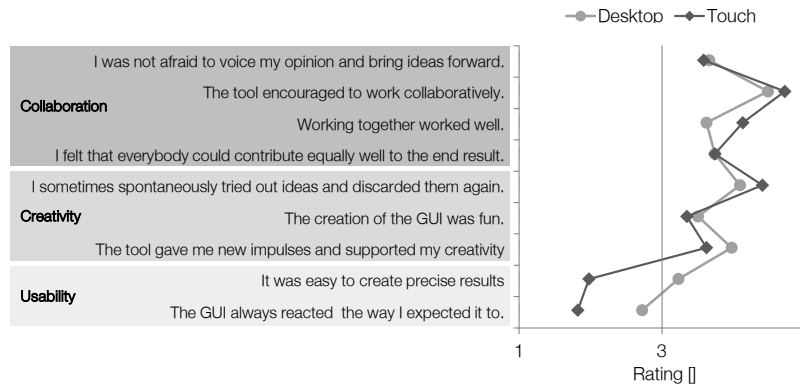


Figure 7: Mean values of the ratings for using the touch interface and the desktop setting. The items are rated on 5-point Likert scales (1=strongly disagree to 5=strongly agree).

In the discussion, participants had different arguments regarding the advantage of touch as well as the desktop environment. In accordance with the usability ratings, benefits of the desktop setting comprise a higher degree of precision and ease of control (“the desktop is the better choice if precise results are needed”). The most recurring argument in favor of touch was the support of collaboration. This finding corresponds to our observations during the study and the trend of the ratings. We noticed that for the desktop setting, usually one person operated the program, while the other team members commented, gave instructions, or voiced ideas. At the interactive table everyone was actively engaged. Some participants stated that touch was more fun (“fun to simply drag the elements into the canvas”) and enhanced creativity (“the level of interaction is much higher so that a lot more ideas come to mind”).

DISCUSSION AND CONCLUSION

In this work, we identify core requirements for S3D prototyping tools. We present the design and development of the S3D-UI Designer, a tool that enables prototyping digital S3D UIs. We implement the tool for designing three virtual depth layers that are instantly displayed on the MirrorBox. We use the MirrorBox as it allows the seamless transition from paper to computer-based prototypes. Due to the modular implementation of the tool it can be extended to several 3D output formats and displays. For future work, the S3D-UI Designer needs to visualize the correct use of depth layers when it comes to the design of more than three layers and their flexible positioning along the z-axis – as, e.g., in the work of Broy et al. [?].

The iterative process in developing the S3D-UI Designer provides requirements for tools that support the prototyping for 3D displays. First, the instant stereoscopic visualization is crucial for the effective development of S3D UIs and the exploration of S3D layouts. Second, the prototyping tool does not need to incorporate comprehensive design functions as, for example, Photoshop. Instead, it should focus on creating the 3D layout. Third, a flexible visualization of depth layers and design aids across layers such as reference lines and highlighting of occlusions are important features for prototyping layered UIs. Fourth, the working environment has a potential influence on the prototyping process and its outcome. While a traditional desktop environment maximizes usability and offers the generation of prototypes with an increased fidelity, an interactive touch surface fosters collaboration for exploring design ideas and variants. We assume that this finding addresses not only prototyping for 3D displays but UI prototyping in general. Nevertheless, future work has to explore how collaboration can be maximized through the used environment and the interaction design of the tool. For example, the integration of handheld devices for each designer could support collaboration and the discussion of individual ideas. Moreover, the extension to other output devices (e.g., autostereoscopic displays) can unfold further requirements.

In the future, we plan to integrate the tool with Unity to enable realtime output on an S3D display. Further studies may investigate the use of the tool with a 3D modality, for example, mid-air gestures using LEAP motion. The code is available on <https://github.com/verenalindner/mirrorbox-designer>.