Chair for Computer Science 10 (Media Computing and Human-Computer Interaction)

RWTH AACHEN UNIVERSITY

ARMenus: An Evaluation of Menu Interfaces in Pen-based AR Applications on Smartphones

Master's Thesis
submitted to the
Media Computing Group
Prof. Dr. Jan Borchers
Computer Science Department
RWTH Aachen University

by
Oliver Nowak

Thesis advisor:
Prof. Dr. Jan Borchers

Second examiner:
Prof. Dr. Torsten Kuhlen

Registration date: 10.01.2019
Submission date: 25.02.2019

Eidesstattliche Versicherung

Nowak, Oliver

Name, Vorname

331397

Matrikelnummer

Ich versichere hiermit an Eides Statt, dass ich die vorliegende Arbeit/Bachelorarbeit/ Masterarbeit* mit dem Titel

AR Menus: An Evaluation of Menu Interfaces in Pen-based AR Applications on Smartphones

selbständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt. Für den Fall, dass die Arbeit zusätzlich auf einem Datenträger eingereicht wird, erkläre ich, dass die schriftliche und die elektronische Form vollständig übereinstimmen. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Aachen, 25.02.2019

Ort, Datum

Unterschrift

*Nichtzutreffendes bitte streichen

Belehrung:

§ 156 StGB: Falsche Versicherung an Eides Statt

Wer vor einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.

§ 161 StGB: Fahrlässiger Falscheid; fahrlässige falsche Versicherung an Eides Statt

(1) Wenn eine der in den §§ 154 bis 156 bezeichneten Handlungen aus Fahrlässigkeit begangen worden ist, so tritt Freiheitsstrafe bis zu einem Jahr oder Geldstrafe ein.

(2) Straflosigkeit tritt ein, wenn der Täter die falsche Angabe rechtzeitig berichtigt. Die Vorschriften des § 158 Abs. 2 und 3 gelten entsprechend.

Die vorstehende Belehrung habe ich zur Kenntnis genommen:

Aachen, 25.02.2019

Ort, Datum

Unterschrift

Contents

List of Figures

List of Tables

Abstract

Modern smartphones allow people to experience Augmented Reality without the need of additional hardware. However, the touch input used in such applications is only two-dimensional and suffers from the fat finger problem. To overcome this, for example for sketching and modeling tasks, mid-air input devices can be used to provide precise and three-dimensional input. Mid-air gestures are often used for basic tasks like moving or resizing virtual objects. In contrast to this, tasks like changing the shape or the color of an object are difficult to describe in intuitive gestures. An usual way to communicate such tasks to the application is the usage of context menus.

In this thesis, we describe the design and evaluation of three context menus for handheld AR using bimanual interaction with a smartphone on the one hand and a 3D trackable pen (the *ARPen*) on the other hand. We propose three menu techniques using mid-air input of the pen as well as touch-input provided by the smartphone.

We conducted a user study to identify the most suitable regions on the smartphone's touchscreen for the placement of our touch menus. Based on those results we implemented *touch-triggered screen menu* and *pen-triggered screen menu*, which are menus which use touch for the item selection, but are opened selecting a target with either the pen or touch. The *AR pie menu*, on the other hand, is a menu technique using the ARPen for both triggering the menu and item selection.

In our final user study, we observed that the *AR pie menu* is significantly slower than the touch menus. However, evaluating the user's preferences, we could not find a clearly preferred menu technique.

Überblick

Moderne Smartphones ermöglichen es, Augmented Reality ohne zusätzliche Hardware zu nutzen. Jedoch ist die Touch-Eingabe für AR Anwendungen nur zweidimensional und leidet unter dem "Fat Finger"-Problem. Deshalb werden, für zum Beispiel Skizzierungs- oder Modellierungsanwendungen, 3D-Eingabegeräte verwendet, welche dem Nutzer präzise Eingaben ermöglichen. Für das Bewegen oder Skalieren von virtuellen Objekten sind "Mid-air"-Gesten sehr gut geeignet, jedoch sind Aktionen wie das Ändern der Form oder der Farbe von Objekten relativ schwer durch Gesten zu beschreiben. In solchen Fällen werden häufig Kontextmenüs zu Hilfe genommen.

In dieser Arbeit wird beschrieben, wie wir Kontextmenüs für die beidhändige AR-Interaktion mit einem Smartphone und einem stiftähnlichen, 3D-Eingabegerät (den *ARPen*) entwickelt haben. Insgesamt stellen wir drei Menü-Techniken vor, welche sowohl "Mid-air"-Eingaben mit dem ARPen als auch Touch-Eingaben über das Smartphone verwenden.

Um eine geeignete Position und Größe für die Touch-Menüs zu finden, haben wir eine Studie durchgeführt, welche untersucht, welche Bereiche auf dem Bildschirm des Smartphones angenehm zu erreichen sind. Auf Basis dieser Studie haben wir die Menüs *Touch-triggered Screen Menu* und *Pen-triggered Screen Menu* entwickelt, bei denen das Menü entweder über eine Touch-, oder Stiftauswahl geöffnet wird. Dabei wird in beiden Fällen die Auswahl der Menüeinträge mit dem Daumen ausgeführt. Bei dem *AR Pie Menu* hingegen, wird das Menü komplett mit dem ARPen genutzt.

In unserer Hauptstudie konnten wir beobachten, dass das *AR Pie Menu* signifikant langsamer war als die anderen beiden Techniken. Jedoch zeigten die Bewertungen der Studienteilnehmer keine klare Präferenz für eine der drei Menütechniken.

Acknowledgements

At first, I would like to thank Prof. Dr. Borchers and also Prof. Dr. Kuhlen for examining this thesis.

I also want to thank my supervisor Philipp Wacker who spend many Tuesday mornings discussing how people hold smartphones, which types of menus exist and how those should look like.

Additionally, I would like to thank my family, my friends and everyone who supported me during this thesis.

Especially, I would like to thank my girlfriend – Yvonne – for your advice at any time of the day and your support since we met.

Conventions

Throughout this thesis we use the following conventions.

Text conventions

The whole thesis is written in American English. The first person is written in the plural form. Unidentified third persons are described in female form.

Chapter 1

Introduction

Augmented Reality (AR) allows the user to add virtual content into the real environment. While research on this topic already started in the early 1990s [Caudell and Mizell, 1992], AR is still in active development. Nowadays, the necessary hardware starts getting cheaper and sophisticated enough to reach the consumer market. Head-Mounted Display (HMD) devices like the Microsoft Hololens[1] offer AR apps to increase productivity (e.g., Dynamics 365 Layout[2]) as well as for entertainment purposes (e.g., RoboRaid[3]). Another possibility to experience Augmented Reality are handheld AR applications for smartphones which use frameworks like Google's ARCore[4] or Apple's ARKit[5]. In contrast to HMDs, handheld AR devices let the user look only through a small frame (the smartphone's display) into the virtually augmented world. Since handheld AR can be used with the smartphone, this technology reaches a huge group of users without any ad-

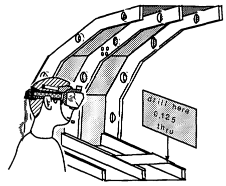


Image from Caudell and Mizell.

AR example application from 1992 by Caudell and Mizell.

Handheld AR reaches a huge group of people.

[1] https://www.microsoft.com/en-us/hololens *(Accessed: 22.02.2019)*

[2] https://dynamics.microsoft.com/en-us/mixed-reality/layout/ *(Accessed: 22.02.2019)*

[3] https://www.microsoft.com/en-us/p/roboraid/9nblggh5fv3j?rtc=1&activetab=pivot:overviewtab *(Accessed: 22.02.2019)*

[4] https://developers.google.com/ar/discover/ *(Accessed: 22.02.2019)*

[5] https://developer.apple.com/arkit/ *(Accessed: 22.02.2019)*

ditional costs for AR-specific hardware. Furthermore, this makes AR quickly accessible and invites the users to spontaneous use when such an application is really needed.

Using 2D touch input for 3D interaction is ambiguous.

When using handheld AR on smartphones, the user can interact with the virtual environment only using two-dimensional touch input on the phone's screen. This impedes the interaction with virtual objects when the interaction has a three-dimensional nature. An example for that is the translation of objects within AR, which is a highly three-dimensional task. The user cannot know in advance, whether an up-movement moves the object upwards, or just increases its distance into the z-direction. When she, then, found out in which direction she is moving the object, she still might need to find a way to move the object into the other direction. Additionally, touch input suffers under the fat finger problem which impedes precise input [Siek et al., 2005, Vogel and Baudisch, 2007]. To increase the precision and also allow 3D input, researchers used, for example, pen-similar input devices which are trackable within the 3D space [Jackson and Keefe, 2016, Wacker et al., 2018, 2019, Wu et al., 2017]. Wacker et al. and Wu et al. presented 3D-trackable pens which are tracked in 3D space using computer vision technologies and, nevertheless, only need a single camera for this.

Trackable pens allow precise 3D input.

Using the selection techniques for mid-air pen input in handheld AR applications presented by Wacker et al. [2019], we introduce several menu techniques for this context. Those menu techniques do not only consist of mid-air interactions using the *ARPen* presented in their work. Instead, they also make use of the touch capabilities of the smartphone which is used in handheld AR applications anyway. Experience AR through the touchscreen of a smartphone on the one hand and interact with the virtual content using a pen on the other hand, is a novel scenario. For this scenario, it is not clear which type of input is recommendable for certain actions and thus, needs to be weighed carefully. In this thesis, we will investigate whether menu interaction should take place in AR using mid-air selection techniques or whether the menu should be usable via touch input. The way people grasp the smartphone when using an ARPen application varies a lot. They

ARPen or touch input?

We observed the users' smartphone grasps and the corresponding interaction area.

vary the smartphone-holding hand, the orientation of the phone and the grasp itself in such a task. We conducted a user study investigating those factors to identify comfortably reachable regions on the smartphone's touchscreen to place our touch menus at.

For the comparison of the menus, we implemented a linear menu layout for the touch input and a pie menu which is placed within the AR environment for the mid-air interaction. In an user study, we then compared three *menu techniques*: one technique using only mid-air input with the ARPen (*AR pie menu*), one technique using only touch (*touch-triggered screen menu*), and one hybrid technique which uses the ARPen for menu triggering, and touch for item selection (*pen-triggered screen menu*). Those menu techniques, should provide fast and accurate interaction with a context menu without interrupting the task the user intentionally planned to do.

We investigated three menu techniques for fast and accurate item selections.

1.1 Outline

In the following, we describe the related work in Chapter 2. There, we will deal with the fundamental works of menu design, how menus are used in Augmented and Virtual Reality and how 3D-trackable, pen-like input devices were already used in research.

Chapter 3 "Menu Design", then, deals with the design process of our menu techniques. We describe how the ARPen works, introduce an example application which could benefit from the three-dimensional input of the pen and give an example of why menus are needed in Augmented Reality. After that, we describe our first study to identify a reasonable region for menus using touch input in Section 3.3. Since we separate the menu triggering and menu layout for our menu techniques, we also briefly describe the results of Wacker et al. [2019] regarding their selection techniques in Section 3.4 "Excursus: Target Selection Using the ARPen". Finally we define the layout and the behavior of our menus for both touch and mid-air input and combine those with two selection techniques from the excursus.

In Chapter 4, we describe the user study we conducted to investigate the performance differences of *AR pie menu*, *touch-triggered screen menu* and *pen-triggered screen menu* as well as the subjective preferences of the participants.

We, then, conclude the results of this thesis and suggest future research regarding bimanual menu interaction in AR in Chapter 5 “Summary and Future Work”.

Chapter 2

Related Work

Since the first graphical computers were available, menus were used to provide easy access to system or application commands. The commands did not need to be remembered anymore which leads to an increased the velocity of command execution. Figure 2.1, for example, shows an graphics application running on SmallTalk-76. There, the user could simply change the drawing tools or change a tool's parameters like the line thickness by selecting the corresponding entry in the menu in the top left.

Researchers began early to investigate how menu performance can be increased changing item orders, layouts and menu structures. Card [1982] investigated how different orderings of menu items affect the search time in a vertical linear list containing 18 items. He found that inexperienced users benefit of an alphabetical order, followed by semantically grouped items. However, the ordering becomes less relevant when the user knows the menu. McDonald et al. [1983] also found similar results. However, they found that categorical grouping lead to faster target selections.

Card showed that advanced users find menu items similarly fast using different item organizations.

Researchers like Kiger [1984], Landauer et al. [1985] and Miller [1981] investigated the trade-off of menu breadth (number of items in one hierarchy level) and depth (number of hierarchy levels) on different input devices. For this, Miller investigated four depth-breadth configurations, all containing 64 menu items (two items on six levels, four

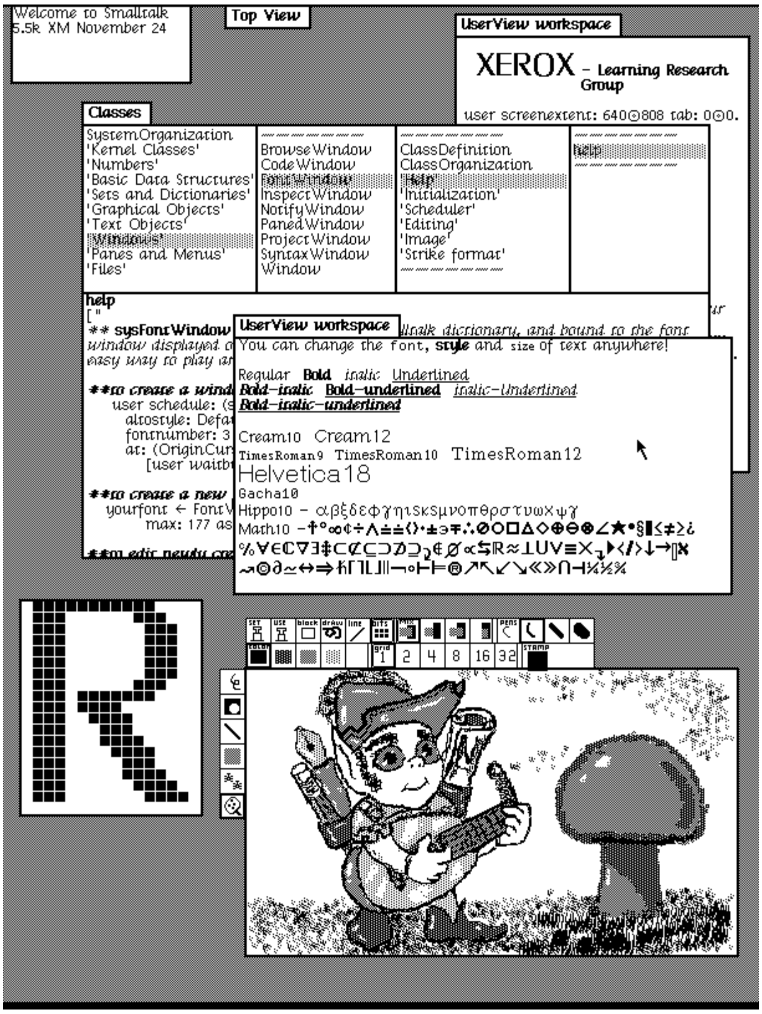


Figure 2.1: SmallTalk-76 already used menus in the 1970s. Image taken from `https://en.wikipedia.org/wiki/Xerox_Alto` (accessed: 22.02.2019).

Menus with eight items on two hierarchy levels perform well.

items on three levels, eight items on two levels and 64 items on one level). The items were grouped semantically and were arranged as shown in Figure 2.2. Miller found that using eight items on two hierarchy levels was the fastest, easiest to learn and least error rate. Kiger [1984] investigated the menu performance with keyboard input similarly, but included two additional menu configurations with a depth of two but with varying breadth (first four, then 16 items and vice versa). Also Kiger reported the superiority of a

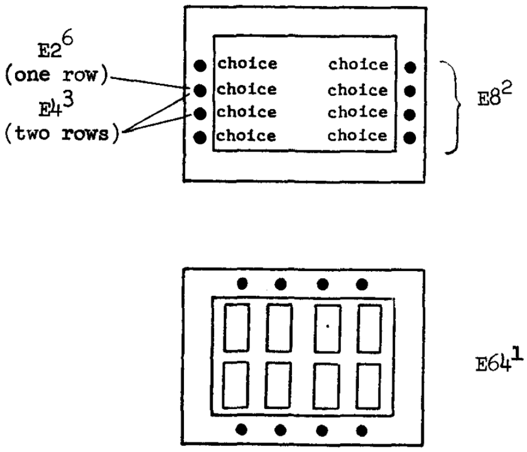
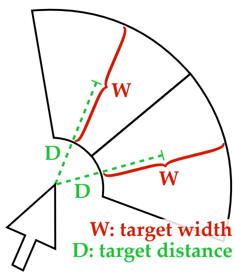


Figure 2.2: Menu implementation by Miller [1981]. The menus were spread over the panel depending on the depth-breadth condition tested. A condition with a menu breadth of b and a menu depth of d is labeled with Eb^d. The items were selected by using the pushbuttons on the panel frame. Image adopted from [Miller, 1981].

menu structure with a depth of two and a breadth of eight items. Nevertheless, Kiger and Miller [1981] recommend to increase the breadth instead of the depth if a semantically grouping does not exactly fit their recommended configuration. This is also what Landauer et al. [1985] suggest. They observed menu structures with a breadth up to 16 items using touch input for the item selection. The authors found that using a branching factor of 16 still allows fast menu selection when categorizing integers and words in alphabetical and numerical ranges.

If necessary, menus should increase in breadth rather than in depth.

Callahan et al. presented menus with a pie layout 1988 (cf. Figure 2.3) and compared this menu type with linear pull-down menus. The pie menu benefits from its radial layout when opened around the user's cursor because, then, the target distance and target width is the same for all menu items. Thus, the item selection time should not vary significantly according to Fitts' Law [Fitts, 1954]. Callahan et al. compared the pie and the linear menu in an task in which participants had to perform repeated item selections on menus containing eight items without any submenus.



Pie menus benefit of equal target distances and target widths (Fitts' Law).

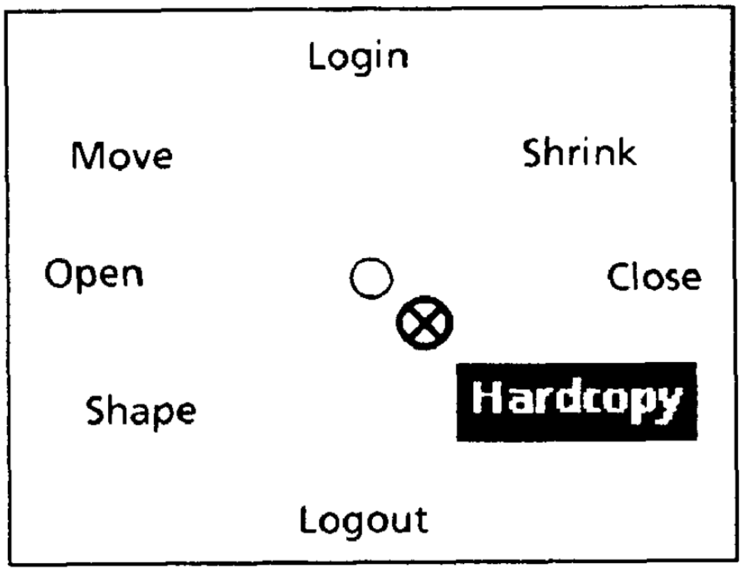


Figure 2.3: The pie menu arranges all menu items equally distributed around the user's cursor. Image taken from Callahan et al. [1988].

The authors found that pie menus performed significant faster than linear menus and have shown a marginally significant lower error rate.

Marking menus allow blind menu selection.

Since each item of a pie menu is mapped to a specific moving direction during the item selection, experienced users could simply memorize the location of a particular menu item and perform a selection by relying upon their muscle memory. Kurtenbach et al. introduced *marking menus* [Kurtenbach, 1993, Kurtenbach and Buxton, 1993, 1994], which make use of the this characteristics of pie menus. With marking menus, the user has not to wait until a menu appears to make a selection. Instead, the user can simply draw a stroke into the direction of the target's position without the menu showing up. However, when the user does not know where the target item is placed the menu appears after a certain amount of time. As shown in Figure 2.4, marking menus can also used with hierarchical menu structures [Kurtenbach and Buxton, 1993]. For this, the user has to concatenate the strokes for each hierarchy level. Kurtenbach and Buxton [1994] showed

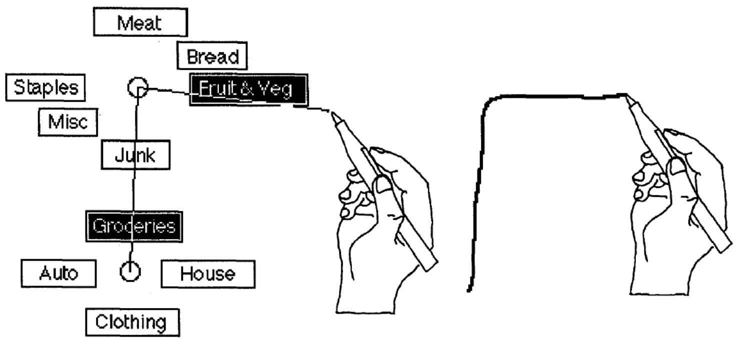


Figure 2.4: When using marking menus, the user can draw strokes to fasten item selection. Image adopted from Kurtenbach and Buxton [1993].

in a field study that marking performs faster than using the pie menu conventionally and that the user's skill increased with time. Additionally, participants commented that marking was not leading to significant more errors. Kurtenbach and Buxton [1993] also investigated various depth-breadth configurations for marking menus. The authors suggested using smaller breadth with increasing depth. However, they mentioned that a menu using a depth of two with eight items per level is also suitable.

2.1 Menus in Augmented and Virtual Reality

Also in Augmented and Virtual Reality menus are used to communicate with the system. Van Teylingen et al. [1997], for example, adopted a tree menu, as often used for file navigation, for data visualization and analysis tasks using a ray-cast selection technique in VR.

Mid-air input for horizontal menus is slower and more error-prone.

Dang and Mestre [2011] observed the effect of menu orientation (*horizontal, 45° and vertical*) on the target selection time in a Fitts' Law pointing task in Virtual Reality. The participant had to touch the target items with the finger and the target distance and width varied. The authors found that the *horizontal* menu was slower and led to a higher error-rate.

Menus can be fixed in the world, on the screen or be placed at objects.

Feiner et al. [1993], Namgyu Kim et al. [2000] and Lee et al. [2011] presented different locations to place UI elements when using AR or VR. In general, the authors differentiate between UI elements which are at a fixed placed within the 3D environment (*world-fixed*), elements which are fixed to the users viewport (*view-fixed* or *display-referenced*), and UI elements which adapt their position to a particular object (*object-fixed*). Lee et al. [2011] additionally splits object-fixed menus into *controller-fixed* menus, which belong to the input device, and *target-fixed* menus which correspond to all remaining objects.

Das and Borst [2010] investigates the effects of menu location (*world-fixed* vs. *object-fixed*), layout (*linear* vs. *pie*) and breadth (*4*, *7* and *10* items) on the selection time and error rate for three different selection techniques. The authors attached a local pointer to the menu (red pointer in Figure 2.5). This pointer (*PAM – pointer-attached-to-menu*) can be controlled by either translate (*PAMT*) or orientate (*PAMO*) the input device. Das and Borst compared those techniques with the conventional *ray-casting* pointing technique. The authors found that *object-fixed* menus and the pie menus were significant faster than their counterparts. Increasing breadth, selection times increased significantly and also error rates increased. Pointing with *ray-casting* was

Pie menus near the target perform better.

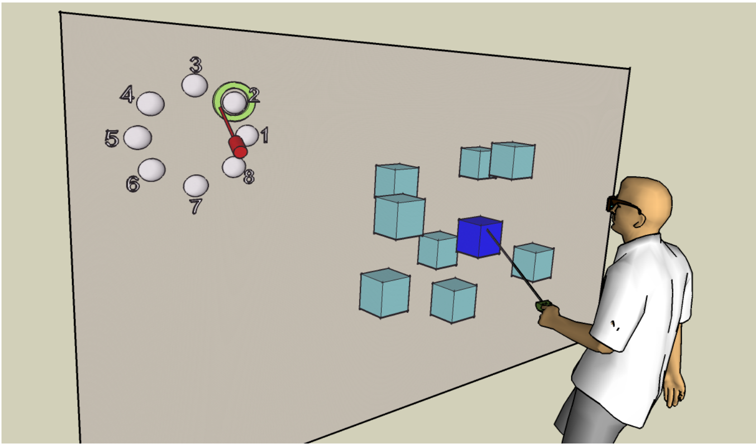


Figure 2.5: Study setup by Das and Borst [2010]. The user selects the target item of a world-fixed menu using a PAM technique. Image taken from [Das and Borst, 2010].

overall faster than the PAM techniques and preferred by the participants.

Gebhardt et al. [2013] investigated different visualizations of pie menu hierarchies and three different menu abort methods using three pointing techniques similar to the PAM techniques. The participants of their study should, inter alia, change the appearance of an object using a menu which is controlled either via *ray-casting*, a translation technique or a rotation technique. For the translation technique (named: *hand projection*) a cursor is shown on the menu which moves parallel to the users arm. For the rotation technique (*hand rotation*), the cursor moves around the menu independent of the hand rotation. Figure 2.6 illustrates the different hierarchy visualizations. The *depth offset* visualization (a) only pushes the parent menus into the background. *In-plane offset* (b) shifts the parent menu into the opposite direction such that the user get can clearly determine which item was clicked to open the new menu. To avoid occlusion, the authors introduce the *linear in-plane offset* visualization (c). In this visualization, all previous menu levels are shown in a horizontal list next to the menu, still highlighting the selected items. The abort conditions were a circular button in the center of the pie menu (*dead zone*), a separate menu item or an additional button on the con-

Gebhardt et al. investigated three pie menu hierarchy visualizations.

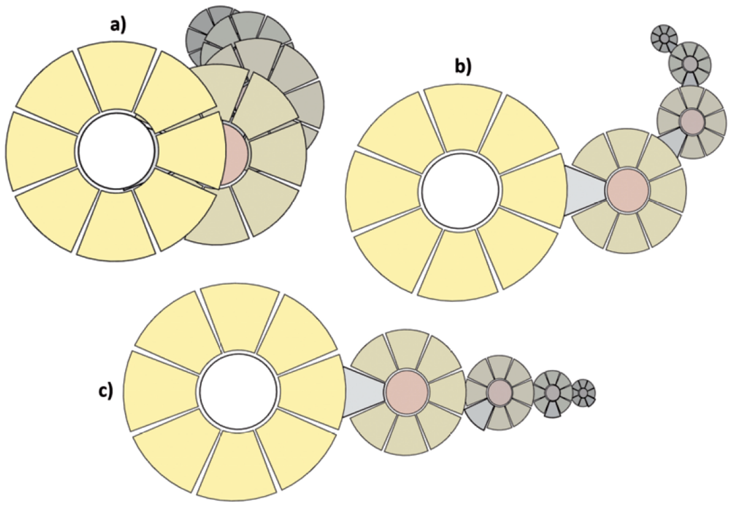


Figure 2.6: Gebhardt et al. investigated three menu visualizations. *Depth offset* (a) shifts the menu to the background. *In-plane offset* (b) shifts the parent menus further away to avoid occlusion of the new menu and its parent and *linear in-plane offset* (c) arranges the previous menus in a list. Image taken from [Gebhardt et al., 2013].

troller. The author's analysis revealed that *ray-casting* was significantly faster. Regarding the hierarchy visualization no significant effects were found. The qualitative results of the study show that *ray-casting* and the *dead zone* were most preferred. The authors mention a slight tendency of the ratings for the *depth-offset* visualization.

Users preferred *ray-casting* for selection and a *dead zone* for abort.

A lot of research in Augmented and Virtual Reality aims to make the interaction as natural as possible. One approach for this is to use the hands directly for the input with the virtual content. Jacoby and Ellis [1992] presented a gesture-based approach to interact with menus in VR. The user had to form a fist with two fingers extended to open the menu. Then, a ray is casted from her extended fingers (cf. Figure 2.7a) with which she can select the target item. By changing the hand posture the user confirms the selection.

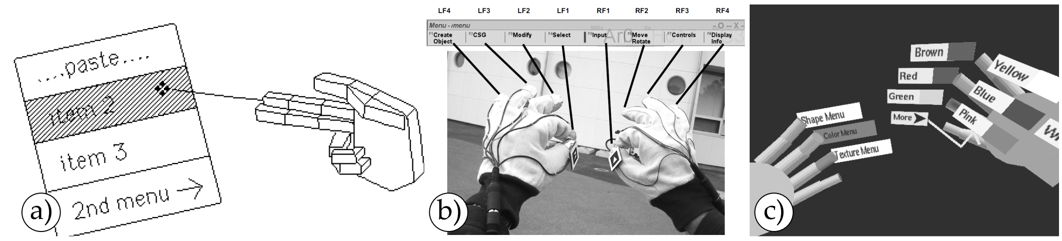


Figure 2.7: Gesture-based AR/VR menu techniques by Piekarski and Thomas [2002a] (a), Jacoby and Ellis [1992] (b) and Bowman and Wingrave [2001] (c). The menu presented by Piekarski and Thomas uses gestures to open the menu and create a ray. The menu system of Tinmith-Hands (b) and the TULIP menu (c) select a menu item by touching the corresponding finger with the thumb. Images are adopted from the corresponding publications.

In contrast to that, Piekarski and Thomas [2002b] presented a menu for which pointing is not necessary. The menu system presented for the *Tinmith-Hands* is a view-fixed menu in which each item is mapped to one finger (the mapping is shown in Figure 2.7b). To select one item the user needs to touch the corresponding fingertip with the thumb. However, using this technique, the user needs to learn how the hands have to be oriented such that the order of the menu items matches to the fingers.

For the menu selection with Tinmith-Hands the user just need to touch her fingertips.

Bowman and Wingrave [2001] presented a similar approach named *TULIP*, which also use the mechanism for item selection of touching a finger with the thumb. However, the authors avoids the need of learning the finger-item mapping by implementing a controller-fixed menu. In TULIP each menu item is placed directly at the corresponding finger (cf. Figure 2.7c).

The TULIP menu shows all menu items directly at your finger.

The most layouts of the previously named AR/VR menu implementations were two-dimensional menus adopted from WIMP[1] desktop interfaces. In contrast to those, is the *Command & Control Cube* (C^3) [Grosjean and Coquillart, 2001, Grosjean et al., 2002] a VR menu in which the items are arranged in a three-dimensional grid (cf. Figure 2.8). The menu appears when the user pinches with the thumb and the index finger. Then, she can move indirectly a cursor from the middle of the $3 \times 3 \times 3$ grid to

[1]Windows, Icons, Menus, Pointers

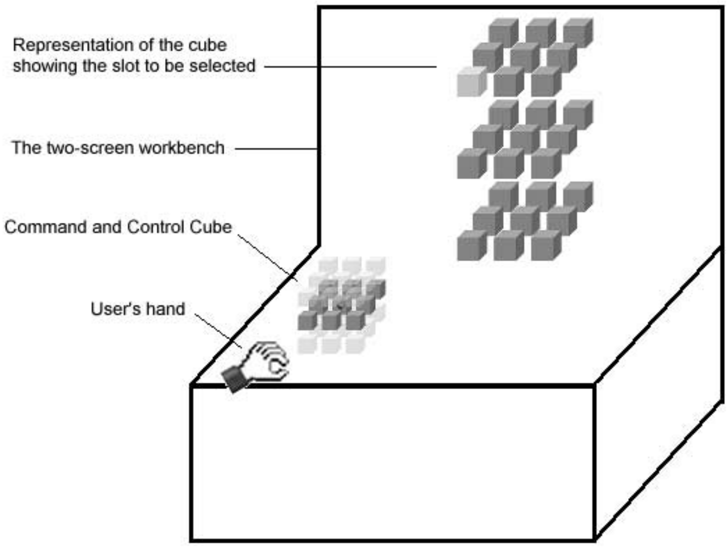


Figure 2.8: Study setup to test the Command & Control Cube. The user has to select the highlighted target in the large representation of the cube. Image taken from [Grosjean et al., 2002].

The Command & Control Cube is a 3D marking menu.

the target item by moving the hand in the corresponding direction. To improve visibility only the horizontal layer is shown in which the user is moving the cursor. Similar to marking menus, each menu item can be identified by a position in the grid. Thus, users can select the menu items blindly by only moving the hand into the corresponding direction. Grosjean et al. [2002] evaluated the performance of C^3 especially for blind usage. For the blind conditions, they indicate whether the user moved with the cursor from one cell to another by auditory or tactile feedback. In a study, in which the participants had to select a target shown on a large representation of the grid (cf. Figure 2.8, the authors found that showing the cube during the interaction lead to faster and more accurate selections in comparison to the blind usage with and without feedback.

2.2 Pen Input in Augmented and Virtual Reality

Another way to interact with virtual objects in VR and AR is pen input. Bowman et al. [1998] used a pen and a tablet to navigate through animal habitants. This technique was also compared to the TULIP menu [Bowman and Wingrave, 2001] and performed faster in their study.

Bowman et al. used a pen and a tablet to move through a virtual animal habitant.

Komerska and Ware [2004] investigated how menus performance benefit from haptical constraints in a fish tank VR environment. For this, the authors used a pen attached to mechanical arm which is able to simulate solid surfaces in mid-air. The authors observed that mid-air menu performance benefits from assistive forces.

Pen input benefits from haptical constraints.

Wacker et al. [2018] compared how accurate people can draw along convex and concave edges as well as visual guides (e.g., straight printed line on a book cover) on physical and virtual objects in Augmented Reality. For this, the authors tracked a pen using a VICON motion tracking system. The authors found that especially for physical objects both surface guides as well as visual guides increase the drawing accuracy.

Visual and surface guides improve drawing accuracy on physical objects.

Jackson and Keefe [2016] presented with *Lift-Off* a 3D sketching application for Virtual Reality. Using Lift-Off also uses a pen for the interaction (a similar approach as used by Wacker et al. [2018]). However, the user is not directly drawing in mid-air, instead, the user starts sketching on paper or a graphics tablet (cf. Figure 2.9a). Then, those sketches are imported into the application and the user can place them into the virtual environment. From those sketches, the user can select a drawn line and drag that line into the 3D space (Figure 2.9b). Finally, the user can define surfaces between those lifted lines and create a 3D sculpture (Figure 2.9c).

Lift-Off avoids inaccurate mid-air sketching by importing 2D sketches into VR.

Figure 2.9: Design process of creating a sculpture with Lift-Off. The user starts sketching on paper (a) and imports those into the application. Then, the user can drag lines from the drawings into the 3D environment and define surfaces between them (b) to create a 3D sculpture (c). Image taken from [Jackson and Keefe, 2016].

The DodecaPen can be tracked accurately using a single camera.

While Wacker et al. [2018] and Jackson and Keefe [2016], used complex and expensive tracking methods to track the user's pen, Wu et al. [2017] presented with *DodecaPen* a 3D trackable pen which needs only a single camera. The DodecaPen uses multiple markers on the end of the pen to compute the 3D location of the pen in relation to the camera in real-time (Figure 2.10a, b). Figure 2.10 c) and d) demonstrate the precision of the tracking method. The authors state an accuracy of 0.4 mm.

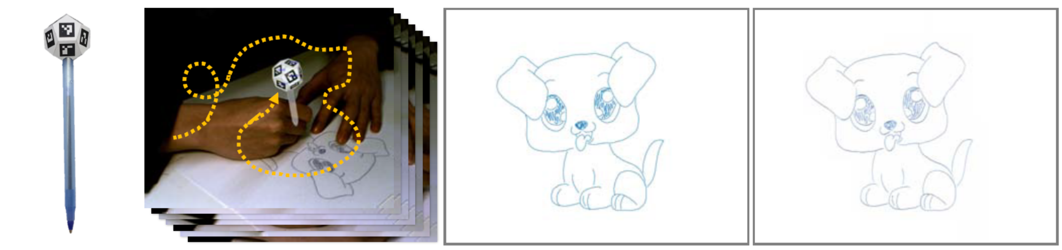


Figure 2.10: The DodecaPen (a) is a real-time trackable pen (b) with high accuracy (c & d). Image taken from [Wu et al., 2017].

Wacker et al. [2019] used the same marker tracking technology as the DodecaPen in their research on object selection and manipulation in handheld Augmented Reality. However, we will describe the operating principle of their *ARPen* in the following chapter and also go into detail about their proposed selection techniques in Section 3.4 “Excursus: Target Selection Using the ARPen”.

Chapter 3

Menu Design

In this chapter, describe the design process of the menu techniques we investigated in Chapter 4 "A Comparison of Mid-Air AR and Touch Menus". For this, we begin to explain the operating principle of the ARPen presented by Wacker et al. [2019]. Then, we describe how we classify menus into application menus and context menus and after an excursus about object selection with the ARPen, we will describe our final menu techniques.

3.1 The ARPen

Appearance of the ARPen.

The ARPen is a pen-shaped 3D printed input device for handheld AR applications. The pen is shown in Figure 3.1. It consists of a twelve centimeters long stick with three buttons near the pen's tip and a cubical box (side length: 4 cm) at the end. arUco marker[1] are placed on each face of the box to track the pen in the 3D space.

[1] https://docs.opencv.org/3.1.0/d5/dae/tutorial_aruco_detection.html *(Accessed: 22.02.2019)*

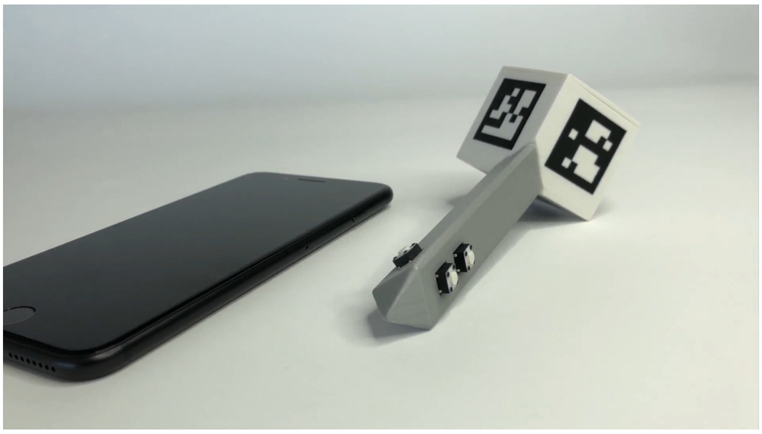


Figure 3.1: The ARPen is an input device with three buttons for the input and a box with unique arUco markers at the end.

3.1.1 Operating Priciple

The pen's location is tracked using ARKit and the arUco framework.

The ARPen can be used in combination with our ARPen app[2] which uses Apple's ARKit framework[3] to provide an AR experience to the user. Figure 3.2 illustrates how the position of the pen is determined. First, the app creates the AR world (in this thesis also: *3D world* or *AR/3D scene*) which described the virtual space with its own origin. The AR world's origin stays at the same location in the physical environment. The systems is able to compute the position of the smartphone in relation to the AR world's origin at any time by using the ARKit framework (a). Additionally, the arUco framework computes the position of the camera-visible markers in relation to the position of the camera (b). Combining the information provided by both frameworks lets us track the location of the markers in relation to the AR world's origin (c). Finally, the position of the pen's tip is calculated using the markers' position and orientation. When the user moves the ARPen such that the markers are outside the view, the system is not able to predict the position of the pen's tip, even if the tip is still visible for the user.

[2] `https://github.com/i10/ARPen` *(Accessed: 22.02.2019)*
[3] `https://developer.apple.com/arkit/` *(Accessed: 22.02.2019)*

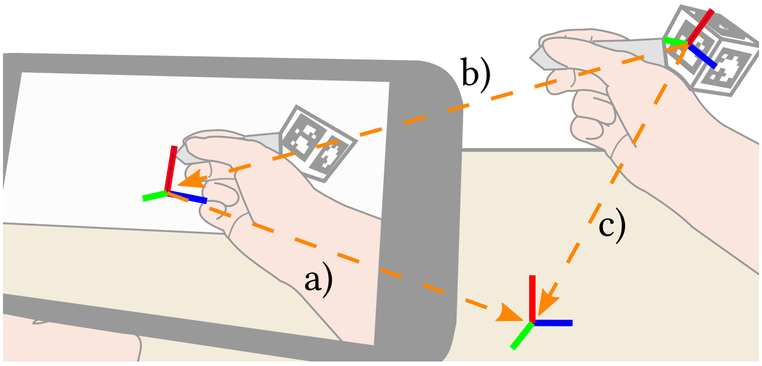


Figure 3.2: For calculating the ARPen's position, the position of the smartphone in relation to the AR world's origin is calculated (a). Using the relative position of the ARPen to the smartphone (b), the position of the pen in relation to the AR world can be calculated (c).

3.1.2 Example Application for Handheld Augmented Reality

Using the ARPen for 3D modeling

Since the ARPen offers input into width, height and depth, one possible application of the pen is 3D modeling. For example, the shape of geometries could be simply changed by dragging a face of the geometry with the pen to the desired position. Change the position of a corner or an edge could also be done analogously.

In 3D CAD (*computer aided design*) software users often have to interrupt already very simple and common operations like translations or extrusion because they need to change their view port, and thus, need to operate a navigation widget. Using a handheld AR application, the user can move more freely fluently change her perspective also during such operations.

Furthermore, doing such a task in AR provides a more accurate intuition of sizes to the user. When the user, for example, wants to design a personalized coaster for a teacup, she could simply place the cup in front of her and scale the modeled coaster such that it matches the cup's dimensions.

3.2 Menu Design for the ARPen Scenario

Menus offer clear and organized UIs.

When using a smartphone for handheld Augmented Reality, the display of the smartphone does not offer a lot of space for UI elements. Especially, thinking about that the UI is used to see the virtual content, the UI should be clear and organized. Taking the CAD application example of the previous section, this app should provide many features like changing the shape or the material of an object, combining two bodies, showing only particular parts of the design at once or exporting the model. Those features are essentially for a modeling software, however, some of them are used only very infrequently and should not be visible to the user all the time. Because of that, the application should provide those features in clearly organized menus the user can reach easily and quickly.

3.2.1 Application and Context Menus

Application menus offer session level actions like switching workspaces or saving the progress.

In general, we distinguish between two types of menus: context menus and application menus. Application menus (cf. Figure 3.3, red) provide actions which effect the state of the application or the session like switching workspaces, saving the current progress or changing the application's appearance. Furthermore, they also often provide a first entry point for basic functionality. For example, a file manager often provides actions for file creation and deletion in the application menu. Since application menus provide actions which are used rather infrequent, they are often placed distant from the user's place of interaction.

Actions provided in a context menu should only provide actions referring to the selected object.

In contrast to that, context menus (cf. Figure 3.3, green) provide actions referring to a certain object. If the user of our exemplary CAD app would like to change the appearance of an object, she would select it, open a context menu containing multiple actions only referring to this object (e.g., changing the shape, color, material) and select the item for changing the color. Context menus are often displayed directly at the position of the users interaction (a cursor or the the object itself).

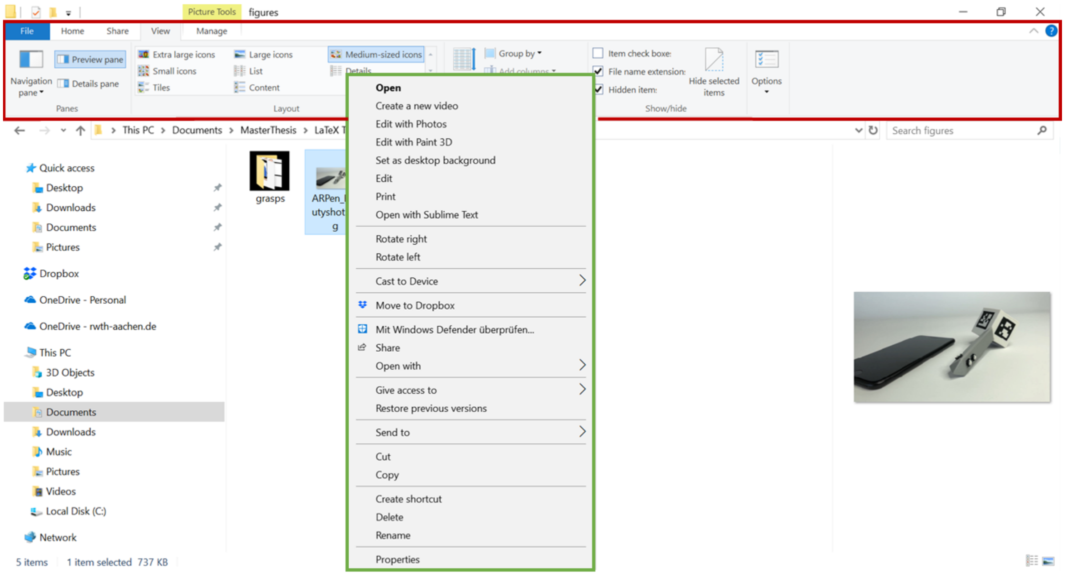


Figure 3.3: Application menus (red) provide application-level and session-level actions, while context menus (green) offer item specific actions. Example taken from Microsoft's File Explorer.

However, very often those types of menus are mixed up. For example, in many applications the user can open a context menu even when she is not pointing at an object. In such a case, the appearing menu offers a subset of actions which can already be found in the application menu like creating a new object or pasting.

This thesis focuses on context menus.

Augmented Reality as well as Virtual Reality are often used to facilitate the representation of three-dimensional content like, for example, the furnishing of a room or a representation of the universe. These applications have a strong focus on the presented objects and thus, context menus are used very frequently. Because of this, we decided to focus on those context menus.

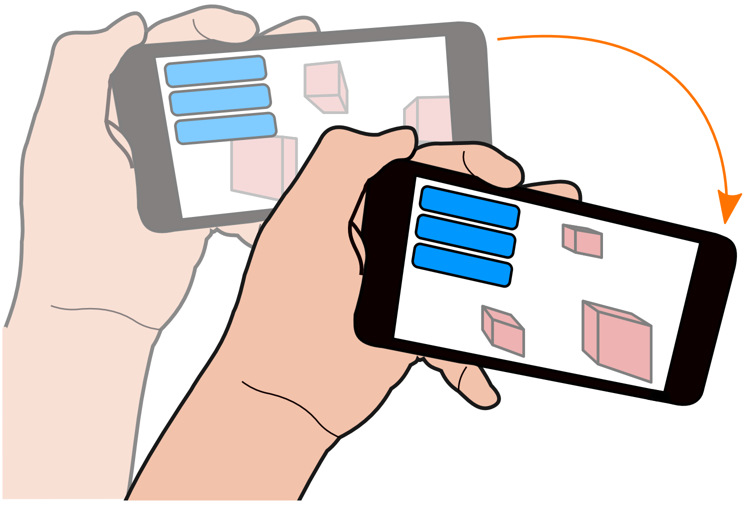


Figure 3.4: A view fixed menu is attached to the user's view, while most other virtual objects stay at the same position in the 3D scene.

3.2.2 Menus in an ARPen Application

Touch menus are view-fixed.

When using the ARPen, we have two possibilities of interacting with the virtual environment. We can either use the pen to interact in mid-air, or we can use the touchscreen of the smartphone the application is running at. In this thesis, we want to compare menus in which the menu items can be selected with the ARPen on the one hand (*AR menus*), and menus in which item selection is done via touch input on the other hand (*touch menus*). Those two types of menus will be placed differently in the AR application. While AR menus will be placed into the 3D scene and keep there independent of where the user is looking at, touch menus will be shown directly on the screen and be *view-fixed*. As illustrated in Figure 3.4, a view-fixed menu keeps perspectively at the same place even if the user changes the perspective by moving the smartphone.

3.3 Study 1: Smartphone Grasp & Interaction Area

When using the ARPen, the smartphone is usually hold in the non-dominant hand while the dominant hand is used for the 3D interaction. Thus, when using the touch menu, the user should be able to select the menu items without the need of frequently moving the dominant arm to the touchscreen. Instead, in many variations of smartphone grasps, the user can still use several fingers of the non-dominant hand to interact with the touchscreen. Using the free fingers of the smartphone-holding hand allows to use the menu quickly. We conducted an user study in which we investigated how people hold the smartphone in the ARPen scenario and which areas they can reach on the touchscreen to identify a suitable area to place touch menus.

3.3.1 Experimental Design

The smartphone size and orientation were varied during the study.

In the user study, the participants were asked to perform the task in one portrait and two landscape orientations. The landscape orientations were defined according to the relative position of the camera to the user's hand. Originally, the landscape orientations were named *'camera away from hand'* and *'camera on hand's side'*. For simplicity, we will describe the landscape orientations with *camera left* and *camera right* assuming the participant hold the smartphone in the left hand. In the case that the participant used her right hand for holding the smartphone, we will mirror the data as illustrated in Figure 3.5.

Additionally, the participants were asked to use the ARPen with two smartphones of different sizes: 4.7" (*small*) and 5.5" (*big*). Both, *orientation* (in the relative manner) and *smartphone size* were counterbalanced.

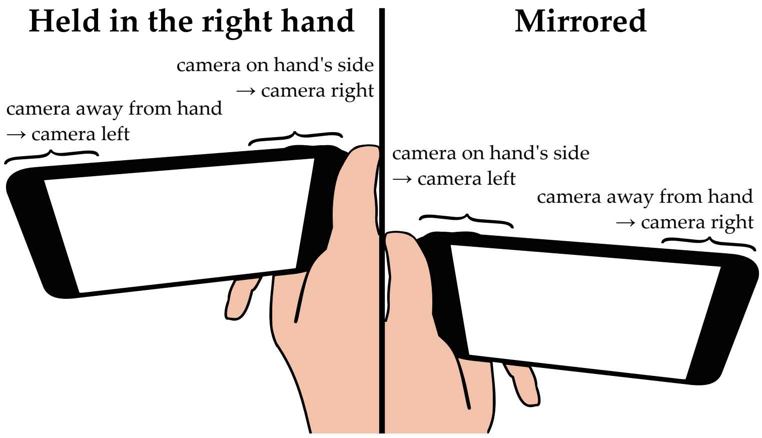


Figure 3.5: When the smartphone was held in the right hand, we mirrored the naming of the *orientation* condition.

3.3.2 Participants

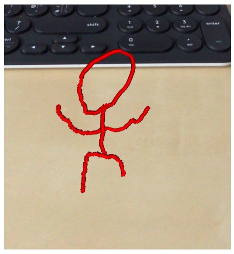
18 persons (male: 12, female: 4, n/a: 2) with an average age of 25.7 years (SD = 3.0 years) participated. One participant was left-handed. The remaining ones were right-handed. One right-handed participant preferred to hold the smartphone in his dominant hand.

3.3.3 Apparatus

We used one iPhones 6s and one iPhone 7 Plus in the study. The iPhone 6s has a size of 138.3 mm × 67.1 mm × 7.1 mm at a weight of 143 grams. It has a 4.7 inch touchscreen (1334 px × 750 px), Apple's A9 chip and a 12 MP camera with a f/2.2 aperture is build in. The iPhone 7 Plus is 158.2 mm × 77.9 mm × 7.3 mm big with a 5.5 inch display with a resolution of 1920 px × 1080 px and weights 188 grams. The device runs with an A10 chip and owns a 12 MP wide-angle camera with a f/1.8 aperture.

3.3.4 Task

The participants were using the ARPen in a sketching application in which they only were able to draw three-dimensional lines. The task of the study was divided into two parts. In the first part, the participants were allowed to draw freely into the AR scene. If the participants liked to, the conductor suggested different motives like several 3D shapes or writing into the air. For the second task, the conductor asked the participants to swipe over the display and fill out the area they can reach with a free finger of the smartphone holding hand. The participants were asked to inform the conductor when they finished one part of the task. During the second task, no visual feedback was provided to avoid that the participants try to reach areas on the screen just for aesthetic reasons.



Participants should draw freely and swipe over the screen after that.

3.3.5 Study Procedure

In the beginning of the study, the conductor explained the purpose of the study, how the ARPen works and how the tasks look like. Then, the participants got the chance to try out the pen in the sketching application. In the beginning of each condition, the smartphone was placed on a table in front of the participant and the conductor named the orientation the smartphone should be held for the following task. Since the location of the camera is not visible from the front side of the smartphone, the landscape orientations were described by the side of the iPhone's home button ('home button left' was used to describe *camera right*). When the participants completed the first part of the task, the conductor took a photo of the smartphone grasp and started the 'touch recording mode' on the smartphone, which was indicated by a white frame around the screen. Then, the participants were asked to perform the second part of the task. When they were done, the conductor stopped the recording, deleted the 3D drawing and placed the smartphone again on the table. After repeating the task with all orientations, the conductor swapped the smartphones and the task was repeated. Finally, the participants were interviewed. During the study, no UI elements were shown on the device screen, such that only the camera image was visible on the screen. The reason for this was the fact that the position of UI elements should not influence the participant's grasp. Additionally, we wanted to avoid a possible aversion of the participants to swipe over those elements in the second part of the task. A double-tap gesture opened a dialog to start recording the touches or to clear the scene.

3.3.6 Measurements

We interviewed the participants, took a photo of their grasps and recorded the reachable areas on touchscreen.

As a result of the first part of the participant's task, we used the photos taken after the sketching task to categorize the different types of grasps. The sketches were not recorded, because this task was only used to distract the user of how they actually grasp the smartphone. If the participants were not able to lift the touching finger off the screen or the smartphone fell down during the task, those grasps were counted as *invalid* and were not considered in the reachability analysis. Touch points were mirrored for participants who preferred to hold the smartphone in the right hand. Furthermore, touch points created by the double-tap gesture were removed. In the interview, we asked the participants which orientation and smartphone size they preferred. We also asked to rank the different orientations. Appendix A shows the user interface the conductor used to gather the data and also includes the asked questions.

3.3.7 Results

Portrait and *camera right* were the most preferred orientations.

Orientation. No participant made a different rating for *orientation* depending on the device size. Thus, Figure 3.6 (left) describes the rankings for both sizes. *Camera left* was ranked worst (14 'worst' rankings). The participants commented that using this orientation made them covering the camera for the most grasps. Therefore, they had to find other less usual grasps. Both, *portrait* and *camera right* were often ranked on the first place (portrait: 9, *camera right*: 8).

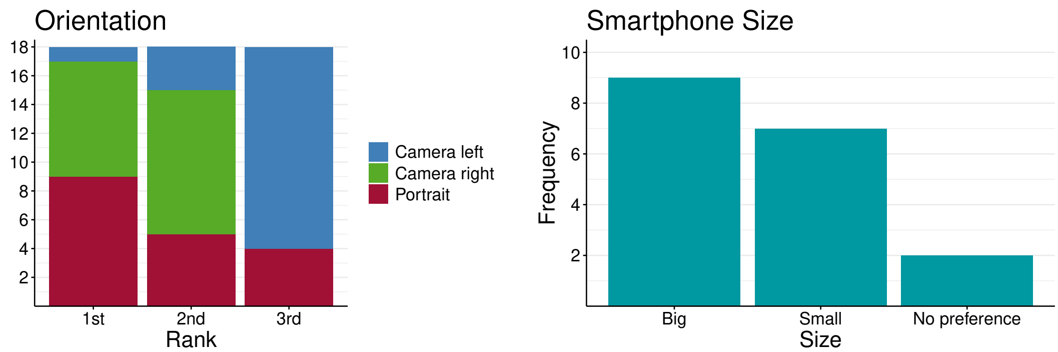


Figure 3.6: User preferences for *orientation* (left) and *smartphone size* (right): The *orientation* was rated equally for both smartphone sizes. The participants preferred using the smartphone in *portrait* and *camera right*.

However, *camera right* was never rated as worst, while *portrait* got four 'worst' ratings. The reason for this, was the limited horizontal viewport which often led to tracking errors during the sketching task.

No clear preference for *smartphone size*.

Smartphone size. The preferences regarding the device size are very balanced (cf. Figure 3.6, right). Nine participants preferred the 5.5 inch smartphone because of the bigger screen and a (rather subjective) wider camera image which facilitate keeping the pen's markers in the viewport. The participants often commented that they preferred this size because they are used to it. On the other hand, seven participants preferred the smaller smartphone. They argued that this size is more comfortable to grasp and that the smartphone weight less. The remaining participants had no preference.

Grasp. Figure 3.7 illustrates all valid grasp categories we found in the study. For the landscape orientations, we were able to categorize the grasps into four groups: *pinkie* (a), *thumb tray* (b), *frame* (c) and *front*(d):

Pinkie

In this grasp the smartphone is placed on the pinkie, while the remaining fingers were extended in the back.

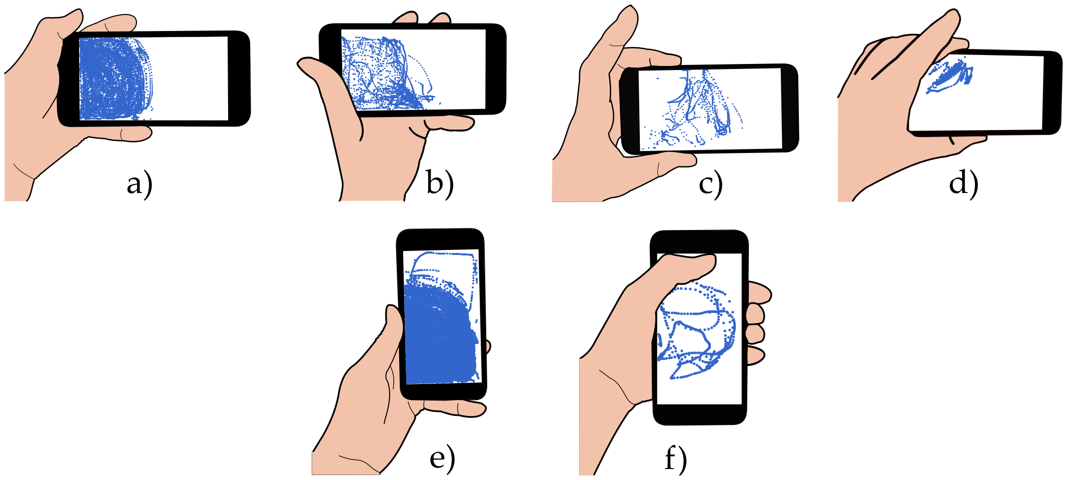


Figure 3.7: Grasp categories and the corresponding touch points (including both *smartphone sizes*) we observed in the study: *pinkie* (a), *thumb tray* (b), *frame* (c), *front* (d), *low portrait* (e) and *high portrait* (f).

Thumb tray
Thumb tray is similar to *pinkie*, but using this grasp the smartphone is placed one the pinkie and the thumb tray.

Frame
For this grasp, the smartphone is held from the side, thumb and middle finger form a frame around the smartphone.

Front
For the *front* grasp, the smartphone is held mostly with the thumb, the index and the middle finger laterally from the front.

Pinkie was used very often in the *camera right* orientation ($\geq 50\%$, cf. Table 3.1). For *camera left*, it was used less often because the smartphone camera was often occluded by the user's hand. Thus, the participants evaded this grasp which resulted in a more frequent use of *thumb tray* in this condition.

		Small	*Big*
Camera right	*Pinkie*	50%	61.1%
	Thumb tray	16.7%	11.1%
	Frame	11.1%	11.1%
	Front	0%	5.6%
	Invalid	22.2%	11.1%
Camera left	*Pinkie*	27.8%	27.8%
	Thumb tray	33.3%	27.8%
	Frame	11.1%	16.7%
	Front	11.1%	16.7%
	Invalid	16.7%	11.1%
Portrait	*Low portrait*	100.00%	88.9%
	High portrait	0.00%	5.6%
	Invalid	0.00%	5.6%

Table 3.1: Grasp frequencies for different orientations and sizes. *Pinkie* and *low portrait* are used most often.

For the portrait condition, participants used two types of grasps. Depending on the height the smartphone is held, we categorize the grasp as *low portrait* (smartphone is held on the bottom, cf. Figure 3.7e) and *high portrait* (smartphone is held above the half height, cf. Figure 3.7). *Low portrait* was used every time with the *small* smartphone and in 88.9% of the cases for the other device (cf. Table 3.1). *High portrait* was used only once.

A low smartphone grasp was used in the most cases for the *portrait* orientation.

Reachable area. For the landscape orientations, we could also observe that even if the grasps differ physiologically, their reachable areas are located and sized very similarly. As observable in Figure 3.7, the reachable areas of *pinkie* and *thumb tray* are located in the bottom left, while the area of *frame* and *front* are located in the top center. Therefore, we grouped those grasps together to calculate the most general reachable areas for both landscape and portrait orientations. For simplicity, we describe the interaction area as rectangle which is calculated from the average boundaries of the participants' touch areas in x and y direction. The location of the different touch regions are described with the origin at the bottom left edge of the touchscreen.

Touches of *pinkie* and *thumb tray* are located near the bottom left edge.

For *pinkie* and *thumb tray*, the touch locations are located near the bottom left edge at (30.4 mm, 28.1 mm) for the *small* smartphone (*big*: (28.3 mm, 33.1 mm)) with a width of 59.2 mm (*big*: 50.9 mm) and a height of 52.2 mm (*big*: 56.0 mm). The other landscape grasps were more located in the center around (46.3 mm, 38.1 mm) for the small device (*big*: (46.0 mm, 45.0 mm)), but have slightly smaller average width and height of 51.7 mm (*big*: 53.0 mm) and 43.9 mm (*big*: 41.1 mm). The touch points for *low portrait* are located around (27.9 mm, 35.3 mm) (*big*: (28.7 mm, 36.0 mm)) and had a width of 54.0 mm (*big*: 54.4 mm) and height of 66.7 mm (*big*: 68.3 mm). Finally, for *high portrait*, which was only used once, the touches are located around (31.0 mm, 59.7 mm) and has an average width and height of 58.9 mm and 78.3 mm.

3.3.8 Discussion

The results of this study are not unambiguous in every case. For example, we could not find a clear preference for *smartphone size*. Instead, there is only a small tendency for the bigger smartphone (with a difference of two participants). However, even if the *big* smartphone was heavier, it suggested a wider camera angle and thus, the participants had the feeling of keeping the ARPen within the view much easier.

The participants had also very different preferences regarding the orientation of the smartphone. While *camera left* (camera on the hand's side) was ranked worst because of the occlusion of the camera, *camera right* and *portrait* were often placed on the first place. While no participant ranked *camera right* as worst, four participants preferred *portrait* the least because of the difficulties keeping the ARPen inside the viewport.

Based on those findings, we used the *big* smartphone in landscape orientation in following ARPen studies.

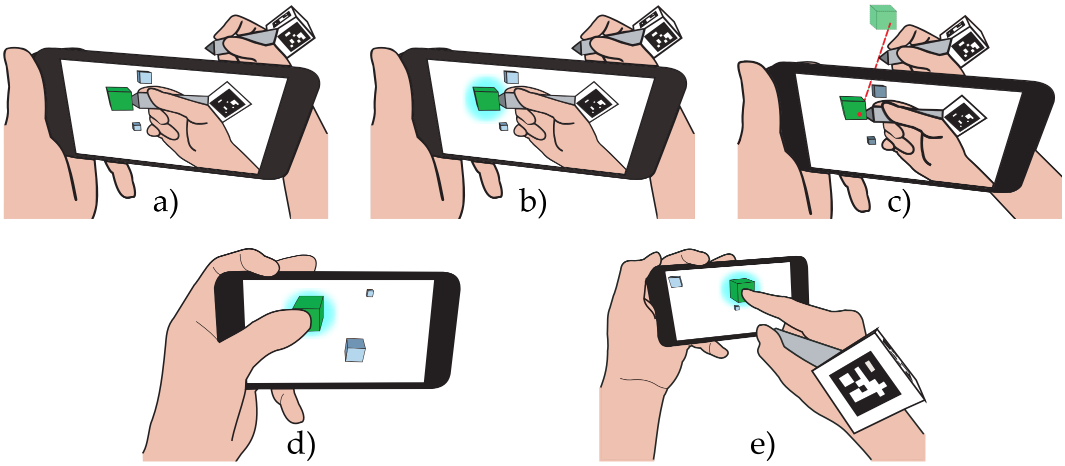


Figure 3.8: Selection techniques investigated by Wacker et al. [2019]: Direct pen selection *without* (a) and *with highlight* (b), *pen ray* (c), *one-handed* (d) and *two-handed touch* (e) selection.

3.4 Excursus: Target Selection Using the ARPen

We decided to focus on context menus in Section 3.2.1 "Application and Context Menus". In contrast to application menus, context menus usually appear once the user selected an object and used a specific trigger to open it (e.g., a right click on desktop computers).

Wacker et al. investigated selection and translation techniques using the ARPen.

In a – to the date of this thesis not appeared – work, Wacker et al. [2019] investigated various techniques for selecting and translating virtual objects in an ARPen application. Figure 3.8 illustrate the different selection techniques. The authors compared techniques which are actuated with the pen (Figure 3.8a,b,c) and techniques which use the touch capabilities of the smartphone (Figure 3.8d,e), too.

For the direct selection *with* and *without highlight* (a, b) the user needs to find the 3D location of a cubic target and press on a button on the ARPen when the pen's tip is inside the target. Those conditions only differ by the feedback provided when the tip is inside the target. When using *pen ray* (c), the tip only has to be perspectively into the object (also named 'Sticky Finger' Pierce et al. [1997] or 'occlusion selection' Bowman and Wingrave [2001]). For the touch selections, the participants could select an object by touching on its projection on the screen with the thumb of the smartphone-holding hand (*one-handed touch*, d) or with the pen-holding hand (*two-handed touch*, e). The advantage of the touch techniques as well as *pen ray* is that the user does not have to estimate how distant the target is.

In the study investigating the selection techniques, the participants should select one of 64 differently sized virtual cubes which were arranged into a $8 \times 8 \times 8$ grid. The authors observed *selection time*, *success*, the *projected size* of the target on the display and the *deviation* to the target.

The study showed that the touch conditions were the fastest (*two-handed touch* significantly faster than each non-touch technique), followed by *pen ray* which was significantly faster than the direct pen selection techniques. However, *pen ray* had the highest success rate (significant to *one-handed touch* and selection *without highlight*) and a small projection size which indicates, as claimed by the authors, that the smartphone has not to be moved a lot for accurate selection.

We use *pen ray* for mid-air selection in our studies.

Since the *pen ray* technique performed quite fast with a high success rate, we decided to use this technique to open the menu with the ARPen in following user studies.

One-handed touch seems to be the most appropriate technique for context menus.

For the touch menus, on the other hand, *two-handed touch* seems to be not quite appropriate for context menus. Usually, a context menu is opened for a very short time, e.g, for actions like duplicating an object. Using *two-handed touch* for that, would require to move the pen-holding hand from the mid-air interaction area to the screen for the menu interaction and then, back to the area behind the smartphone to continue the task. Because of this and the not existing significant differences regarding the performance of the touch selection techniques, we decided to use *one-handed touch* for our touch menus.

3.5 Finalized Menu Techniques

After finding appropriate ways of opening a menu, the appearance and the behavior of the menus needs to be discussed. For context menus, the two most often used menu types are linear and pie menus. While linear menus are very common when using desktop computers or smartphones, there is no standard menu type in Augmented and Virtual Reality.

3.5.1 Menu Appearance and Behavior

In the user study in Section 3.3 "Study 1: Smartphone Grasp & Interaction Area", we found that *pinkie* or *thumb tray* were used most frequently (72.2% for the *big* smartphone). Thus, we placed the touch menu on the left side within 50.9 millimeters such that the thumb can be used to select the menu items comfortably. Linear menus are well known on a smartphone for the users and they have the advantage that the menu items are distributed vertically, which keeps the interaction area near the thumb. For those reasons, we decided to use this menu type for the touch menu.

We implement the touch menu using a linear layout.

In contrast to the touch menu, using the AR menu, the user has much more freedom of movement due to the fact that she can use the complete arm for the input, instead of only her thumb. Since pie menus perform better than linear menus as shown for example by Callahan et al. [1988], Das and Borst [2010] and Komerska and Ware [2004] and the ease of movement with the ARPen, we decided to implement the AR menu as pie menu. Figure 3.9 shows the final design of both the AR menu (left) and the touch menu (right). For both, the size of the items is adapted to the available space such that the item height (opening angle for the pie menu) increases with a decreasing item number.

The AR menu uses a pie layout.

Besides the visual appearance of the menus, also the location and hierarchy structure need to be weighed carefully. While the touch menu will be view fixed and thus, always shown directly on the display (cf. Section 3.2.2 "Menus in an ARPen Application" (p. 24)), the AR menu can be located everywhere within the 3D world. For menus in Virtual Reality, Das and Borst [2010] found that item selection is faster for menus which appear near the target object ('contextual location') than for menus which always open at the same place. Building on this, we decided to place the menu directly at the location of our pen tip for the AR menu.

The pie menu appears around the ARPen's tip.

Regarding the design of menu hierarchies, a menu of a higher hierarchy level also shows up directly around the pen tip for the pie menu. To avoid an overlapping of the new appearing menu and its parent, all predecessor menus are shifted back along the z-axis as described by Gebhardt et al. [2013] (cf. Figure 3.9, left). For the touch menus, we also followed the strategy of keeping the menu near the location of the the thumb. To reach this, the new menu moves from the right to the parent's location (Figure 3.9a) and overlays the origin menu completely as shown in Figure 3.9b.

The pie menu's center is used to revisit previous menus.

Both menus support to go back to an earlier hierarchy level. As recommended by Gebhardt et al. [2013], we keep some space in the middle of the pie menu (the 'dead zone' [Gebhardt et al., 2013]) and place a circular button with a 'back' label there (cf. Figure 3.9, left). When the user selects this button, the menu fades out, and the predecessor menus shift to the front. Moreover, the user can go back to a predecessor menu by clicking at a not occluded part of it.

Touch menus use a right-swipe or a 'back' button to traverse back.

Going back is solved similarly for the touch menu. For this menu, a back button is placed on the top of the menu (cf. Figure 3.9, right). Additionally to that, we adapted Apple's left-edge swipe gesture[4] for the menu. This animated gesture inverses the 'appearing menu' animation and should feel like shifting the menu back to its origin place.

[4] https://developer.apple.com/documentation/uikit/uinavigationcontroller *(Accessed: 22.02.2019)*

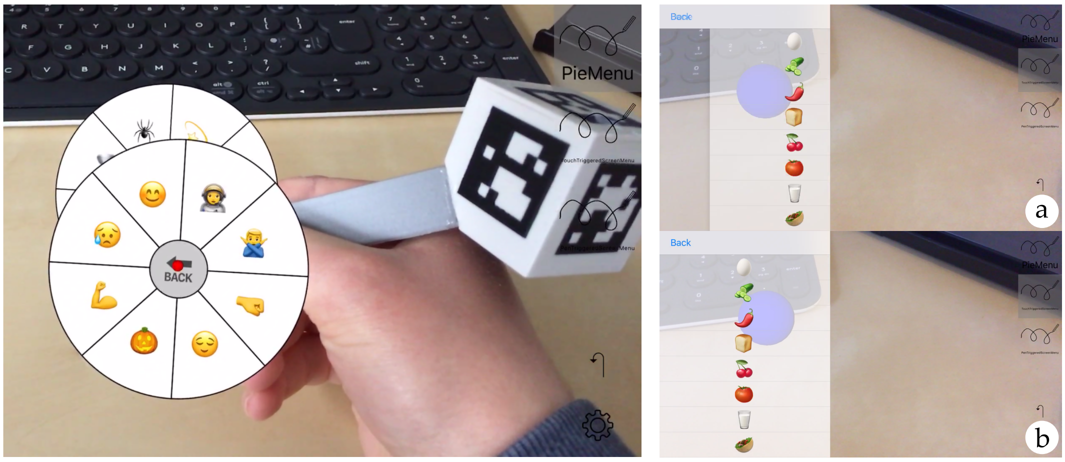


Figure 3.9: Menu layouts for the AR menu (left) and the touch menu (right). The pie menu of the next hierarchy level always appears directly around the pen tip (red dot) in front of its parent. A new touch menu shifts from the right to the left (a) and occludes the previous menu completely (b). The blue ball is the initial target which was selected to open the menu (occluded of the menus when using the pie menu).

All animation durations were adapted to the animation duration of the UINavigationController of 0.51 seconds in average.

The height of the menu items of the touch menu is constrained by the number of items and the size of the smartphone. However, this matches to the standard size of a table row in iOS when using the iPhone 7 Plus in the following study. For the pie menu, we decided to adapt the size of the menu in a way that it is completely visible from the nearest position the ARPen is still trackable. This corresponds to an outer radius of 4.5 cm. The back button has a diameter of 1 cm. However, in comparison to the touch menu, the pie menu has the advantage that the user can move the smartphone nearer to the menu and which leads to a higher projected size of the menu.

3.5.2 Resulting Techniques

We investigate a touch-only, a pen-only and one technique using both input types.

In the following study, we want to compare the two menus designs using the different available input types. For this, we combine menu-triggering techniques and our menu designs which results in a *menu technique*. Additionally to one technique only using touch input and one technique using only mid-air input with the ARPen, we also investigate a third technique mixing both input types. Since the main interaction with an virtual object is done using the pen in an ARPen scenario, when using this technique triggering the menu is done using *pen ray*, but the following item selection is done using touch input. In summary, we will compare the following menu techniques (cf. Figure 3.10):

AR pie menu (*PIE*)
: A pie menu opens around the ARPen's tip within the 3D world. Both menu activation and item selection is done using the *pen ray* technique.

Touch-triggered screen menu (*TSM*)
: To open the linear *touch-triggered screen menu*, the user selects the target by touching on its projection on the screen. The following item selection is done via touch input.

Pen-triggered screen menu (*PSM*)
: The *pen-triggered screen menu* is a linear touch menu which is shown directly on the smartphone. The menu opens when the target is selected via *pen ray*.

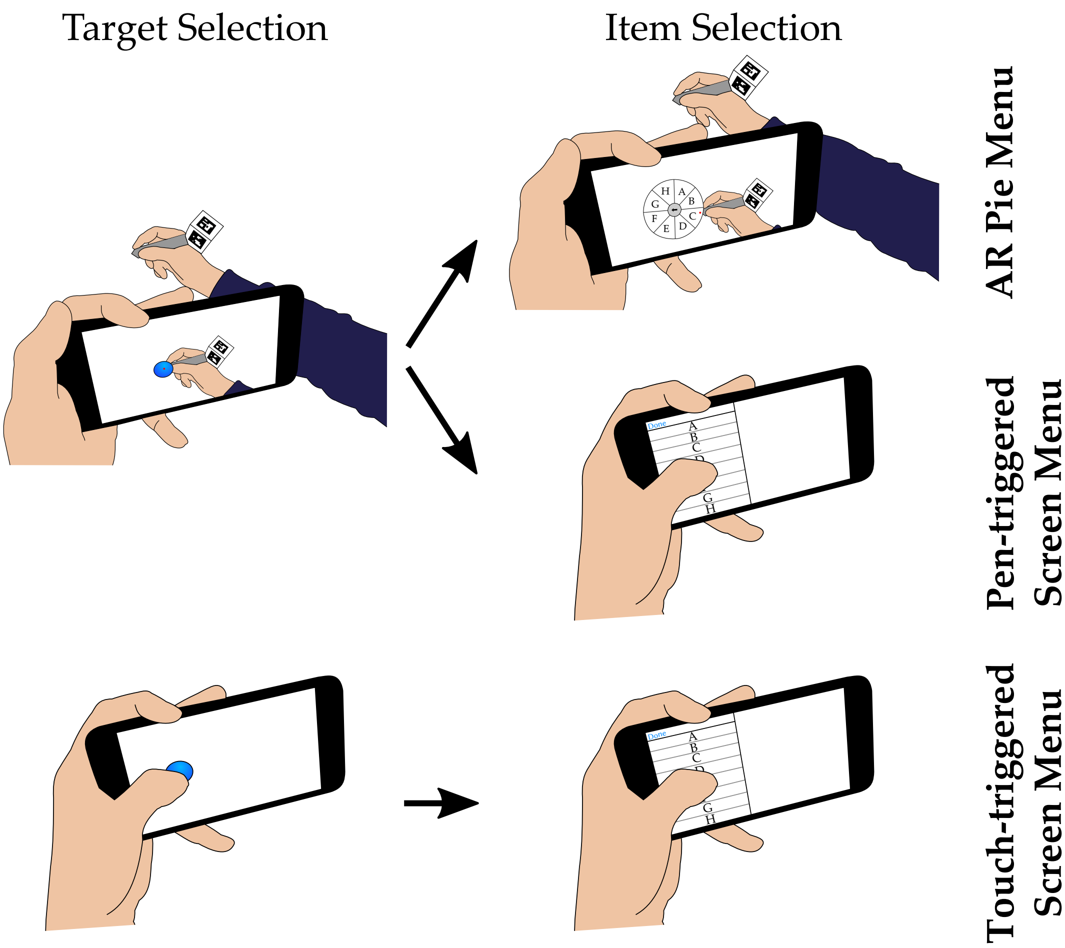


Figure 3.10: The three menu techniques investigated in the following study. *AR pie menu* uses the ARPen for both, triggering the menu and selecting an item in the resulting pie menu. For the *pen-triggered screen menu*, opening the menu is also done using the pen, but consecutive item selection is done via touch input. *Touch-triggered screen menu* uses for both tasks touch input.

Chapter 4

A Comparison of Mid-Air AR and Touch Menus

In this chapter, we describe the user study we conducted to compare the the presented menu techniques to each other.

4.1 Experimental Design

For the study, the participants should select two items of a 2-level menu containing eight items labeled with various emojis. The participant should select the target emojis using one of the three *menu techniques* (*AR pie menu*, *touch-triggered screen menu* and *pen-triggered screen menu*). The target emoji for each level was determined by the *item position*.

For each trial, the emojis were sampled randomly from the available emojis in iOS 12. However, we removed flag emojis and only picked one randomly selected representative emoji from emoji groups with a very similar appearance as, for example, the zodiac emojis. We did this to keep the emojis easy to describe and to memorize. No emoji appeared multiple times within the two levels of one menu which guaranteed that the target emoji could not be found

We removed flag emojis and emojis having a very similar appearance from our sample set.

on the second hierarchy level if the participant selected a wrong emoji on the first level.

Menu breadth and depth were adopted from related work.

The *menu breadth* (number of items within one menu level) and the *menu depth* (number of hierarchy levels) were chosen because this configuration corresponds to realistic application scenarios of context menus. Additionally, research has shown that menu performance with this (and similar) configuration still allows accurate and fast item selection [Das and Borst, 2010, Kiger, 1984, Kurtenbach and Buxton, 1993, Miller, 1981].

We counterbalanced *menu technique* using a Latin Square. *Item position* was randomized. Originally, each participant should do 64 repetitions such that every combination of item positions was done by each participant (eight items on the first and eight on the second level). However, participants of a pilot study commented that the number of repetitions was too high which led to fatigue of the arms. Because of this, we reduced the number of repetitions per condition to 32.

4.2 Participants

We conducted the study with twelve participants. The participants were 22 to 27 years old (mean: 24.42 years, sd: 1.38 years). Three participants were female, the other nine male. All participants were right-handed.

4.3 Apparatus

As result of our preliminary study, we used the iPhone 7 Plus for this study. The iPhone has a size of 158.2 mm × 77.9 mm × 7.3 mm with a weight of 188 grams. The display has a resolution of 1920 px × 1080 px on a 5.5 inch display. The iPhone has a 12 MP wide-angle camera with a f/1.8 aperture.

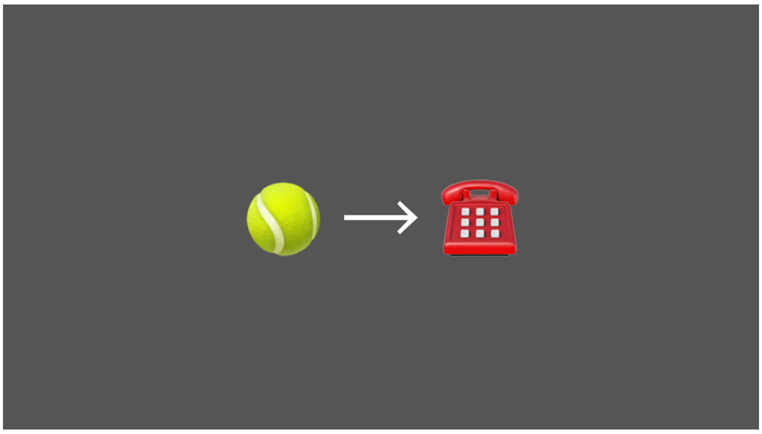


Figure 4.1: Layout of the second smartphone to present the next target emojis. The participant should select the left emoji on the first hierarchy level and the right emoji on the second.

Additionally to the iPhone 7 Plus, we used another iPhone 6S to display the target item labels. This iPhone, has a 4.7 inch display with a resolution of 1334 px × 750 px.

4.4 Task

The participants had to open the context menu and then perform two item selections.

The participants should open a menu and select the two target emojis shown on the second smartphone. Figure 4.1 shows how the emojis were presented on the device. The participants were asked to memorize the two target emojis. Nevertheless, we did not hide the target emojis when the user started the selection. Then, they should open the menu by selecting a blue ball which has a radius of 2 cm and select the menu items containing the target emojis.

In the case, that the participant moved the pen outside the trackable area, the virtual ball representing the pen's tip disappeared. When the user moved the pen such that the markers are within the viewport, the tip appeared again. If the user selected a wrong item on the first level such that the second emoji was not inside the appearing submenu, she needed to do a backstep and correct the last selection.

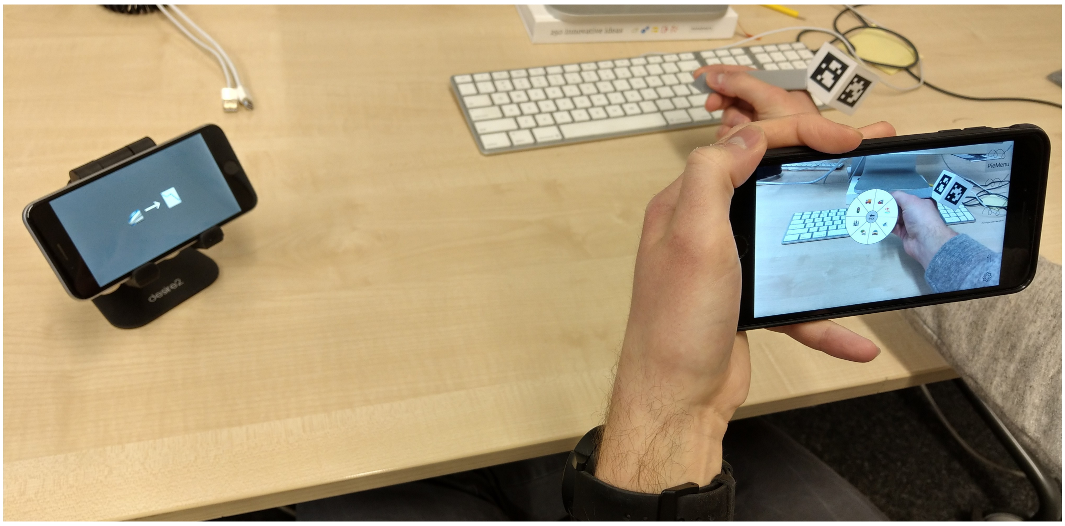


Figure 4.2: Setup of the second study. The participants should perform the task without having the second smartphone within the viewport.

4.5 Study Procedure

The participants should hold the smartphone using the *pinkie* grasp.

In the beginning of the study, the conductor introduced the ARPen to the participant, explained how it works and demonstrates the first menu technique and the task. As shown in Figure 4.2, the smartphone showing the target emojis was placed to the left of the participant. The participant should be able to comfortably look at the target emojis without having them within the viewport of the smartphone running the AR application. Additionally, the participants were asked to hold the smartphone using the *pinkie* grasp to avoid the grasp to possibly influence the performance of the menu technique.

Before we started recoding the item selections, the participants had the chance to try out the current menu technique. After the participant did all selections, the participants answered some questions regarding the current menu on the questionnaire. Then, the conductor explained the next technique and the participant repeated the task using this technique. In the end, the participant ranked the menu techniques.

4.6 Measurements

During the study, we recorded *selection time* for each item selection. The first time measurement started when the menu was triggered and stopped with the first item selection. The second measurement, then, was measured between the first and the second selection. Additionally, we observed how many *backsteps* the user made and the *success* of the trial. If a trial was not successful, we recorded the following types of error:

We recorded *selection times*(for *hierarchy level* and *overall*), *success*, the number of *backsteps* and *subjective ratings*.

Wrong item
: The first selection was correct, the second item was not.

Outside menu
: The user made an selection outside the bounds of the menu.

Accidental close
: The user made a step back in the first hierarchy level.

We differentiate between three types of error.

Additionally, the users rated the condition for *subjective accuracy* of item selection, *comfort* and the *ease of search* on five-level Likert-Scales and ranked the *menu techniques* from 1 (best) to 3 (worst). The questionnaire we used for this study is attached in Appendix B "Study 2: Questionnaire".

In summary, we recorded 32 trials for each of the 3 *menu techniques* with 12 participants which results in 1152 total trials.

4.7 Results

We analyzed *selection time* using a repeated-measure ANOVA with the user as random variable. *Success*, *backsteps* and the *subjective ratings* were analyzed using Friedman tests. For the evaluation, we counted the number of *backsteps* and *success*, and averaged the log-transformed selection times. For the analyses of *selection time*, we removed the trials including 1 or more backsteps (12 trials). Paired-sample t-tests with corrections using the Holm method were used for the post-hoc analysis.

The *AR pie menu* was significant slower.

Overall selection time. *Menu technique* had a significant effect on the *overall selection time* ($F_{2,22} = 153, p < 0.001$). The post-hoc test revealed that *AR pie menu* was significant slower (mean (M): 4.52 s, standard deviation (SD): 1.78 s) than *touch-triggered screen menu* (M: 3.08 s, SD: 1.1 s) and *pen-triggered screen menu* (M: 3.06 s, SD: 1.04 s). No other significances were found.

Level selection time. We also found significant effects of *menu technique* ($F_{2,22} = 127.51, p < 0.001$) and *hierarchy level* ($F_{1,11} = 28.12, p < 0.001$) on *level selection time*. *AR pie menu* performed significantly slower than *TSM* and *PSM* ($p < 0.001$, both), and selections were significant faster on the first hierarchy level for this technique ($p < 0.001$). We also found a significant interaction ($F_{2,22} = 16.84, p < 0.001$). While the repeated-measure ANOVA stated a significant effect of *hierarchy level* on *level selection time*, the

Technique	*Level*	Significance		
PIE	$1st$	A		
PIE	$2nd$		B	
TSM	$1st$			C
TSM	$2nd$			C
PSM	$1st$			C
PSM	$2nd$			C

Table 4.1: Interaction significances of *menu technique*×*hierarchy level*. Entries which are not connected with the same letter are significant different.

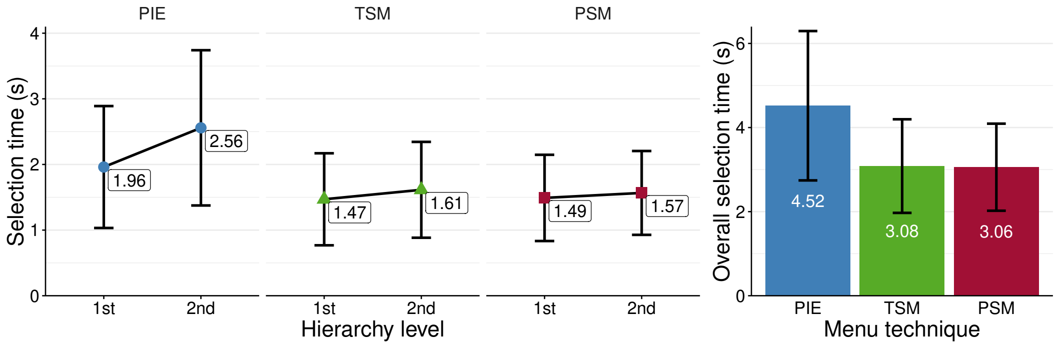


Figure 4.3: Selection times of each menu technique per hierarchy level (left) and overall (right). The bars show the standard deviation. *AR pie menu* performed significant slower overall and for each level.

	Success		*Backsteps*
	Mean	SD	
AR pie menu	99.22%	1.41%	3
Touch-triggered screen menu	97.92%	3.08%	3
Pen-triggered screen menu	98.44%	1.63%	6

Table 4.2: Success rates and number of backsteps of the different menu techniques. For each technique, we measured 384 trials.

post-hoc test revealed only significant differences with the *AR pie menu* for the two levels ($p < 0.001$). Pairwise comparisons showed significant differences for each combination of *menu technique* × *hierarchy level* on (*PIE*, $1st$) and (*PIE*, $2nd$) ($p < 0.01$ for (*PSM*, $2nd$) vs. (*PIE*, $1st$) and (*TSM*, $2nd$) vs. (*PIE*, $1st$), otherwise $p < 0.001$). The result of the post-hoc test is shown in Table 4.1. Figure 4.3 shows the average selection times per *hierarchy level* (left) and *overall* (right).

Success and Backsteps. No errors of the types *outside menu* and *accidental close* occured. However, we observed 12 *wrong item* errors. Table 4.2 shows the success rates and the number of backsteps the participants made for each *menu technique*. A Friedman test did not revealed any significance.

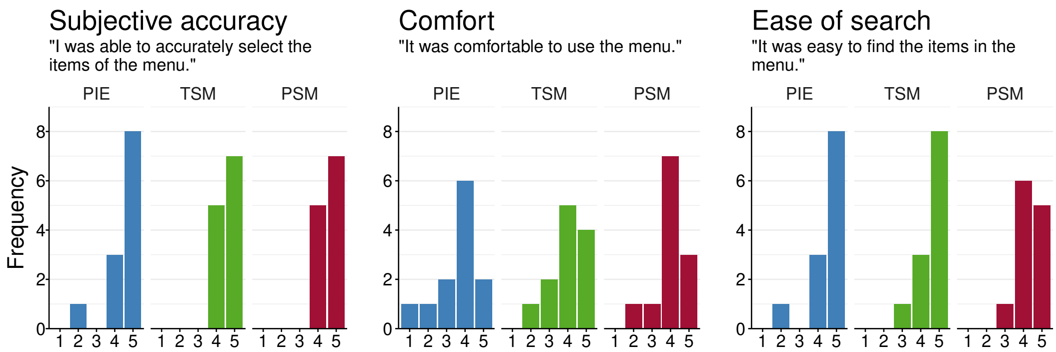


Figure 4.4: Distribution of the ratings for *subjective accuracy*, *comfort* nor *ease of search* (5 = 'totally agree').

Subjective ratings were similar for all techniques.

Subjective ratings. The preferences regarding *subjective accuracy*, *comfort* nor *ease of search* did not differ very much between the different menu techniques. Figure 4.4 shows how the participants rated each of the techniques. Statistical tests also showed no significant differences of *menu technique* on neither *subjective accuracy*, *comfort* nor *ease of search*.

PIE and *TSM* polarized our user group.

The rankings for the *menu techniques* were also very divided. As shown in Figure 4.5, are both first and last place dominated by *AR pie menu* (first: 6, last: 5) and *touch-triggered screen menu* (first: 4, last: 6). *Pen-triggered screen menu*, on the other hand, was only ranked twice on the first and once on the third place and, thus, was ranked in 75% if the cases on the second place. All participants ranked *TSM* on the first place, ranked *PIE* on the last place. The other way around was *TSM* ranked on the last place by five of six participants preferring *PIE*.

Comments. The participants commented that it was uncomfortable to keep the arm raised when using *AR pie menu* (four comments) and *pen-triggered screen menu* (three comments). For the screen menu techniques, four participants mentioned that the lower items were hard to reach. Three participant commented that the grasp for the technique was uncomfortable and that the smartphone was too heavy.

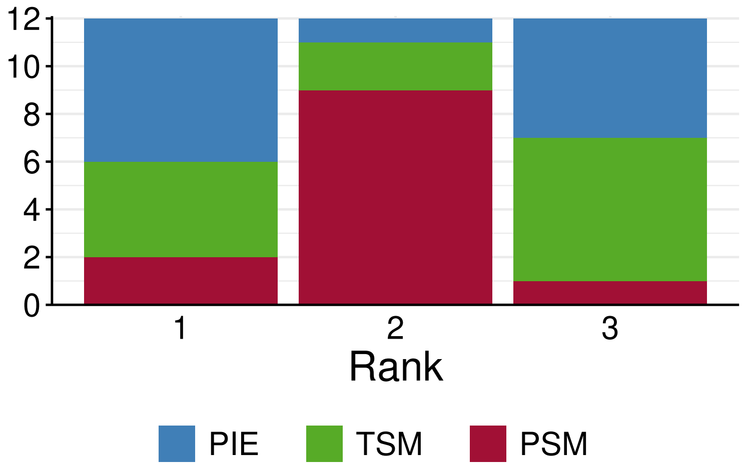


Figure 4.5: Ranking of *menu techniques*. *PIE* and *TSM* were ranked very similarly having the most rankings on both, the first and the last place.

4.8 Discussion

Although pie menus are highly recommended in research, the touch menus performed significantly faster than the *AR pie menu* by having similar high success rates. Splitting the task into the item selection on the first and the second hierarchy level has shown that even if we disregard the effect of the menu hierarchy, the *AR pie menu* performed slower. Thus, only considering the performance aspects of the menu, the user benefits more from the type of input used for the menu rather of its layout. While we could not find any significant differences between the two *hierarchy levels* on the selection time for both *touch-triggered screen menu* and *pen-triggered screen menu*, this was the case for *PIE*. One reason for that might be that the menu for the second level did not appear at the location of the first menu. The always new locations might surprised the users and forced them to process the new location first, before continuing with the item selection. Two users' also mentioned clearly stated that they would prefer the menu to be always at the same location. However, further investigations exchanging the hierarchy implementation need to be done to find the cause of this performance difference.

The changing position of the second *AR pie menu* might cause the slower selections.

Even if using touch input was faster, we cannot provide clear recommendations for the techniques.

Regarding the participants' preferences, the *AR pie menu* and the *touch-triggered screen menu* were polarizing. When one of both was ranked as best, the other one was most often ranked as worst. Since *pen-triggered screen menu* was only ranked twice as best, we can not give a clear recommendation which of the menu techniques is the best candidate to use in an ARPen application.

4.9 Limitations

Unfortunately, when we analyzed the gathered data, we found a bug in the software which resulted in a repetition of all trials of one menu technique when one trial was reset. In summary, we recorded 133 trials multiple times of which 56 were at *PIE*, 48 at *TSM* and 29 at *PSM*. For our evaluation, we searched for the duplicate pairs (technique, target positions), took the earliest one and discarded the remaining duplicates. Selecting the data points this way ensures that we got as much data points from the beginning phase of the item selection as possible and also data points from the phase when the user got more advanced with the technique. Nevertheless, we analyzed whether the position of a selection had an effect on the selection time using a repeated-measure ANOVA and we did not found any significant differences for both, the origin data set and the sampled set.

However, the purpose of this study was to find out whether menus interaction should take place mid-air or on the smartphone in our bimanual handheld AR scenario. For this, we selected – based on the literature – a menu breath of eight on two hierarchy levels. Varying both could lead to clearer recommendations for the type menu in such an application. We also decided to draw the AR menu always around the ARPen's tip while *TSM* and *PSM* always stayed at the same location. Fixing the location of the AR menu might affect its performance positively. Finally, we found that the fact that the menu is places mid-air or on the touchscreen has an huge impact on the performance.

Chapter 5

Summary and Future Work

5.1 Summary and Contributions

In this thesis, we investigated three menu techniques for the interaction in Augmented Reality using a smartphone on the one hand and a 3D trackable pen – the ARPen – on the other hand. This work aimed to evaluate whether the menu interaction should be done using the ARPen in mid-air or whether the touch capabilities of the smartphone should be used to reach fast, accurate and comfortable menu interactions.

Due to the novelty of this bidirectional handheld AR scenario, we first investigated how people prefer to hold the smartphone when using the ARPen and which areas they then can reach on the touchscreen to determine the location and the size of the presented touch menus. For this, we conducted a user study in which the participants were asked to sketch freely with the ARPen into the 3D space and touch the reachable regions on the smartphone afterward while holding the smartphone in three different orientations. We observed that the participants preferred holding the smartphone using *pinkie* or *thumb tray* for the two landscape orientations. For the portrait orientation, the users clearly preferred the *low portrait* grasp. We decided to continue using the *camera right* orientation due to the high number of 1st place ranking and no placements on the last place. However, one should notice that the portrait orientation was ranked slightly more often on the first place, but also got more rankings on the last place. Using *pinkie* or *thumb tray* and holding the smartphone with the camera on the right, the participants reached in average an area of 5.1×5.6 cm^2 comfortably on the iPhone 7 Plus.

We then presented the results of Wacker et al. [2019] regarding object selection and decided to use *pen ray* and *one-handed touch* to trigger our menus. We combined *one-handed touch* with a linear menu layout for the *touch-triggered screen menu* and *pen ray* with a pie layout for the *AR pie menu*. Additionally, we decided to mix mid-air and touch input in the *pen-triggered screen menu*.

We compared those techniques in a user study with twelve participants. In this study, we asked the participants to trigger a menu and select two items of a two-level menu. All menu techniques had very high success rates. The *AR pie menu* was significantly slower than *TSM* and *PSM* and was the only technique with significant different selection times on the two hierarchy levels. However, the subjective rankings of the participant show no clear preference for one menu. Moreover, we could see that the menu techniques were polarizing our participants. All participants preferring the *touch-triggered screen menu* ranked the *AR pie menu* on the last place. And vice versa, only one participant preferring the pie menu did not ranked *TSM* on the last place. However, the participants rated the techniques similarly regarding the accuracy, the comfort or the ease of search a particular item. Thus, further investigation is needed to identify the reason for this extreme difference in user's preferences.

In summary, we could observe that using touch input increases the selection speed significantly without having contrary effects on the accuracy. Nevertheless, we are not able to provide a clear recommendation for touch menus due to the user's rankings.

5.2 Future Work

In following studies, we plan to investigate the difference of the selection time between the first and the second hierarchy level for the *AR pie menu*. Using different hierarchy implementations might reduce the performance gap between those.

In our second user study, we used a fixed menu breath and a fixed number of hierarchy levels derived from existing research. Since we could not find any statistically significant differences regarding the error rate, varying those factors – especially increasing them – could reveal more general performance differences between our presented menus.

In general, we plan to explore more menu techniques for the ARPen scenario. For example, selection techniques with local menu cursors as proposed by Das and Borst [2010] or Gebhardt et al. [2013] could be compared to our pie menu implementation. Especially combining those techniques with touch input on the smartphone could improve their performance significantly, having the low selection times of the touch menus in mind. Alternatively, using the mechanisms of marking menus as presented for the Command & Control Cube [Grosjean and Coquillart, 2001, Grosjean et al., 2002] could increase the performance of any pie-similar hierarchical menu in such a scenario.

Furthermore, modern AR devices are capable to detect surfaces in the real word. We plan to place our menus on those solid surfaces to observe the effect of the haptic feedback they provide. Again, those menus could be combined using marking mechanisms.

Since the user still has to pay attention to keep the ARPen tracked during the mid-air item selection, we plan to investigate menu techniques which allow mid-air interaction using the 6 DOF tracking of the smartphone. In such a menu, for example, the user could create a view-fixed cursor which, then, is controlled by orienting the smartphone. Such a technique could facilitate interactions near the smartphone which run into the danger of tracking lost.

Appendix A

Study 1: User Interface for the Interview

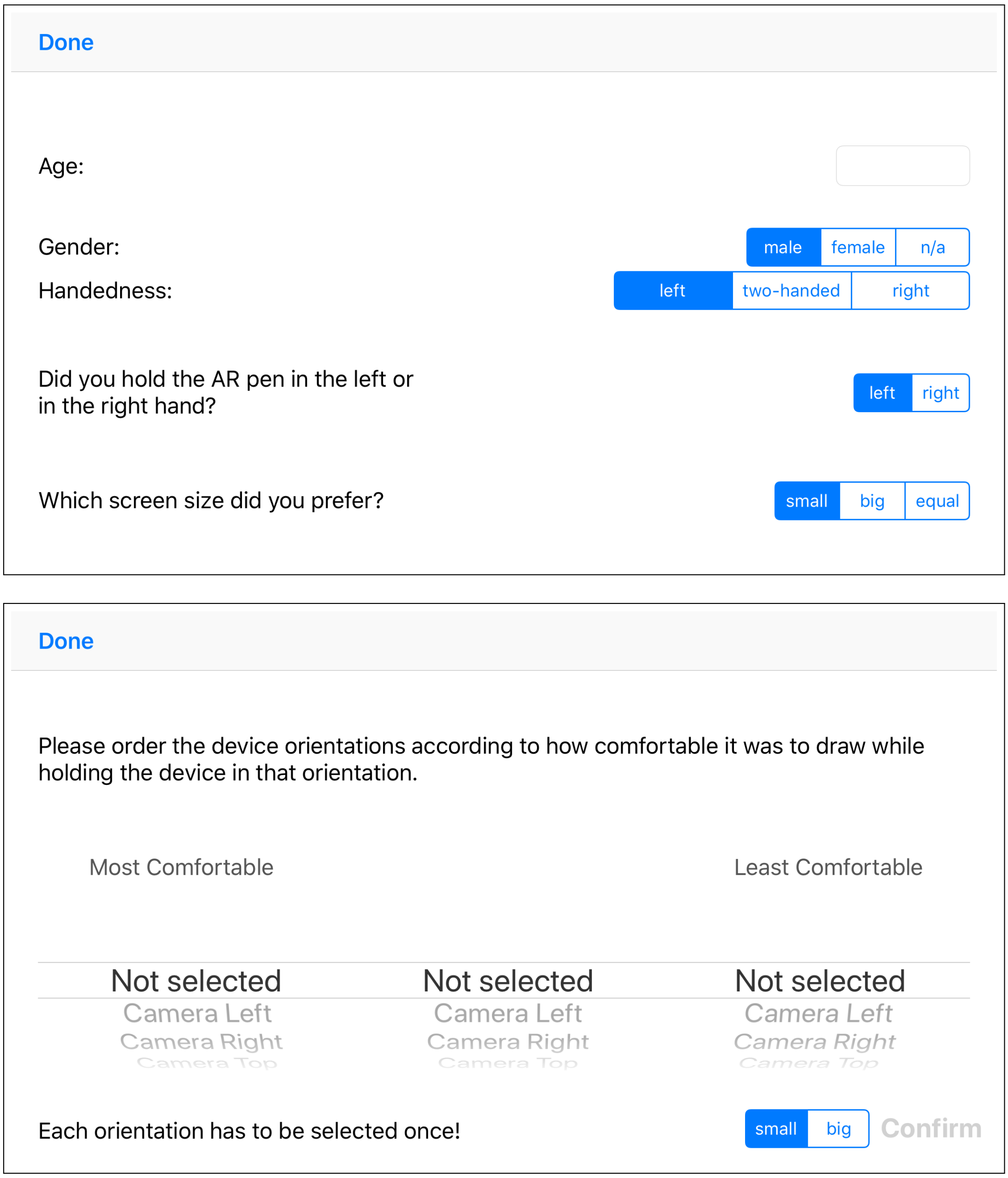


Figure A.1: User Interface for the Interview in study 1

Appendix B

Study 2: Questionnaire

ID:

Age:

Gender: ☐ male ☐ female ☐ n/a

Please rank the menus (1: best, 3: worst).

AR Pie Menu:
Touch-triggered Screen Menu:
Pen-triggered Screen Menu:

AR Pie Menu

I was able to accurately select the items of the menu (on both hierarchy levels).

Totally disagree ☐ ☐ ☐ ☐ ☐ Totally agree

It was comfortable to use the menu.

Totally disagree ☐ ☐ ☐ ☐ ☐ Totally agree

If it was uncomfortable, please describe what was uncomfortable:

It was easy to find the items in the menu (on both hierarchy levels).

Totally disagree ☐ ☐ ☐ ☐ ☐ Totally agree

General Comments:

Figure B.1: First page of the questionnaire (study 2): Subjective information, the menu ranking and the questions about the *AR pie menu*

Touch-triggered Screen Menu

I was able to accurately select the items of the menu (on both hierarchy levels).

Totally disagree ☐ ☐ ☐ ☐ ☐ Totally agree

It was comfortable to use the menu.

Totally disagree ☐ ☐ ☐ ☐ ☐ Totally agree

If it was uncomfortable, please describe what was uncomfortable:

It was easy to find the items in the menu (on both hierarchy levels).

Totally disagree ☐ ☐ ☐ ☐ ☐ Totally agree

General Comments:

Figure B.2: Second page of the questionnaire (study 2): Questions about the *touch-triggered screen menu*

Pen-triggered Screen Menu

I was able to accurately select the items of the menu (on both hierarchy levels).

Totally disagree ☐ ☐ ☐ ☐ ☐ Totally agree

It was comfortable to use the menu.

Totally disagree ☐ ☐ ☐ ☐ ☐ Totally agree

If it was uncomfortable, please describe what was uncomfortable:

It was easy to find the items in the menu (on both hierarchy levels).

Totally disagree ☐ ☐ ☐ ☐ ☐ Totally agree

General Comments:

Figure B.3: Third page of the questionnaire (study 2): Questions about the *pen-triggered screen menu*

Bibliography

D.a. Bowman and C.a. Wingrave. Design and evaluation of menu systems for immersive virtual environments. *Proceedings IEEE Virtual Reality 2001*, (0106):149–156, 2001. ISSN 1. doi: 10.1109/VR.2001.913781. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=913781.

Doug A. Bowman, Jean Wineman, Larry F. Hodges, and Don Allison. Designing animal habitats within an immersive VE. *IEEE Computer Graphics and Applications*, 18(5):9–13, 1998. ISSN 02721716. doi: 10.1109/38.708555. URL http://ieeexplore.ieee.org/document/708555/.

J. Callahan, D. Hopkins, M. Weiser, and B. Shneiderman. An empirical comparison of pie vs. linear menus. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '88*, pages 95–100, New York, New York, USA, 1988. ACM Press. ISBN 0201142376. doi: 10.1145/57167.57182. URL http://portal.acm.org/citation.cfm?doid=57167.57182.

Stuart K. Card. User perceptual mechanisms in the search of computer command menus. In *Proceedings of the 1982 conference on Human factors in computing systems - CHI '82*, pages 190–196, New York, New York, USA, 1982. ACM Press. ISBN 0925-5214. doi: 10.1145/800049.801779. URL http://portal.acm.org/citation.cfm?doid=800049.801779.

Thomas Caudell and David Mizell. Augmented reality: an application of heads-up display technology to manual manufacturing processes. In *Proceedings of the*

Twenty-Fifth Hawaii International Conference on System Sciences, pages 659–669 vol.2. IEEE, 1992. ISBN 0-8186-2420-5. doi: 10.1109/HICSS.1992.183317. URL http://ieeexplore.ieee.org/document/183317/.

Nguyen Thong Dang and Daniel Mestre. Effects of menu orientation on pointing behavior in virtual environments. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 6773 LNCS, pages 144–153, 2011. ISBN 9783642220203. doi: 10.1007/978-3-642-22021-0_17. URL http://link.springer.com/10.1007/978-3-642-22021-0{_}17.

Kaushik Das and Christoph W. Borst. An evaluation of menu properties and pointing techniques in a projection-based VR environment. In *3DUI 2010 - IEEE Symposium on 3D User Interfaces 2010, Proceedings*, pages 47–50. IEEE, mar 2010. ISBN 9781424468447. doi: 10.1109/3DUI.2010.5444721. URL http://ieeexplore.ieee.org/document/5444721/.

Steven Feiner, Blair MacIntyre, Marcus Haupt, and Eliot Solomon. Windows on the world. In *Proceedings of the 6th annual ACM symposium on User interface software and technology - UIST '93*, pages 145–155, New York, New York, USA, 1993. ACM Press. ISBN 089791628X. doi: 10.1145/168642.168657. URL http://portal.acm.org/citation.cfm?doid=168642.168657.

Paul M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47(6):381–391, 1954. ISSN 0022-1015. doi: 10.1037/h0055392. URL http://doi.apa.org/getdoi.cfm?doi=10.1037/h0055392.

Sascha Gebhardt, Sebastian Pick, Franziska Leithold, Bernd Hentschel, and Torsten Kuhlen. Extended pie menus for immersive virtual environments. *IEEE Transactions on Visualization and Computer Graphics*, 19(4):644–651, apr 2013. ISSN 10772626. doi: 10.1109/TVCG.2013.31. URL http://ieeexplore.ieee.org/document/6479193/.

Jérôme Grosjean and Sabine Coquillart. Command & Control Cube: a Shortcut Paradigm for Virtual Environments. In Bernd Fröhlich, Joachim Deisinger, and Hans-Jörg Bullinger, editors, *Immersive Projection Technology and Virtual Environments 2001*, pages 1–12, Vienna, 2001. Springer Vienna. ISBN 978-3-7091-6221-7.

Jérôme Grosjean, J. M. Burkhardt, S. Coquillart, and P. Richard. Evaluation of the Command and Control Cube. In *Proceedings - 4th IEEE International Conference on Multimodal Interfaces, ICMI 2002*, pages 473–478. IEEE Comput. Soc, 2002. ISBN 0769518346. doi: 10.1109/ICMI.2002.1167041. URL `http://ieeexplore.ieee.org/document/1167041/`.

Bret Jackson and Daniel F. Keefe. Lift-Off: Using Reference Imagery and Freehand Sketching to Create 3D Models in VR. *IEEE Transactions on Visualization and Computer Graphics*, 22(4):1442–1451, apr 2016. ISSN 10772626. doi: 10.1109/TVCG.2016.2518099. URL `http://ieeexplore.ieee.org/document/7383322/`.

Richard H. Jacoby and Stephen Ellis. Using virtual menus in a virtual environment. *Proceedings of SPIE - The International Society for Optical Engineering*, (May):39–48, jun 1992. doi: 10.1117/12.59654. URL `http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=988025`.

John I. Kiger. The depth/breadth trade-off in the design of menu-driven user interfaces. *International Journal of Man-Machine Studies*, 20(2):201–213, feb 1984. ISSN 00207373. doi: 10.1016/S0020-7373(84)80018-8. URL `https://www.sciencedirect.com/science/article/pii/S0020737384800188`.

Rick Komerska and Colin Ware. A study of haptic linear and pie menus in a 3D fish tank VR environment. In *Proceedings - 12th International Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, HAPTICS*, pages 224–231. IEEE, 2004. ISBN 0769521126. doi: 10.1109/HAPTIC.2004.1287200. URL `http://ieeexplore.ieee.org/document/1287200/`.

Gordon Kurtenbach and William Buxton. The limits of expert performance using hierarchic marking menus. In *Proceedings of the SIGCHI conference on Human factors in* *computing systems - CHI '93*, pages 482–487, New York, New York, USA, 1993. ACM Press. ISBN 0897915755. doi: 10.1145/169059.169426. URL http://portal.acm.org/citation.cfm?doid=169059.169426.

Gordon Kurtenbach and William Buxton. User Learning and Performance with Marking Menu. In *Proceedings of the SIGCHI conference on Human factors in computing systems: celebrating interdependence*, pages 258 – 264, New York, New York, USA, 1994. ACM Press. ISBN 978-0-89791-650-9. doi: 10.1145/191666.191759. URL http://portal.acm.org/citation.cfm?doid=191666.191759.

Gordon P. Kurtenbach, Abigail J. Sellen, and William A.S. Buxton. An Empirical Evaluation of Some Articulatory and Cognitive Aspects of Marking Menus. *Human—Computer Interaction*, 8(1):1–23, mar 1993. ISSN 15327051. doi: 10.1207/s15327051hci0801_1. URL http://www.tandfonline.com/doi/abs/10.1207/s15327051hci0801{_}1.

Gordon Paul Kurtenbach. *The design and evaluation of marking menus*. PhD thesis, 1993. URL https://pdfs.semanticscholar.org/f365/94111ee510c640a67ff9056b91a5749358fe.pdf?{_}ga=2.40392359.403016951.1542095583-1868355239.1542095583.

T. K. Landauer, D. W. Nachbar, T. K. Landauer, and D. W. Nachbar. Selection from alphabetic and numeric menu trees using a touch screen. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '85*, volume 16, pages 73–78, New York, New York, USA, 1985. ACM Press. ISBN 0897911490. doi: 10.1145/317456.317470. URL http://portal.acm.org/citation.cfm?doid=317456.317470.

Hyeongmook Lee, Dongchul Kim, and Woontack Woo. Graphical menus using a mobile phone for wearable AR systems. In *Proceedings - 2011 International Symposium on Ubiquitous Virtual Reality, ISUVR 2011*, pages 55–58. IEEE, jul 2011. ISBN 9780769544205. doi: 10.1109/ISUVR.2011.23. URL http://ieeexplore.ieee.org/document/6068307/.

James E. McDonald, Jim D Stone, and Linda S Liebelt. Searching for Items in Menus: The Effects of Organization and Type of Target. *Proceedings of the Human Factors Society Annual Meeting*, 27(9):834–837, 1983. ISSN 0163-5182. doi: 10.1177/154193128302700919. URL `https://journals.sagepub.com/doi/pdf/10.1177/154193128302700919http://journals.sagepub.com/doi/10.1177/154193128302700919`.

Dwight P. Miller. The Depth/Breadth Tradeoff in Hierarchical Computer Menus. *Proceedings of the Human Factors Society Annual Meeting*, 25(1):296–300, oct 1981. ISSN 0163-5182. doi: 10.1177/107118138102500179. URL `http://journals.sagepub.com/doi/10.1177/107118138102500179`.

Namgyu Kim, G.J. Kim, Chan-Mo Park, Inseok Lee, and S.H. Lim. Multimodal menu presentation and selection in immersive virtual environments. In *Proceedings IEEE Virtual Reality 2000 (Cat. No.00CB37048)*, number FEBRUARY, page 281. IEEE Comput. Soc, 2000. ISBN 0-7695-0478-7. doi: 10.1109/VR.2000.840509.

W. Piekarski and B.H. Thomas. Tinmith-Hand: unified user interface technology for mobile outdoor\naugmented reality and indoor virtual reality. In *Proceedings IEEE Virtual Reality 2002*, volume 2002, pages 2–3. IEEE Comput. Soc, 2002a. ISBN 0-7695-1492-8. doi: 10.1109/VR.2002.996542. URL `http://ieeexplore.ieee.org/document/996542/`.

Wayne Piekarski and B.H. Thomas. Unifying augmented reality and virtual reality user interfaces. *University of South Australia, Wearable Computer Laboratory, School of Computer and Information Science, Mawson Lakes, SA*, 5095:15, 2002b. URL `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.19.4947{&}rep=rep1{&}type=pdf`.

Jeffrey S. Pierce, Andrew S. Forsberg, Matthew J. Conway, Seung Hong, Robert C. Zeleznik, and Mark R. Mine. Image plane interaction techniques in 3D immersive environments. In *Proceedings of the 1997 symposium on Interactive 3D graphics - SI3D '97*, pages 39–

ff., 1997. ISBN 0897918843. doi: 10.1145/253284.253303. URL http://portal.acm.org/citation.cfm?doid=253284.253303.

Katie A. Siek, Yvonne Rogers, and Kay H. Connelly. Fat finger worries: How older and younger users physically interact with PDAs. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3585 LNCS:267–280, 2005. ISSN 03029743. doi: 10.1007/11555261_24.

Ron Van Teylingen, William Ribarsky, and Charles Der Van Mast. Virtual data visualizer. *IEEE Transactions on Visualization and Computer Graphics*, 3(1):65–74, 1997. ISSN 10772626. doi: 10.1109/2945.582350. URL http://ieeexplore.ieee.org/document/582350/.

Daniel Vogel and Patrick Baudisch. Shift: A Technique for Operating Pen-Based Interfaces Using Touch, 2007. URL http://portal.acm.org/citation.cfm?doid=1240624.1240727.

Philipp Wacker, Simon Voelker, Adrian Wagner, and Jan Borchers. Physical Guides. In *Proceedings of the Symposium on Spatial User Interaction - SUI '18*, pages 25–35, New York, New York, USA, 2018. ACM Press. ISBN 9781450357081. doi: 10.1145/3267782.3267788. URL http://dl.acm.org/citation.cfm?doid=3267782.3267788.

Philipp Wacker, Oliver Nowak, Simon Voelker, and Jan Borchers. ARPen: Mid-Air Object Manipulation Techniques for a Bimanual AR System with Pen & Smartphone. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems - CHI '19*, 2019.

Po-Chen Wu, Robert Wang, Kenrick Kin, Christopher Twigg, Shangchen Han, Ming-Hsuan Yang, and Shao-Yi Chien. DodecaPen. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology - UIST '17*, pages 365–374, New York, New York, USA, 2017. ACM Press. ISBN 9781450349819. doi: 10.1145/3126594.3126664. URL http://dl.acm.org/citation.cfm?doid=3126594.3126664.

Index

Typeset February 25, 2019