On-Arm Body Centered 3D User Interfaces

by

Ayaskant Panigrahi

B.Tech., Indian Institute of Information Technology, Design & Manufacturing, Jabalpur, 2020

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

in the
School of Interactive Arts and Technology
Faculty of Communication, Art and Technology

SIMON FRASER UNIVERSITY
Fall 2023

Declaration of Committee

Name:	**Ayaskant Panigrahi**
Degree:	**Master of Science**
Thesis title:	**On-Arm Body Centered 3D User Interfaces**
Committee:	**Chair:** Alireza Karduni Assistant Professor, Interactive Arts and Technology
	Wolfgang Stuerzlinger Supervisor Professor, Interactive Arts and Technology
	Halil Erhan Committee Member Associate Professor, Interactive Arts and Technology
	Chris Shaw Examiner Professor, Interactive Arts and Technology

Ethics Statement

The author, whose name appears on the title page of this work, has obtained, for the research described in this work, either:

a. human research ethics approval from the Simon Fraser University Office of Research Ethics

or

b. advance approval of the animal care protocol from the University Animal Care Committee of Simon Fraser University

or has conducted the research

c. as a co-investigator, collaborator, or research assistant in a research project approved in advance.

A copy of the approval letter has been filed with the Theses Office of the University Library at the time of submission of this thesis or project.

The original application for approval and letter of approval are filed with the relevant offices. Inquiries may be directed to those authorities.

Abstract

Today's 2D user interfaces (UIs) are mature, while their 3D counterparts have not yet reached a similar state. For Virtual Reality (VR) systems, directly using 2D interactive elements as 3D "floating" UIs introduces challenges, such as the lack of haptic feedback when "touching" a virtual mid-air button or when the hand appears behind a panel. I propose VRAMBRACE, an intuitive VR system control method that takes advantage of proprioception and passive haptic feedback to enable interaction with menus, sliders, and keyboards on the user's forearm. VRAMBRACE overlays UI elements on different sides of the non-dominant forearm and enables direct interaction with these elements with the dominant hand's index finger. My evaluation reveals that users perform similarly with VRAMBRACE compared to interacting with the same UI elements floating near the arm for short tasks. As arm registration and tracking were not sufficiently accurate for text entry with VRAMBRACE, users typed faster with a mid-air keyboard. Promisingly, some users still preferred VRAMBRACE due to the added haptic feedback and sensation of touch between the fingertip and the surface of the arm. Findings from my in-depth analysis suggest that VRAMBRACE would benefit from better tracking, calibration, and arm modelling.

Keywords: Virtual Reality; 3D User Interface; Augmented Reality; Human-Computer Interaction

Acknowledgements

I want to express my heartfelt gratitude for the expert guidance of my supervisor, Dr. Stuerzlinger, who supported me at every step. His suggestions, nudges, and advice motivated me to make progress and do my best continually.

I am grateful for the insightful feedback from Dr. Erhan, right from the framing and shaping of the initial concepts to the final version of the prototype.

I am thankful for the support of my friends, especially Aunnoy and Eneas. They were always willing to lend their ears when I needed someone to talk to. I thank all the members of my lab for making my Master's experience a memorable one. I also want to thank Srinadh and Amish, without whom my stay in Vancouver would have felt that much farther from home.

I am indebted to the steadfast support of my family, from their support in me starting my program remotely despite time zone differences to being in constant touch with me throughout my program. Talking with them always brightened my day.

I thank all the members of my thesis committee and the SIAT Graduate program for providing me with the optimal structure, tools, and environment to carry out my research work.

Table of Contents

List of Figures

Chapter 1

Introduction

The interaction paradigms for 2D user interfaces (UIs) have matured over many years [47]. In comparison, interaction in 3D virtual environments has not yet reached the same level of maturity. Core interaction methods like raycasting/virtual hand for selection [1] and teleportation for navigation [65] are increasingly standardized across all Extended Reality (XR) hardware, especially at the consumer level. However, system control interactions involving menus or text/symbol input do not have a common 3D/XR equivalent that works flexibly across all platforms. Simple implementations of "floating" UIs only instantiate traditional 2D UI panels in XR by projecting them onto planar or curved surfaces, placed some distance away from the user. Interaction with such panels is typically through raycasting, which makes it difficult to interact with small and/or dense content. The virtual hand technique, where users can interact directly with a "floating" UI, suffers from a lack of haptic feedback, conflicting visual cues when the hand appears to be behind or penetrate the panel, and also faces similar issues with accuracy when it comes to small and dense layouts. Further, issues arising from a mismatch between disparity and optical focus cues, known as the vergence-accommodation conflict, are present in most VR and AR displays and can adversely affect user performance in 3D selection tasks, particularly those within arm's reach [6].

Body-anchored interfaces can mitigate some of the issues associated with floating interfaces, ensuring that UI elements are always within the user's reach. Several XR systems, such as the Microsoft HoloLens 2 and the Meta Quest 3, offer hand-anchored interfaces for system control. Interaction with hand-anchored interfaces typically relies on hand-tracking through computer vision methods, which are prone to occlusion issues, limiting interactions to relatively simple interaction methods. The limitations of hand-tracking become more apparent when one (or part of one) arm or hand covers the other, effectively discouraging many bimanual gestures, and tracking quality/reliability frequently deteriorates in such cases, too. A generalization to (fore)arm-based interfaces is thus still challenging due to the technical limitations of most XR systems, which cannot track the (fore)arm reliably and accurately enough to be of practical use.

One approach to tracking the arm's pose involves using an inverse kinematics solver, which tries to estimate the pose of the elbow/arm from the pose of the headset and hand controllers [41, 13]. Such approaches tend not to be very reliable, especially when the elbow is moved while the hand and head pose are kept fixed, leading to poor usability and potential frustration in users [28]. Other systems overcome arm tracking challenges through bespoke hardware to detect a touch on the arm, e.g., [22]. The chief limitation of this approach is that it relies on specialized hardware which might not be easily replicable or mass-produced and that it (also) relies on an external arm tracking method. Due to the constraints discussed above, UI elements in XR systems are thus frequently placed around the hand, less frequently on the hand or close to the (fore)arm, and only very rarely on the (fore)arm itself.

In this thesis, I present VRAMBRACE, an intuitive XR system control method that takes advantage of proprioception and passive haptic feedback to enable interaction with menus, sliders, and keyboards on the forearm's surface (as discussed in Chapter 3). Named as a play on the word *vambrace*, a piece of armour that protects the forearm, VRAMBRACE overlays sleeve-like UI elements on different sides of the user's forearm and enables direct interaction on these surfaces with the index finger of the other hand. One of the possible benefits of the approach I propose is that due to proprioception the user could consistently be aware of the location of (at least some of) the UI elements without needing to change their locus of attention. Proprioception is the ability of humans to intuitively know where their body parts are spatially located in relation to each other. Another advantage is the passive haptic feedback afforded from both the fingertip and the arm surface to confirm that an interaction has occurred, which is frequently missing in floating interfaces. Further, VRAMBRACE affords two natural and intuitive mode-switching methods, the first of which operates by rotating the arm, with different arm surfaces revealing different anchored UI elements. The second mode-switching method works through non-dominant hand actions, such as pressing the trigger of the non-dominant hand controller to switch between capitals and small letters on the keyboard. To summarize, my main contributions are:

- Design and development of a novel bimanual XR system control technique, VRAMBRACE, which presents functionally rich and dense UI elements on the forearm.
- A new, intuitive calibration technique to register the forearm for on-body interfaces.
- A novel two-layered mode-switching method through rotation of the forearm and non-dominant hand controller actions.
- A holistic evaluation of system control and text entry tasks, comparing VRAMBRACE to body-anchored floating UI elements.

Chapter 2

Related Work

In this chapter, I present a review of the current research most relevant to my thesis. In addition, I also note that there are other reviews of 3D menu or system control designs, e.g., by Dachselt and Hübner [14] and a technical note focusing on AR by Brudy [11, pp. 1-8].

2.1 Floating 3D User Interfaces

Below, I summarize my review of floating UI elements located in the user's surroundings.

The earliest examples of floating 3D user interfaces appeared in the early 1990's [12, 24, 53, 52]. For other related work on early system control and VR menu systems, please refer to [27].

Most commercial headsets available today (e.g., Google Cardboard and Meta Quest Pro) implement floating menus that can be interacted with by using a ray originating from controllers, bare hands, or the head (i.e., via raycasting). Such menus, while spatially located away from the user and typically out of arms' reach, can still come within the user's reach, e.g., when the user navigates closer to them. When they are outside the field of view, they can be brought back into the user's view either through manual user intervention or when the system detects the menu is outside the user's view and then transitions the menu to use a head/gaze following mode. Some menus also float within the user's reach and can be activated by intersecting them with the controller or by pointing with a finger at the menu items. Instead of relying on raycasting, VRambrace focuses on direct selection with the (virtual representation of the) dominant hand's index finger, also to leverage the effect of the passive haptic feedback provided by the surface of the forearm.

Gebhardt et al. [17] tested extensions to hierarchical pie menus in immersive virtual environments. Their pilot evaluation found that the pick-ray / raycasting selection method performs better than hand rotation and hand projection, possibly due to the strain caused by arm rotations or movements. They also investigated an extended pie menu system with sliders, colour pickers, buttons, and checkboxes. VRambrace supports several UI elements, including Buttons, Sliders, and Color Pickers, and also introduces a virtual keyboard on the arm.

In a comparison with standard (drop-down) menu designs and other input modalities (e.g., voice and hand gestures), Pourmemar and Poullis [46] found that users performed better with head-pointing and nested radial menus in AR. They also reported that users of their system ranked hand gestures as the least preferred, due to increased physical exertion. While their work focuses on nested menus floating in front of the user in AR, I present a rich system of on-arm UI elements that work seamlessly across AR and VR.

2.2 Body-Anchored User Interfaces

In this subsection, I discuss user interfaces that focus on interaction with UI elements that are located relative to the user's body, but not on the body/skin itself.

Early experiments with 3D menus combined raycasting for menus outside of arm's reach and hand-anchored interfaces with options for direct interaction when menus were closer to the user [36].

Today, many commercial solutions support bare-hand tracking through the Leap Motion (now Ultraleap) controller or headsets with such technology built into them. Therefore, XR headsets like the Microsoft Hololens 2 and Meta's Quest 2/3/Pro can display UI elements anchored around the tracked hand(s).

Azai et al. [4] presented the Open Palm Menu, in which UI elements are placed 10 cm away from the non-dominant hand, either in a vertical or a horizontal direction. They found that users performed best when the UI elements were placed in a vertical direction close to the hand, which presents a reasonable baseline when developing new interfaces, such as the one I propose in this thesis.

Monteiro et al. [38] found that while users of their system strongly preferred menus placed on a wall as compared to close to the controllers, they performed well regardless of the UI placement (wall or controller) or UI type (list/panel or radial). Their findings suggest that users prefer to interact with menus that occupy a small area in their field of view (FOV) when they need to perceive the effect or result of their interaction with the menus on the environment. This finding supports the compact on-arm UI design of VRambrace, as it minimizes occlusion of the FOV by near-arm UI elements and instead limits the core interaction area to just the surface of the forearm (where the user only needs to naturally drop the arm to get an unobstructed view).

Through a system control technique named HandPoseMenu [42], Park et al. explored different hand poses or gestures as a mode-switching mechanism. They displayed near-hand UI elements as a vertical list of options corresponding to each gesture. While their study focused on accurately classifying bare-hand gestures for mode-switching, such interactions could be used either with other bare-hand tracking methods (e.g., the Leap Motion controller) or with equivalent controller action mappings. Thus, I used an analogous mode-switching technique with controllers in VRAMBRACE.

Li et al. [29] investigated the quantitative and qualitative aspects of pointing at different locations *in the vicinity of* the arm, at least 4 cm away from the skin. I follow a similar bimanual approach, using the dominant arm's index finger for interactions with UI elements anchored to the non-dominant hand but for UI elements on the arm's surface. VRAMBRACE explores functionally richer interactions with sliders, a touch panel, and a keyboard presented directly on the user's arm, going beyond the simple pointing task investigated by Li et al. [29].

2.3 On Body User Interfaces

On-body interfaces involve UI elements that are spatially co-located with the surface of one or more body parts. As accurate body tracking hardware is still relatively uncommon, differences in tracking implementations and UI display methods afford different kinds of interactions with varying levels of accuracy. Previous body tracking systems have explored the use of computer vision techniques [21], optical markers [29], bio-acoustic signals [22], smartwatches [62], and other bespoke hardware to enable body tracking. The tracked body part is then used for anchoring the UI elements to the palm [61], forearm [3, 29], or even the waist [60, 5, 28]. Finally, the display method can involve projectors mounted above or on the user's body [22, 20], or VR/AR HMDs, e.g., [3, 29].

Early research by Liang and Green [30], and Shaw and Green [51] presented the idea of mapping the rotation of the hand to menu selection or mode-switching via a 1-DoF Ring Menu. Mine et al. [37] presented many interaction techniques that use proprioception and haptic feedback to provide natural ways of interacting with virtual objects. Bowman and Wingrave presented a bimanual menu system called TULIP [10], where different menu items appeared on the fingers of both hands. The non-dominant hand was mapped to top-level menus, while the dominant hand was mapped to specific options in the next menu level. The user selected an option by performing a pinch gesture between the corresponding finger and the thumb. While they found that the TULIP menu occupied minimal area within the user's FOV compared to floating or pen-tablet menus, novice users initially found the indirect interaction slightly confusing. In contrast, all of the interactions in VRAMBRACE are performed directly by the extended index finger on the UI elements displayed on the forearm, thus avoiding this issue altogether. It is also worth noting that with sufficient training, TULIP can be used as an eyes-free system control technique, even though users in the study always kept their hands inside their view throughout. In a related vein, Piekarski and Thomas developed Tinmith-Hand [44, 45], which also focused on mapping finger pinches to options in the menu. They took the idea further by connecting AR and VR applications, adding head-pointing controls, and introducing nudging for fine control in a CAD modelling scenario.

Kohli and Whitton [26] found through their Haptic Hand prototype that presenting UI elements on the non-dominant hand enabled more precise input. Through several example applications for their Armura prototype [21], Harrison et al. demonstrated an effective use of hand and arm gestures coupled with contextual data to provide input. Despite the relatively small input gesture set, this work inspired my exploration of bimanual interaction coupled with hand gestures, such as switching between different menu modes or mapping certain gestures to the Shift key in a keyboard interface.

Lin et al. demonstrated PUB [31], an eyes-free technique that relies on the haptic feedback afforded by the forearm surface to achieve increased accuracy in tapping discrete locations on the forearm. They also showed how taps and sliding gestures on the arm can be mapped to a music player app controls on a phone. They found that users of their system could discriminate up to 8 points along the surface of the forearm without looking at their arm. For VRambrace, I focused on interactions requiring the user to look at their forearm. This distinction is important since VRambrace supports dense UI elements like a virtual keyboard, which contains as many as 30 buttons on a single side of the forearm.

Azai et al. [3] used the arm surface to enable different touch and sliding interactions. In related work [5], they also demonstrated an on-body menu technique called the Tap-Tap menu. The Tap-Tap menu has buttons placed at different locations on the body, including the arms, abdomen, and legs, which are easily accessible in a seated position. The user interacts with the menus using gestures, such as pointing, tapping, and opening/closing the fists. However, these two systems suffered from substantial tracking reliability issues (as visible in the demonstration videos). The authors did not evaluate the developed prototypes with a formal user study. Compared to their work, I further expand the techniques for on-arm interfaces by exploring dense interfaces, such as a full keyboard for text entry, which necessitates accurate finger and arm tracking, and evaluate the interface in a user study.

Reich et al. [48] developed ArmTouch, where they used the area on the forearm near the wrist to present UI elements. The users interacted with the elements through horizontal and vertical swipe gestures. They found that users performed similarly, regardless of whether they used a touchpad or their forearm's skin as the support surface for the interactions. They also found that the users preferred horizontal swipes to vertical ones, as horizontal swipes provided users with a more active area to work with. Instead of just targeting the wrist area, I designed VRambrace to extend the interactive area further to cover the entire length of the forearm while also supporting denser UI elements like a keyboard. I additionally use multiple sides of the forearm to present different UI elements, with rotation around the forearm's length as a mode-switching mechanism.

Lediaeva and LaViola [28] compared combinations of menu placement (spatial, arm, hand, and waist), shapes (linear and radial), and selection techniques (raycasting, head-, and eye-gaze). While their findings indicated that users rated the arm placement for menus to be least preferred, they reported one of the possible reasons for this could be the lack of accurate tracking of the elbow. Motivated also by their results, I moved away from IK solutions [50] for VRambrace, where I rely on a Vive tracker placed near the elbow to mitigate some of the problems caused by poor forearm tracking.

In a recent study, Yu et al. [64] explored the unique design space of combining on-body and mid-air interactions. Their work focused on the various design patterns that arise through the combination of mid-air and on-body elements. They evaluated this idea in an expert evaluation with free-form exploration. My work focuses instead on the possibilities of on-arm interactions, which allows us to investigate the benefits of on-body interfaces in greater detail, especially from a quantitative standpoint.

Reiter et al. [49] investigated combining on-arm UI elements with gaze control. They introduced three interaction techniques centred around a one-handed "Look and Turn" interaction, which combines gaze pointing with arm turning and a pinch gesture. The arm turning gesture to switch between menus is similar to the rotational mode-switching in VRambrace. However, I intentionally designed VRambrace as a bimanual technique that relies on the dominant hand's index finger for pointing and selection. Further, I used the entire forearm surface to present functionally different kinds of UI elements. VRambrace also does not depend on eye tracking, thus freeing the eyes of the users to look at the UI elements or the environment.

Tran et al. [59] explored augmenting the haptic feedback experienced by users in on-body interfaces with vibrotactile feedback. They achieved this by mounting a small motor on a fingernail and relied on *referred phantom sensations* to create the effect of "clicks". Their findings support the benefits of augmenting touch with vibrotactile feedback. While Tran et al. focus on designing expressive tactile feedback patterns through multiple psychophysical experiments, my goal with VRambrace is to explore richer UI controls beyond simple buttons or discrete alerts.

2.4 Text Entry in XR

Text entry in VR has its own set of challenges [23], prompting unique solutions, up to and including portable keyboards [43]. Although many studies on text entry in VR (see [15] for a comprehensive literature review), most of these studies use the QWERTY layout, as most target users are familiar with it. I leverage this for VRambrace, too.

When evaluating PalmType [61] against touchpad-based keyboards, Wang et al. found that users typed 39% faster with their optimized layout that presented the QWERTY keyboard on the palm of the non-dominant hand (7.66 WPM) compared to typing with a touchpad (5.5 WPM). Additionally, even with the unoptimized rectangular QWERTY layout users achieved on average 15% faster speed (6.33 WPM) than with the touchpad baseline. Users of their system also preferred using PalmType to the touchpad-based baseline method, which motivates my exploration of dense on-body UI elements. Similar to PalmType, VRambrace is designed as a bimanual technique, with the dominant hand's index finger pressing on keys laid out on the non-dominant hand. However, instead of using the palm, VRambrace uses the entire length of the forearm to present the keyboard, thus allowing for bigger keys and more gaps between letters. Further, this frees up the palm of the non-dominant hand to perform mode-switching actions, e.g. pressing the controller's trigger to switch the letter case in VRambrace.

Previous work by Grubert et al. [18] showed that in VR a video inlay of the user's hand while typing with a physical keyboard can decrease error rates while reaching a mean speed of 38.7 WPM when typing with both hands. Pham and Stuerzlinger [43] found that users preferred seeing a video passthrough of their hands while typing, and could achieve mean speeds approaching 60 WPM with a physical keyboard and ten-finger typing. This motivates my development of an AR video passthrough mode for VRambrace.

Speicher et al. [57] compared head-pointing, controller-pointing, controller-tapping, freehand, and discrete and continuous cursor control to evaluate selection-based text entry in VR. According to their results, ray-casting-based controller-pointing achieved the fastest typing speed at 15.4 WPM. A study by Lu et al. [32] investigated eye blinks as an alternative to dwell for a head-pointing-based keyboard. They found that blinks were able to achieve a typing speed of 13.47 WPM, compared to 11.65 WPM with dwell. Jimenez and Schulze explored pinch as a selection/activation technique along with head-pointing [25] for typing in VR, but their technique was not formally evaluated by the authors. Yildirim and Osborne [63] compared flat and curved keyboards with controller pointing in a VR study. Their results identified that the flat layout (17.35 WPM) performs better than the curved one (11.81 WPM).

Meier et al. [35] presented TapID, where they leveraged tap location and tap finger detection through a bespoke wrist-worn device to interact with UI elements. While they do not evaluate text entry performance, system control and text entry UI elements could be presented on flat surfaces near the user or on the user's forearm, as shown in their application prototypes. In related work, [58], Streli et al. presented TapType, a text entry technique that enables users to touch type by tapping patterns onto everyday surfaces. As shown in their accompanying video, the user could even use the thighs as a flat surface for touch typing while seated. Further, they developed a Mixed Reality prototype for TapType on the Meta Quest 2 to demonstrate the benefit of added passive haptic feedback while typing. Users of the TapType system achieved a mean speed of 19.2 WPM in the third evaluation block while tapping the fingers of both of their hands on a table. However, the evaluations in both of these works focused only on the accuracy of their machine learning-based systems but did not evaluate the user performance of the prototypes with on-body interfaces. My work closes this gap and focuses on a direct comparative study between on-arm and mid-air interactions.

One recently introduced solution by He et al. [23] involves a multimodal text entry system, which the authors named TapGazer. In TapGazer, users type by tapping their fingers either on a touchpad or anywhere else within the reach of the hand-tracking sensors through a body-word wearable device, e.g., on their thighs. To support a QWERTY layout, the 26 letters are mapped to at least one of the eight non-thumb fingers, e.g., "qaz" is mapped to the pinky of the left hand, "wsx" to the ring finger, and so on. The two thumbs are used for different editing functions, such as word selection and deletion. Once the user starts typing, the interface suggests a list of potential candidate words. Gaze is then used to resolve the ambiguity between these words. However, ambiguity can also be resolved with additional taps when gaze tracking is unavailable. In a user study where participants typed both seated and standing in VR, results showed that seated participants who typed on a touchpad could type at 44.81 WPM, i.e., 79.17% of their usual speed on a physical QWERTY keyboard when using both hands. Standing participants, who typed on their thighs using a wearable device, typed at 45.26 WPM, i.e., 71.91% of the typing speed with a physical keyboard.

Song et al. [54] found that users of their hand-gesture-based keyboard system could perform faster mode switches compared to pressing explicit mode-switching buttons. Like their system, I used finger and hand rotation gestures to switch between modes in VRambrace. However, as VRambrace in its current iteration is a controller-based technique instead of relying on bare-hand interaction, I mapped the Shift/letter-capitalization action to the controller's trigger button action, which is activated by the index finger. Further, while the system developed by Song et al. investigated only the performance of mid-air keyboards, I also study the effect of placing a keyboard on the user's forearm on text entry performance.

Chapter 3

VRambrace System Description

3.1 Rationale and Design Iterations

I present VRAMBRACE as a bimanual, intuitive system-control technique that leverages hand and forearm tracking. The technique uses the non-dominant forearm as a frame of reference for the body-centred positioning of the UI. The interaction with the UI elements is supported by *proprioception* and the forearm surface for passive haptic feedback. The dominant hand's index finger is then used to interact with the different UI elements. Thus, the user experiences haptic feedback from two sources – the fingertip and the point on the forearm with which it is in contact.

My first controller-free prototype for the technique used a Leap Motion hand-tracking unit attached to the front of the VR headset, similar to previous work [7, 8] and recent commercial headsets, such as the Varjo XR-3. While the tracking of the fingers and hand joints was stable, I soon ran into issues with unreliable arm and wrist tracking. The tracking reliability of the Leap Motion hand-tracking method was reduced particularly when the dominant hand occluded parts of the non-dominant hand, an issue that occurred frequently, as VRAMBRACE involves bimanual interaction. For VRAMBRACE, both good bimanual and forearm tracking are crucial for making the user believe that the UI is anchored to their arm and follows its movements. Yet, most computer vision-based arm-tracking methods with HMD-mounted cameras require the user's arm to be in front of the headset, e.g., [7]. Only the most recent HMD models, such as the Meta Quest 2 or 3 and the Apple Vision Pro, present exceptions, as they now feature cameras that also point downwards, similar to previous research [40], affording much more ergonomic hand and arm positions.

I tried to improve this by integrating Leap Motion hand tracking data with the elbow position computed through a popular IK solution for Unity [50]. The IK solution provided me with an estimate of the elbow position based on the positions of the controllers and the HMD. Yet, this still did not yield consistent results across different arm poses. Changes in the rotational pose of the controllers occasionally caused the reported elbow position to shift away from the real position of the elbow. Further, while the IK solution was flexible

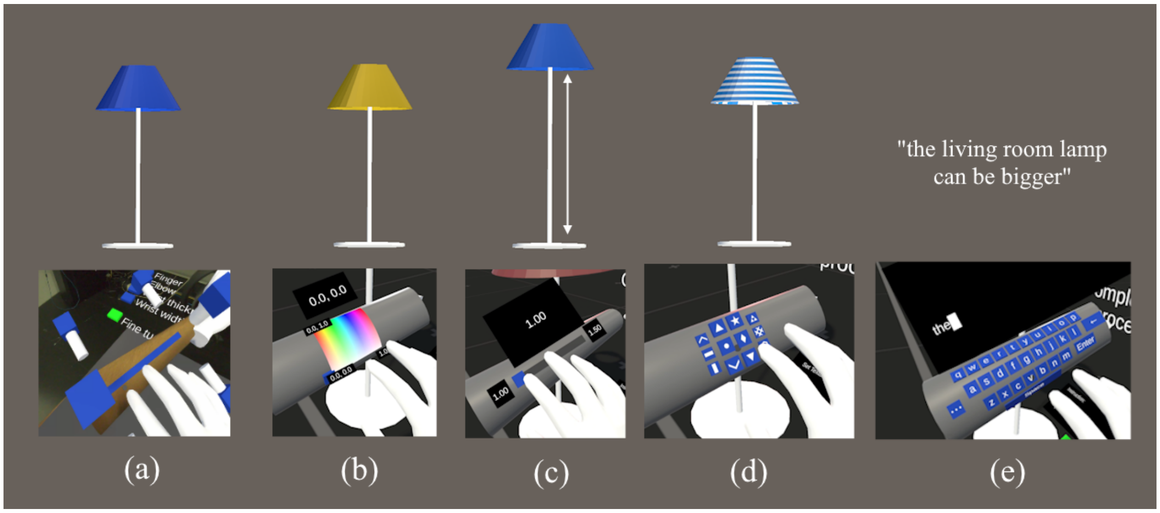


Figure 3.1: (a) Arm being calibrated in AR mode, and (b – e) VR views of different UI controls to change colour, height, texture, and annotation of a virtual floor lamp.

enough to be calibrated to various arm lengths and avoided some unusual arm poses, several aspects of the calibration process buried in the plugin options did not generalize well enough to be useful outside this specific plugin/apparatus setup.

As VRambrace relies on forearm rotation to switch between different UI modes, I needed a robust estimation of the elbow position for various arm poses involving rotations around the length of the forearm. Further, I needed a repeatable and intuitive way of calibrating multiple offsets to account for the varying arm sizes of different users, which proved to be especially important in the AR mode when the users can see their real forearm as the backdrop for the UI elements.

I developed the final VRambrace prototype by using two Valve Index/Knuckle controllers to track the hands, combined with an HTC Vive tracker worn just above the elbow on the non-dominant hand to get accurate and stable elbow tracking. In the absence of full finger tracking, I represent the user's hands in my prototype with two virtual hands that are always in an open-handed pose (all fingers extended in a relaxed manner). I track the tip of the dominant hand's index finger by asking users to keep that finger straight and calibrating that finger's length at the start with a technique previously presented by Wagner et al. [16]. The Index/Knuckle controllers that I used have wristbands that permit users to release their grip on the controller, thus allowing more freedom in hand poses. Positioning the tracked devices near the wrist and elbow also enables intuitive calibration that captures the corresponding offsets to the nearest tracked device and keeps tracking approximations to a minimum. The resulting (relatively) robust and stable tracking then enables users to interact even with dense UI elements like virtual keyboards.

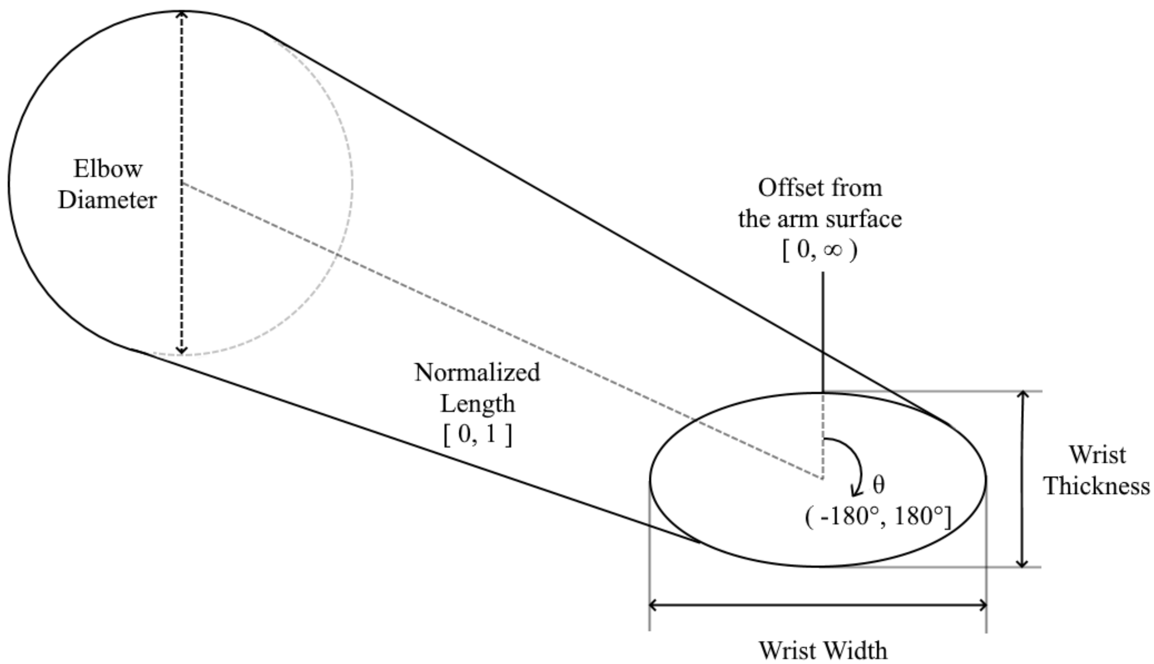


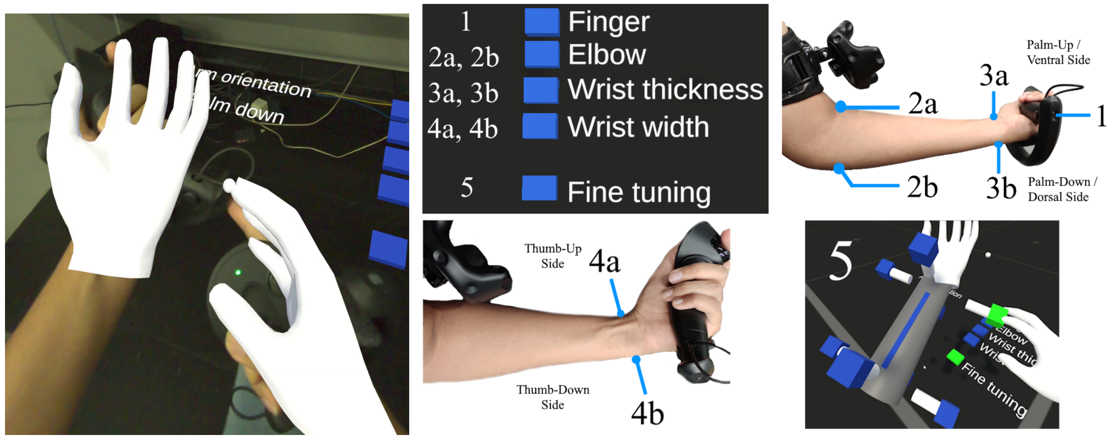
Figure 3.2: The arm model for VRAMBRACE is similar to that used in Bergstrom et al. [9]. Each point above the surface is represented by Normalized length, Rotation angle and Offset from the arm surface.

3.2 Simplified Arm Model

I initially experimented with approximating the arm as a rectangular prism, analogous to the conceptual model of presenting different UI on four sides of the arm. I quickly ran into problems as offsets that worked for one side of the arm would not work for the other sides, thus resulting in a poor approximation of the arm. Consequently, I approximate the human arm as a conical volume, similar to Bergstrom-Lehtovirta et al.'s work [9]. More specifically, I use a circular cross-section near the elbow and an elliptical near the wrist, with the conical volume in between constructed by interpolating between these two cross-sections, see Figure 3.2. In this model, each point is represented by three values, resulting in a modified cylindrical coordinate system:

1. **Normalized Length** – Ranging from 0 near the elbow to 1 at the wrist.
2. **Rotation Angle** θ – Ranging from $-180°$ to $180°$, with $0°$ representing the palm-up / ventral side of the forearm.
3. **Offset from the arm surface** – Points above the surface have positive values, with 0 representing points on the surface.

I iteratively developed a custom calibration process to obtain the radii for the ellipses near the elbow and wrist; see the next subsection. Even though I used a curved surface for



(a) (b)

Figure 3.3: (a) Index Finger Calibration (Step 1) while in AR mode, and (b) the remainder of the 5-Step Calibration Process with the calibration points illustrated in the photos and the Fine Tuning offset adjustment (Step 5) shown in VR mode.

the forearm, I conceptually still consider it to have four faces or sides based on the rotation around the length of the forearm, i.e., the wrist rotation. These four states are: palm facing upwards (ventral forearm), thumb pointing up, palm facing down (dorsal forearm), and thumb pointing down (which is harder to reach, so unused). Based on area constraints, I assign different UI elements to different sides of the arm. During design iterations, I found that the thumb-up side usually offers limited vertical space, so I placed thinner UI elements on this face, e.g., a slider, while I reserved the bigger elements (e.g., the keyboard) to the palm-up and palm-down sides.

3.3 Calibration Process

I capture all the dimensions required to calculate the dominant hand's index finger length and to construct the non-dominant arm's digital arm model through a 5-step calibration process (see Figure 3.3b). Each step is initiated by selecting the corresponding virtual button from a calibration panel listing all five steps in order. The calibration steps involve one or more presses of the 'A' button on the dominant-hand controller while the dominant hand's extended index finger points at a predetermined set of points on various features of the non-dominant forearm to specify different offsets. During this calibration, I encouraged users to keep their non-dominant hand as still as possible with the palm facing upwards. To verify the accuracy of the registration, i.e., the virtual arm model generated from calibration steps, the user could also toggle between passthrough AR and opaque VR at any time by pressing on the thumbstick.

The five steps of the calibration process are detailed below:

Index Finger: As users use their dominant hand's index finger to interact with all the UI elements, the first step of the calibration process involves obtaining the offset in pose between the extended index fingertip and the controller. For this, the user touches a fixed point on the bottom of the controller (the USB port of the Index/Knuckle controller) held in the non-dominant hand with their extended index finger and presses the 'A' button (Figure 3.3a). I assume that once the straps of the controllers are tightened, this relative offset between the index finger tip and the controller is fixed, and the user constantly maintains a hand posture with their index finger extended. This also implies that each time the user adjusts their hold or re-grips the controller they would need to perform this calibration step to get the best experience.

Elbow Diameter: I record two selections on the elbow by asking the user to press the 'A' button on the controller while the extended fingertip is touching the inner/ventral side (Figure 3.3b, Step 2a) and the outer/dorsal side (Figure 3.3b, Step 2b) of the elbow diametrically opposite the first point. These two points allow us to calculate the offsets from the Vive tracker worn above the elbow and the radius of the circular elbow end of the arm model. I also record the base rotation for the palm-up mode in this step.

Wrist Thickness: Similarly, the user selects two more points, one in the middle of the ventral/palm-up side (Figure 3.3b, Step 3a) and another on the dorsal/palm-down side (Figure 3.3b, Step 3b) of the wrist, diametrically opposite the first point. Using these points, I can compute the smaller radius of the ellipse at the wrist and its offset for the arm model.

Wrist Width: Then the user is asked to use their extended fingertip to touch the side of the wrist close to the thumb while holding their palm upwards (Figure 3.3b, Step 4a), followed by touching the diametrically opposite side of the wrist (Figure 3.3b, Step 4b), which records another two data points. These two points allow us to calculate the wrist offset from the controller and the larger ellipse radius at the arm model's wrist end.

Fine Tuning: In this mode, eight virtual knobs or handles appear, four of them close to the wrist while the other four are close to the elbow (Figure 3.3b, bottom-right). Together, these knobs allow the user to fine-adjust all the offsets and radii captured in the previous steps. The user can pull or push on these knobs while holding onto the 'A' button on the controller. The user can subjectively test if the calibration closely approximates their real arm by touching the blue strips that appear on each face of the arm. The user can also switch to the AR mode to align the transparent arm cutout/passthrough video as closely to their real arm as possible using these virtual knobs.

3.4 Two Layered Mode-Switching

VRambrace offers two mode-switching methods, and the user can use either option in parallel. First, the user can rotate their non-dominant forearm [51, 3, 29], which in VRambrace is the primary mode-switching mechanism to switch between different UI elements.

Secondly, the user can perform non-dominant hand controller actions analogous to performing bare-hand gestures [26, 42, 54], which in VRambrace is used to switch between modes specific to a UI element, e.g., switching the letter case while working with the keyboard. Mapping controller trigger actions to an index finger pointing gesture is also common across plugins and SDKs in commercial headsets (including Meta Quest 2 and SteamVR-based devices), which further supports my choice of controller trigger actions for mode-switching with the non-dominant hand.

3.5 VRambrace Interactions and Controls

With VRambrace, rotating to the three different faces of the forearm serves as a mode-switching mechanism, revealing three sets of different controls or UI elements, Figure 3.1. I designed the compact on-arm UI design of VRambrace to minimize occlusion of the user's view of the virtual environment by showing only near-arm UI elements. Specifically, I limited the core interaction area to just the surface of the forearm, which the user can naturally drop to have an unobstructed view of the environment. Further, if the UI elements that are not currently being used are not visible, this helps lessen the users' cognitive load [27, pp. 382]. As a cursor, I show a small circular disc on the arm closest to the location of the index fingertip as it approaches the surface of the forearm. This cursor is hidden while interacting with UI controls, as each control has its own visual feedback.

Informed by horizontal and vertical space constraints that I identified during initial design iterations, I typically present a 2D Slider and Touch Panel on the palm-down/dorsal side, a 1D Slider on the thumb-up side, and a QWERTY keyboard on the palm-up/ventral side of the forearm (see Figure 3.1). One of the reasons for this is that I found the palm-up surface to be (generally) less curved than the other faces, which worked well with dense UI elements like the keyboard. While these are not the only possible assignments, I believe (based on my experience) that these are among the best design choices.

The next few paragraphs describe the four individual UI control elements.

3.5.1 2-Axis / 2D-Slider / Color Picker

The 2D slider allows the user to continuously adjust any numerical 2D data by sliding over a 7 cm $\times$ 7 cm slightly curved "plane", with a 1.5 cm $\times$ 1.5 cm knob indicating the current state of the 2D selection. The surface of the UI is curved to match the curvature of the forearm at the location of the UI. To make this into a colour picker, I mapped the X and Y axes from the 2D slider to the hue and saturation of the colour (with the brightness "value" set to a constant 50% in the HSV colour system); see Figure 3.1.

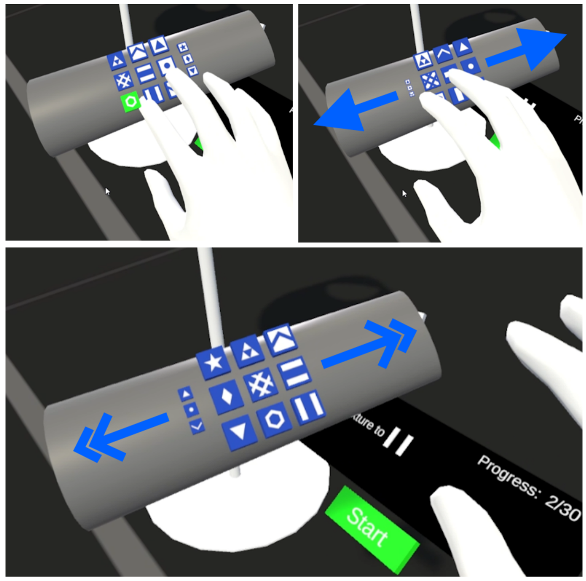


Figure 3.4: Touch Panel Gestures (a) Tapping, (b) Swiping, and (c) Flicking after touch release.

3.5.2 Touch Panel

Taking the idea of a 2D panel further, the touch panel contains an infinitely scrolling list of tiles over a 6 cm × 8 cm area. The 1.5 cm × 1.5 cm tiles are arranged in a grid (with three rows × four columns). The user can interact with this panel using classic touch gestures like tapping to select, swiping horizontally to scroll left and right, and flicking to scroll through the items even after releasing the touch (Figure 3.4).

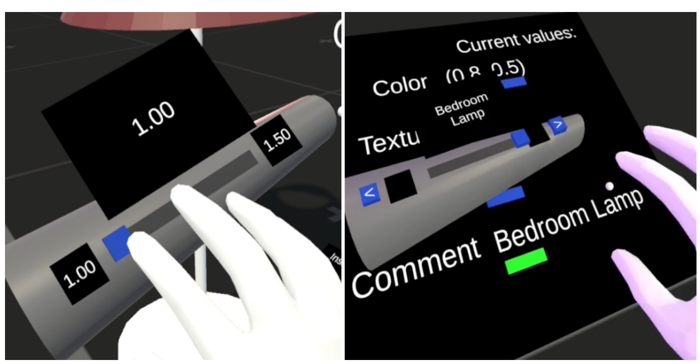


Figure 3.5: 1D-Slider in (a) continuous and (b) discrete modes. In the discrete mode, the knob snaps to the nearest option from a customizable list of 'stops' along the length of the slider, in this example, "Bedroom Lamp."

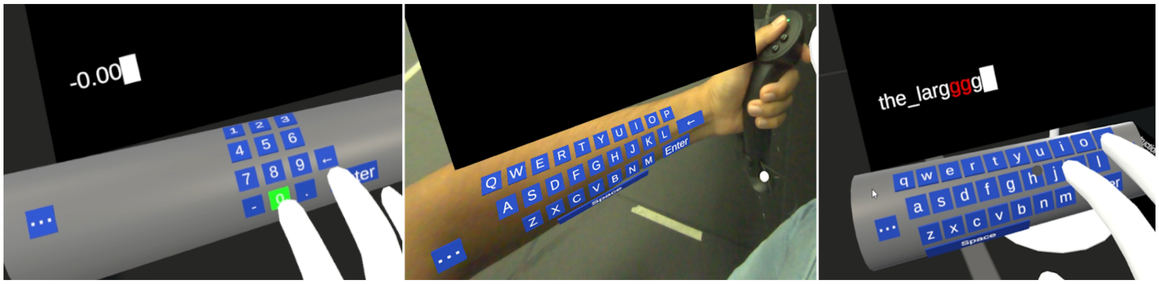


Figure 3.6: From left to right: (a) Numpad, (b) AR view of capital letters, and (c) error highlighting in the text entry box.

3.5.3 1-Axis / 1D-Slider

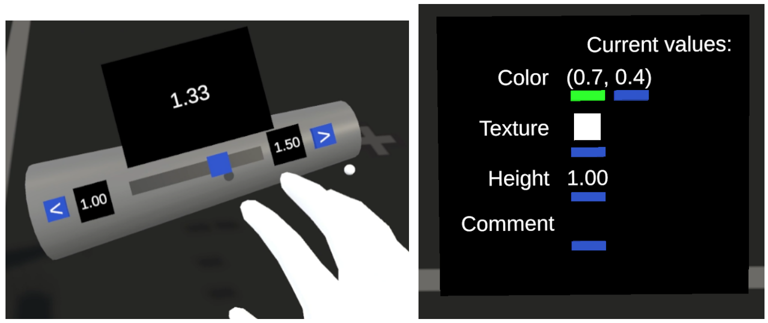
Shown on the thumb-up side of the forearm, the 1D slider can control continuous or discrete values (Figure 3.5). The control's long surface (10 cm × 1.5 cm) allows relatively precise continuous number input. The slider can also select discrete numerical steps or options from a list using snap points and labels on the slider, analogous to a dropdown box. Similar to the other UI elements, a 1.5 cm × 1.5 cm knob indicates the current state of the interaction.

3.5.4 Alpha-numeric keyboard

Present on the palm-up face, the QWERTY keyboard contains buttons for all letters, Backspace, Enter, and the Spacebar, with an additional button to switch to a Numpad mode. The Numpad mode serves as an alternative method to enter highly precise values quickly in numeric fields when sliders do not work well (e.g., architectural dimensions that are known beforehand and which must be set exactly to 4 or 5 digits of precision, Figure 3.6). The regular buttons are 1.2 cm × 1.2 cm in size, with the special buttons like Enter and Backspace slightly larger at 3 cm × 1.2 cm. The Spacebar, which is 10 cm long, occupies the entire bottom row. The keys are separated 2 mm horizontally and 30° angularly between rows in the other direction so that the keyboard conforms to the shape of the curved virtual arm, which (roughly) yields the same 2 mm gap on average in that direction. The entered text is displayed in a panel just above the keyboard, with any spaces visualized as underscore characters ('_') for clarity.

The user can also use controller actions in the form of pressing (and holding) the trigger on the controller held in the non-dominant hand to add a layer of mode-switching analogous to HandPoseMenu [42], which serves as a Shift key mechanism to toggle between small and capital letters (Figure 3.6).

Finally, the text entry field also supports error highlighting, which can be integrated with a spellchecker (Figure 3.6).



(a) (b)

Figure 3.7: (a) Previous and Next Buttons for Focus Traversal shown on the two ends of a 1D-Slider, and (b) Properties Panel with buttons to indicate the property being currently set (active one in green, other buttons in dark blue).

3.6 UI Focus Traversal

A traditional UI dialogue uses focus traversal to navigate between input elements/fields. In VRambrace, and as one option for a focus traversal mechanism, the user can tap on the "previous" or "next" buttons shown on each face of the forearm (Figure 3.7a) to cycle through the parameters they are changing, e.g., switching between changing the height of a lamp (1D numeric data) to adding an annotation to the lamp (text entry). For easy access, these buttons are present on the left and right extremes on each face of the forearm. Alternatively, the user can select virtual buttons positioned below each property field in a static, floating UI dialog present in the environment to change the focus. The user can select these buttons by bringing the index finger of their dominant hand close to the desired button (i.e., hovering) and then confirming the action by pressing the 'A' button on the controller held in the dominant hand, similar to the buttons for the calibration steps. However, using this option to select the active UI element requires them to traverse a proportionally longer distance, as the UI dialog is further away from the forearm than the two on-arm focus buttons (Figure 3.7b).

3.7 Smart contextual toggling of UI control elements

Further, I use smart contextual switching between different UI elements by showing only relevant UI controls for each specific data type, e.g., for changing a 2D numerical value like colour, I show the 1D-Slider, the 2D-Slider, and the Keyboard in Numpad mode (editing a single number at a time as required with the slider and the Numpad). Another example of this occurs when editing a text field: in the accompanying figure, I show the 1D-Slider in the discrete mode populated with configurable text options (thus acting like a dropdown, see Figure 3.5b) and the Keyboard in the Alphabet mode. This helps the user to access the relevant UI element faster and removes some visual clutter / cognitive load.

3.8 Right- and left-handed user support

VRambrace supports left- and right-handed users, which the user can change simply by customizing the mapping of dominant and non-dominant hands in my prototype system. The application handles all required changes to make this work, e.g., by correcting the orientation of all UI elements by flipping them. To make the rotational mode-switching similar across both left- and right-handed users, I also reverse the rotational direction along the forearm, i.e., flipping anti-clockwise and clockwise rotations to access different UI elements.

3.9 Baseline: Body-anchored floating mid-air UI

As a baseline, I developed a body-anchored floating Mid-Air UI version for each UI element in VRambrace: 1D-Slider, 2D-Slider, Touch Panel, and Keyboard (top row in Figure 4.2). The Mid-Air UI elements worked similarly to VRambrace, i.e., the user interacts directly with them using the index finger of their dominant hand, just in mid-air. Similar to the work by Azai et al. [4], I placed these UI elements at an offset of 10 cm from the forearm, on the side of the arm that is away from the body. I made this decision as I noticed in my initial design iterations that placing the UI elements closer to the body (i.e., between the forearm and the waist) would force the user to place their arm much further away from their body in an uncomfortable position. In other words, if the floating UI elements were placed closer to the body, the user would have to bend forward and strain their neck to look downwards at these elements. All four Mid-Air UI elements were completely flat (unlike the On-Arm condition, where the 2D-Slider and the Keyboard were curved to conform to the shape of the user's forearm).

Chapter 4

User Study

I conducted the user study in VR, where I mapped the controls mentioned above to different attributes of a virtual floor lamp, including its height and colour. This served as an application scenario demonstrating the usage of all four types of controls while also doubling as tasks in my formal usability evaluation. More specifically, I used the colour picker version of the 2D-Slider to set the colour of the lampshade and, with it, the light emitted, and I mapped different texture patterns on the lampshade to the tiles of the Touch Panel. The 1D-Slider changed the height of the lamp in the user study, and the Keyboard was used to add an annotation to the lamp.

4.1 Research Hypotheses

My hypotheses for this user study were:

- **H1:** Users perform simple system control tasks faster and with fewer errors using VRambrace vs. Mid-Air.
  - Users perform faster with VRambrace.
  - Users make fewer errors with VRambrace.
- **H2:** Users perform text entry tasks faster and with fewer errors using VRambrace vs. Mid-Air.
  - Users perform faster with VRambrace.
  - Users make fewer errors with VRambrace.
- **H3:** Users subjectively prefer VRambrace to Mid-Air interaction.
  - Users find VRambrace easier to use.
  - Users find VRambrace easier to learn.
  - Users find VRambrace less tiring to use.

– Users rank VRambrace higher in terms of overall preference.

The design of VRambrace is motivated by the known benefits of on-body interfaces due to the tactile feedback on the finger from touching the arm, the sensation of touch on the forearm, and proprioception [37, 22, 59]. I thus expected users to appreciate the added passive haptic feedback and that the interaction technique makes it easy to access the UI elements quickly, as the user always knows where their forearm is and, thus, where the UI elements are. Previous work exploring on-arm interfaces has generally received good reviews from users [48, 59].

I conducted my study to verify the above-mentioned hypotheses and to understand the effect of the different UI Placement conditions [28] on user performance during system control tasks. In Phase 1 of my study, I focused on simple and short system control tasks like setting height, colour, and texture through UI elements corresponding to H1. Phase 2 of my study investigated text entry tasks. For each task, the UI elements were placed on the arm (On-Arm with VRambrace) or in Mid-Air, i.e., at an offset 10 cm away from the arm. I collected both objective performance data logged automatically and subjective measures through questionnaires.

4.2 Study Design and Tasks

To understand the effect of UI Placement, i.e., to compare the On-Arm and Mid-Air conditions, I conducted a within-subjects study in VR (Figure 4.1, Figure 4.2). This study was divided into two phases: the 1D and 2D sliders and the touch panel were used in the first phase to complete short parameter setting tasks, and the keyboard was used in the second phase to complete a text entry task. I separated out the text entry task into its own phase as, compared to the other tasks, I found in my pilots that this task took substantially longer and was more complex due to the larger number of atomic interactions, i.e., virtual key presses. Another difference between the phases was that participants did not have to switch between sides of the forearm in Phase 2, as the keyboard always appeared on the palm-up side. While the user could switch to the AR mode during the calibration steps, the user performed the actual tasks for the study in the VR mode. I did this to ensure a consistent environment for all users, removing possible confounding factors related to distracting elements in the background. The reason behind this decision was that during my initial design iterations I noticed that the static virtual hand pose (where the fingers were always extended in a relaxed manner) were particularly disconcerting and distracting in the AR mode. Still, to get feedback about the AR mode, I still asked users to try out VRambrace in AR mode at the end of the study before completing the Post-Study Questionnaire.

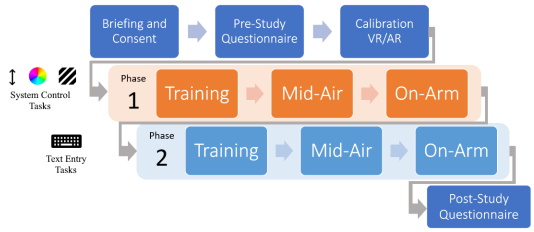


Figure 4.1: Study Design and Procedure.

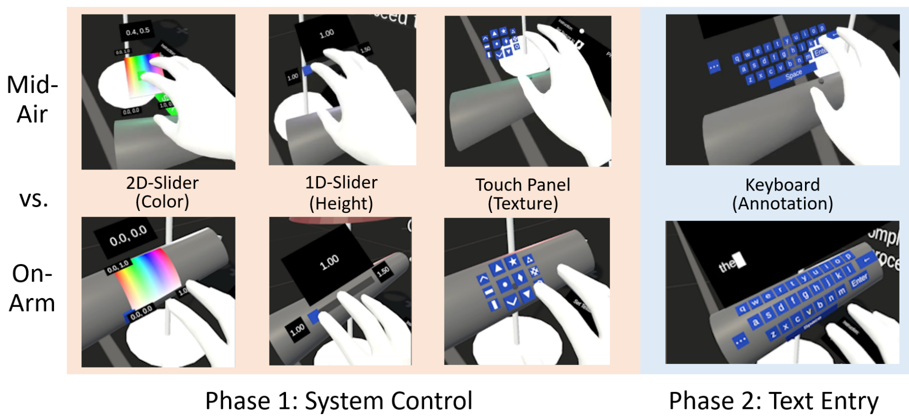


Figure 4.2: I compared Mid-Air and On-Arm UI elements in my study.

4.3 Phase 1: Sliders and Touch Panel

The first phase involved using the 1D-Slider, 2D-Slider, and Touch Panel to change the properties of a virtual floor lamp (see Figure 3.1). Each trial in Phase 1 consisted of three subtasks:

- **Height** – Change the height of the floor lamp to match a randomly set target value between 1 m and 1.5 m using the 1D-Slider that appeared on the thumb-up side of the forearm. The system considered this subtask complete when the participant set a value within 0.01 m of the target.

- **Texture** – Change the texture of the lampshade by choosing the image to match the target, which was one of any 24 pre-generated images that appeared on the palm-down side as the tiles of an infinitely scrolling Touch Panel.

- **Color** – Change the hue and saturation of the lamp's light colour, with the target being a set of two random values between 0 and 1, corresponding to colour hue and saturation. This subtask was considered complete when the participant set each of the two values within 0.1 of the target values using the 2D slider that appeared on the palm-down side.

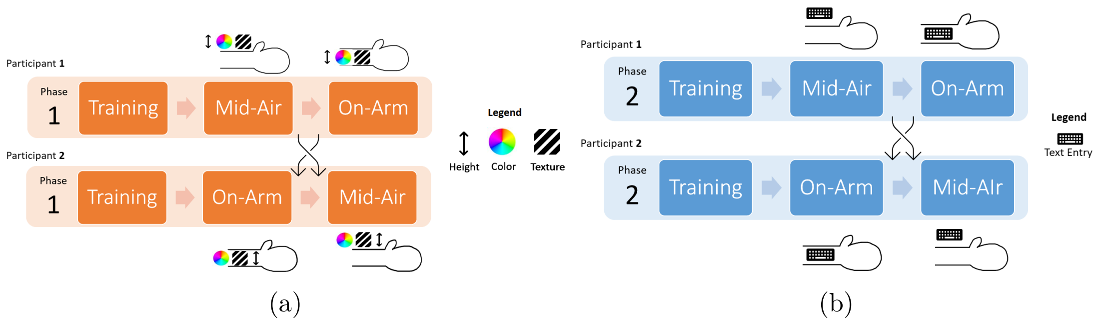


Figure 4.3: Counterbalanced order of tasks in (a) Phase 1 and (b) Phase 2.

The target value for each subtask was shown in a panel in front of the participant, followed by a "Start" button. The order of these subtasks was fixed within Phase 1 for each participant to assist in learning the rotation/mode associated with each UI element but was varied/counterbalanced between participants. Participants completed the set of 3 subtasks 5 times for each of the two UI Placement. Participants performed five repetitions × three subtasks × two UI Placement = 30 trials.

Both the order of tasks and UI Placement were counterbalanced between participants by following a Latin Square design with 6 items (2 conditions × three subtasks) to prevent ordering effects (Figure 4.3a). I disabled the contextual availability of multiple UI elements for a given subtask to avoid a confounding factor related to the choice of UI element used to complete the task. In other words, exactly one UI element was shown for each subtask on the corresponding face of the forearm.

4.4 Phase 2: QWERTY Keyboard

The second phase of the user study involved typing a target phrase using the virtual QWERTY keyboard, which appeared on the palm-up side of the forearm. The target phrase was randomly selected from a standard phrase set [33]. Similar to phase 1, participants had to transcribe five phrases for both UI Placement conditions, i.e., five phrases × two conditions, for 10 text entry tasks. The order of the conditions was again counterbalanced across participants (Figure 4.3b).

4.5 Participants

I recruited 12 participants for the study (7 female), all right-handed except one being left-handed. Five participants were between 18 and 24, while the other seven were between 25 and 34 years old. Half the participants were familiar with VR and AR technology, having used it at least once per month. Six participants had significant experience working with 3D / CAD software (at least a few times a month). Eight participants played computer

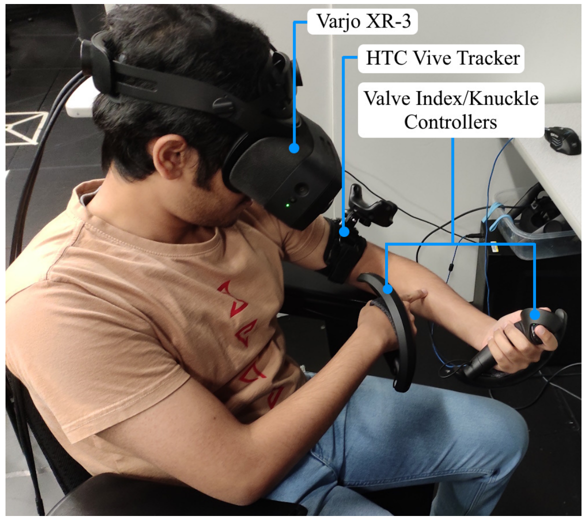


Figure 4.4: Participant using VRAMBRACE.

games at least a few times each month. Each experimental session took approximately 75 minutes, and participants were compensated with $15 or equivalent course credits.

4.6 Apparatus / Equipment

I used a VR-capable high-end PC with an RTX 3080 Ti graphics card for the experiment, driving a Varjo XR-3 AR HMD (see Figure 4.4), which weighs 980 g. I built the experimental software with Unity 2020.3 using the OpenXR pipeline. As mentioned previously, I used two Valve Index/Knuckle controllers (one in each hand) and an HTC Vive tracker on the upper arm near the elbow of the non-dominant hand. To overcome the lack of full-finger tracking, I asked the participants to maintain a constant hand pose, i.e., to keep the index finger stretched out. I used outside-in tracking through 4 Valve Base Stations 2.0 placed at the corners of the tracking area. As mentioned previously, I did not use the hand-tracking provided by the Varjo XR-3, as it did not enable us to track the user's elbow (and thus the forearm), and the bimanual handtracking suffered from occlusion issues.

4.7 Procedure

Each participant started the experiment with a short briefing session and signing the consent form. Afterward, they answered a pre-study questionnaire about their demographic information, handedness, and experience with VR and/or 3D CAD software. I then set the handedness preference in my software to ensure the correct mapping for the participant's dominant and non-dominant hands. This was followed by a detailed explanation of the experiment, including the calibration steps and the tasks. During the experiment, participants were asked to remain seated, resting their non-dominant arm comfortably on the armrest. Participants were also asked to either roll up the sleeves or remove any loose-fitting sweaters as per their preference. Participants with tight-fitting, long-sleeved clothing continued throughout the experiment with the sleeves down, including the calibration phase. I did not make it mandatory for the participants to wear short-sleeved clothing. This was to ensure the conditions during the study remained similar to real-life usage scenarios.

At this point, I assisted the participants in performing a thorough calibration. Each individual's arm model values and offsets were saved and served as the base registration for the remainder of the study for that person. Still, I asked participants to perform the fingertip calibration each time they took off and put on the controllers to prevent potential errors arising from slight changes in grip and hand positions. They were asked to maintain a constant pose of their dominant hand, with their index finger extended throughout their interactions. While the base registration worked most of the time without further adjustments, the participants were also asked to verify the registration each time they took off and put on the headset.

After this, I switched the system to VR mode, and participants proceeded to a training phase. Here, they were allowed to try out the different input elements until they were comfortable with the setup, up to 3 trials for each condition. These training tasks also served as a verification step that the calibration had worked. Then, the participants started the first phase of the experiment, i.e., using the Sliders and Touch Panel to match randomly selected target values for the lamp properties as fast as possible. Participants automatically progressed to the next subtask when they chose the correct value (for the texture) or if their selected value was within 0.1 (for the 2D-Slider) or 0.01 (for the 1D-Slider) of the target value. After each of the two conditions, participants completed a NASA TLX questionnaire. They also answered questions about ease of use, learning, and perceived calibration accuracy (this last item only for the On-Arm condition) on a 5-point Likert scale. Additionally, they could enter comments about their experience in a free-form text box.

The participants then proceeded to the second phase, i.e., the text entry task. After that, and similar to the first phase, they filled out questions related to ease of use, ease of learning, and the NASA TLX questionnaire at the end. After the participant completed both text entry tasks with the two conditions and the corresponding questionnaires, they were invited to freely explore the On-Arm UI elements in AR mode and give feedback on the whole VRambrace interface. Following this, they completed a short post-study questionnaire, where they ranked and justified their overall preference for UI Placement. They also answered questions related to the calibration and were asked to compare standard floating UIs to body-anchored, on-arm interfaces using free-form text boxes.

Chapter 5

Results

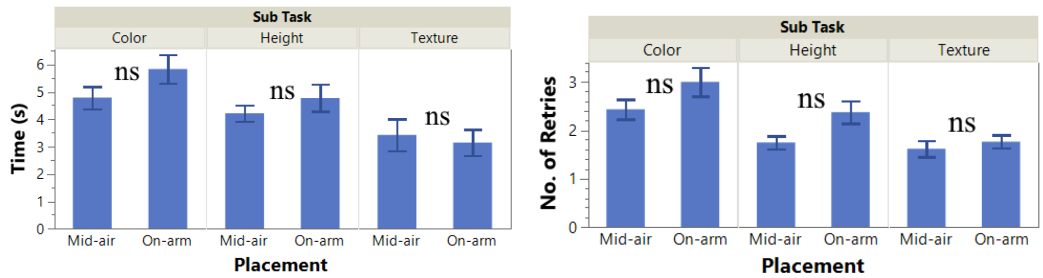
I used JMP 16 to perform the statistical analysis for the repeated measures / within-subject experimental design, except for the MSD Error Rate analysis, where I used SPSS 29 to run a non-parametric test. To ensure the data was normally distributed before each parametric test, I verified that skewness and kurtosis were within reasonable bounds ($-1.5 < Skewness, Kurtosis < 1.5$) [34, 19]. As specified in the corresponding paragraphs, when the data was not normally distributed I performed log transformations for some measures and excluded extreme outliers before analysis.

5.1 Phase 1: Sliders and Touch Panel

I compared the effect of UI Placement (Mid-Air vs. On-Arm) on completion times for the three subtasks in Phase 1. As the initial data was not normally distributed, I applied a log transform on the completion times and excluded seven outliers (1.9%). I followed a similar process for the number of retries for each subtask, which is the number of attempts made by the participant to complete a subtask and proceed to the next. I also performed a log transform and excluded two outliers (0.6%). For both of these measures, I found no significant differences ($p > 0.05$ for all three subtasks, Figure 5.1). A visual inspection of the mean completion time grouped per Trial Number showed a generally decreasing trend, with the first trial typically taking longer than subsequent trials (Figure 5.2).

The overall task load measured through the NASA TLX questionnaire followed a normal distribution, yet the UI Placement condition did not significantly affect the task load ($p = 0.38$). I mapped the 5-point Likert scale ratings for Ease of Use and Ease of Learning into values ranging between -2 to 2. For Ease of Learning, I excluded one outlier (4.2%); the ratings followed a normal distribution. I found no significant effect of the condition on Ease of Use nor on Ease of Learning ($p = 1$, respectively $p = 0.45$).

I also mapped the 5-point scale for the (Perceived) Calibration Accuracy of the On-Arm condition between -2 and 2. In this scale, -2 represents "Large offset from the real arm", while 2 represents "Very accurate". The ratings followed a normal distribution. I found no



(a) (b)

Figure 5.1: Mean (a) completion times in seconds and (b) number of retries for each subtask in Phase 1 for the two UI Placement conditions, with error bars showing one standard error of the mean. None of the differences were significant.

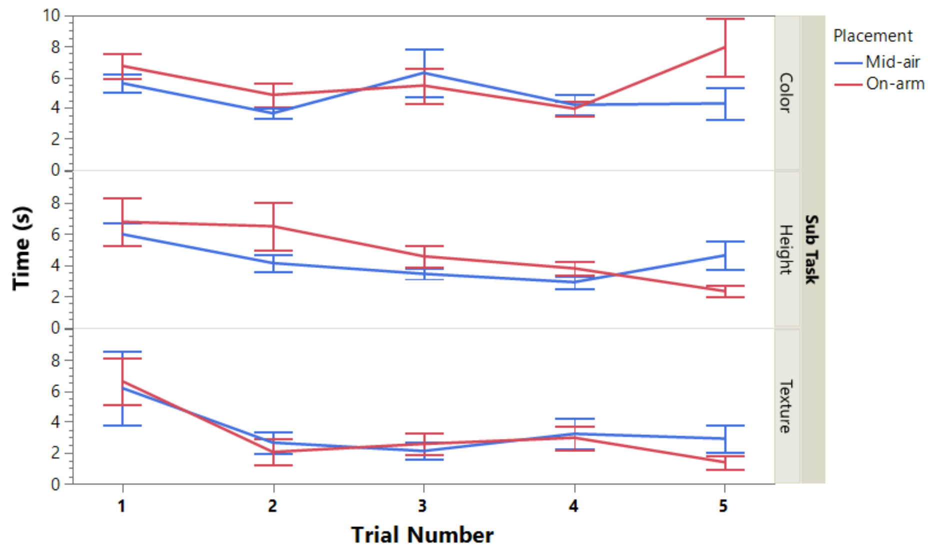
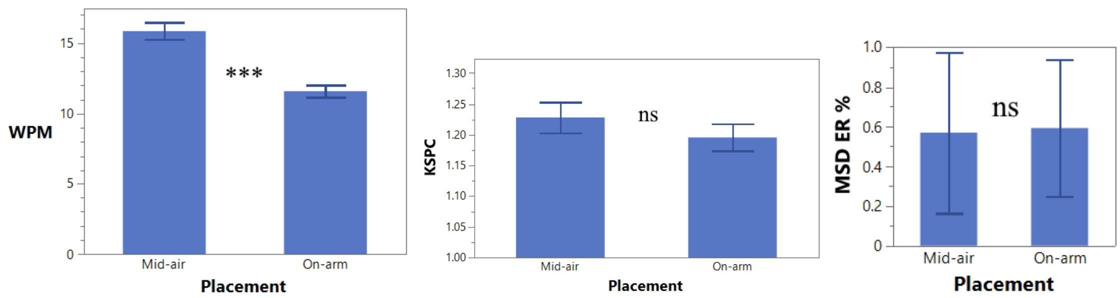


Figure 5.2: Mean completion times for each subtask in Phase 1 for the two UI Placement conditions by trial number, error bars showing one standard error of the mean.

significant effect of the different UI elements in Phase 1 (1D-Slider, 2D-Slider, and Touch panel) on the Perceived Calibration Accuracy ($p = 0.41$), i.e., the UI elements, on average, felt somewhat accurate for all UI elements ($M = 0.75, 0.17$, and 0.58, respectively).

5.2 Phase 2: QWERTY keyboard

Analyzing the text entry performance (in Words per Minute or WPM, where a word comprises of a sequence of any 5 characters [2]), I found the speed of using the keyboard to be normally distributed, with Mid-Air being significantly faster compared to the On-Arm keyboard (M = 15.88 Mid-Air, M = 11.6 On-Arm, $F(1, 11) = 22.37$, $p < 0.001$, Figure 5.3a). A visual inspection of the mean typing speed grouped per Trial or Phrase



(a) (b) (c)

Figure 5.3: Mean (a) WPM, (b) KSPC, and (c) MSD ER % by UI Placement condition, error bars showing one standard error of the mean. Mid-Air was significantly faster, but there were no significant differences in KSPC and MSD ER % between the conditions.

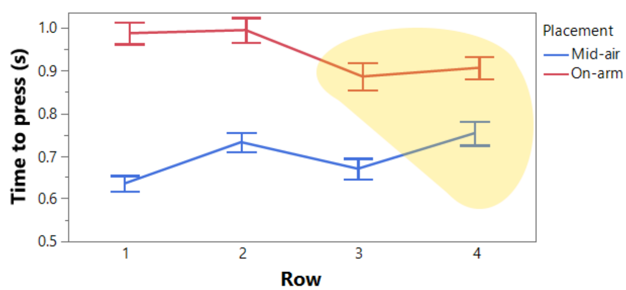
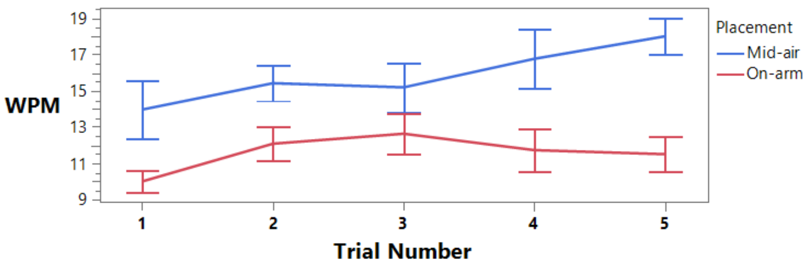


Figure 5.4: Mean time before key press by UI Placement and Keyboard Row Number, error bars showing one standard error of the mean. All Mid-Air combinations were significantly faster except for the group highlighted in yellow.

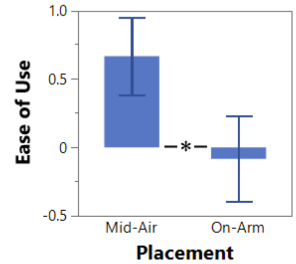
Number showed an increasing trend for the Mid-Air keyboard, and for the first 3 trials for the On-Arm keyboard (Figure 5.5a).

The log-transformed keystrokes per character (KSPC, i.e., the ratio of the total number of keys selected to the length of the typed text [55]) values followed a normal distribution. The placement of the keyboard, i.e., the two conditions, did not significantly affect KSPC ($p = 0.37$, Figure 5.3b). Additionally, I used the Minimum String Distance Error Rate (MSD ER) metric introduced by Soukoreff and MacKenzie [56] to calculate the error rate. I ran a Wilcoxon Signed Rank Test using SPSS 29 because the data did not follow a normal distribution. I found no significant differences in the MSD ER ($Z = -0.28, p = 0.78$, $M = 0.57\%$ and 0.59% in Mid-Air and On-Arm respectively, Figure 5.3c).

Digging deeper, I analyzed the effect of UI Placement and the keyboard row number on the time taken to hit a key. More specifically, I assigned row 1 as the top row, while row 4 was the spacebar. I performed a log transform on the times and removed 10 outliers (0.23%), after which the data conformed to a normal distribution. I found a significant main effect of UI Placement ($F(1, 11.62) = 32.41, p < 0.001$) but no significant main effect of keyboard row number ($p = 0.15$). The interaction was also significant ($F(3, 31.67) = 8.15, p < 0.001$), and all Mid-Air combinations were significantly faster except for the group comprising



(a)



(b)

Figure 5.5: Mean (a) WPM in Phase 2 separated by the trial number, and (b) Mean Ease of Use ratings for the two UI PLACEMENT conditions. The MID-AIR keyboard was rated significantly easier to use than the ON-ARM one. All error bars show one standard error of the mean.

(4, MID-AIR), (3, ON-ARM), and (4, ON-ARM), as shown by post-hoc Pairwise Tukey HSD tests. Beyond the statistical tests, a visual inspection showed that the differences in the mean time to hit a key were highest for row 1, suggesting that local registration issues might have contributed to the difference, i.e., some keyboard rows worked better than others ($Diff(M) = 0.35, 0.26, 0.22, 0.15$ for Rows 1 to 4, Figure 5.4).

Like Phase 1, I mapped the 5-point Likert scale ratings from -2 to 2 for further analysis. The overall task load measured through the NASA TLX questionnaire, Ease of Use ratings and Ease of Learning ratings followed a normal distribution. I did not find a significant difference between the MID-AIR and ON-ARM conditions for NASA TLX and Ease of Learning ratings ($p = 0.07$ and $p = 0.1$ respectively). However, I found the MID-AIR keyboard was perceived to be easier to use ($F(1, 11) = 5.21, p = 0.04$, Figure 5.5b).

5.3 Post-study questionnaire

Most participants preferred the MID-AIR placement of UI elements ($N = 8, 67\%$) over the ON-ARM placement. Still, when each participant was asked to justify their preference, five participants (42%) liked the physical sensation / haptic feedback provided by their arm in the ON-ARM condition. Three participants (25%) did not feel comfortable with pressing on or “poking” their arm to use the UI. Finally, four participants (33%) commented that they would prefer using ON-ARM UI for short tasks requiring fine control or adjustments.

Half of the participants found the AR mode for calibration helpful and easier to use than the VR mode ($N = 6, 50\%$). When asked to compare their experience using the on-arm and arm-anchored UI to a short video clip of UI interaction using raycasting which was shot from inside a Meta Quest 2 headset, seven participants (58%) stated that they would prefer having UI elements close to them, near or on the arm (similar to the experiment) instead of farther away. When asked to compare the proportions of the virtual arm model and their real arm, five participants mentioned localized registration issues (42%). Three participants

(25%) noticed the absence of full finger tracking, as the virtual hand models in the system were always shown as being in a relaxed pose.

Chapter 6

Discussion

My objective for the user study was to understand the effect of UI Placement on system control and text entry tasks. I discuss some salient findings from the study in this section.

6.1 Similar performance and task load for simple/short tasks

The three subtasks in Phase 1 (setting height, color, and texture of a virtual lamp) were designed to understand how UI Placement affected the performance and task load of short system control tasks. As can be seen from the results, I did not find a significant effect of UI Placement on the completion times and number of retries (see Figure 5.1). Further, I saw no significant differences in the multi-dimensional rating data collected after each condition – NASA TLX task load ratings, ratings for Ease of Use, and Ease of Learning for the UI elements. These findings do not support my hypothesis H1. H3 is also not supported in general except that Ease of Use was higher for Mid-Air in Phase 2, which refutes this aspect of H3.

One possible explanation is that the haptic feedback and touch sensations, which differentiate the On-Arm and Mid-Air conditions, do not significantly impact short system control tasks. Thus, the choice between the UI Placement for short tasks could be left to user preference. In the post-study questionnaire, four users (33%) mentioned they preferred the On-Arm UI for short tasks as it gave them a sense of more control or accuracy. They stated that their motivations were the sense of stability of working on the arm surface and added support from the haptic feedback.

Further, while the Mid-Air UI elements were completely flat, the 2D-Slider was curved to fit the shape of the calibrated virtual arm. The curvature was necessary to maintain a consistent level of physical contact with the arm. However, this could have exacerbated any issues arising from differences between the virtually modelled and real arm. The lower mean score for Perceived Calibration Accuracy for the 2D-Slider supports this argument. The need for finer calibration for the On-Arm condition could have led some participants to prefer the Mid-Air condition. Three participants (25%) noted the lessened need for fine calibration in the Mid-Air condition. P3 wrote, "*I felt more confident with mid-air controls.*"

6.2 Higher WPM with Mid-Air keyboard

I designed Phase 2 of the study to investigate the effect of UI Placement on typing tasks, which are longer and more complex than the subtasks in Phase 1. Phase 2 of my study was directly related to my hypothesis H2.

Participants achieved better typing speeds using the Mid-Air keyboard, as can be seen by the higher mean WPM, which refutes my Hypothesis H2 ($M = 15.88$ Mid-Air, $M = 11.6$ On-Arm, Figure 5.3a). Participants still maintained a similar KSPC and MSD Error Rate in both conditions, which fails to support H2 (Figure 5.3b, Figure 5.3c). The difference in typing speed can be largely attributed to localized registration issues, i.e., localized differences between the positions of the real and virtual arms' surfaces. My reasoning is supported by the row-wise analysis of the time taken to press a key, which showed us that participants took longer to press keys on the first, i.e., the top row of the QWERTY keyboard ($Diff(M) = 0.35, 0.26, 0.22, 0.15$ for Rows 1 to 4, Figure 5.4) with On-Arm, but not significantly so. Many participants found accessing the keys on the first row harder, sometimes making multiple attempts before the virtual key press was registered. The top row was located on an area of the forearm that is more curved than the comparatively flatter regions for Rows 2 and 3. Participants thus occasionally even rotated their arm slightly to get a better look at the top row when it curved away from their working view, which could have further affected their performance.

The Mid-Air keyboard was rated as easier to use by the participants, refuting this aspect of my hypothesis H3. However, similar to Phase 1, there were no significant differences between the UI Placement regarding Ease of Learning and NASA TLX Task Load, thus failing to support H3. One possible reason is that the angle of rotation and overall pose of the arm on the armrest changed between the Mid-Air and On-Arm conditions. For the Mid-Air condition, participants tended to place their hands closer to their bodies. The participants could still comfortably use the keyboard in this pose as the Mid-Air keyboard was floating 10 cm away from the arm. For the On-Arm keyboard, however, the participants sometimes needed to keep their hand further away from the body to bring the area with the UI into their view. The need to keep their forearm a bit further apart from the body and the angle at which they had to orient the headset to look at the On-Arm UI possibly contributed to the difference in the ratings. P6 specifically noted the pose of the arm in the On-Arm condition as being uncomfortable to maintain. Two participants commented on the headset's weight on their heads while working with the keyboard, as they needed to tilt their heads further down to look at the UI, which made them feel the headset's weight more. P5 commented that using the On-Arm keyboard felt like they were repeatedly "poking" themselves, which could be because of their long fingernails. P11 mentioned they were uncomfortable with pressing on their inner arm, especially virtual buttons near their wrist, as they suffered from a sensory processing disorder.

While the size of the keys was uniform across all conditions (1.2 cm $\times$ 1.2 cm), to fit the entire On-Arm keyboard onto one side of the arm, I used a fixed angular distance of 30° between rows of keys, which - together with the shape of the forearm - resulted in a slightly varying vertical spacing between rows of keys. This could have affected the ease of using the On-Arm keyboard. Four participants mentioned they would like to have bigger keys. Both P8 and P9 mentioned feeling constrained by the available space on their arm.

While the issue of small keys potentially affected both UI Placement conditions, typing only single letters at a time was still perceived to be more physically demanding than typing on a physical keyboard. P2 commented, "*I felt like an old person typing with just one finger when I usually use multiple fingers/both hands. Even on the phone when I'm 'typing' with just one finger, I'm using [a] swipe to type a whole word with one movement*". P6 said, "*I like having tactile clicks for each of the buttons even if they're just my arm. However, the effort it takes to maintain a pointing gesture and raise my arm up and down for each key is a lot more effort than traditional keyboard experiences.*"

6.3 Importance of Calibration and Robust Tracking

The importance of fine calibration and robust tracking was evident from my initial design explorations and pilots. By ensuring participants did not wear loose clothing, I avoided clothing being a confounding factor for the calibration. Also, based on my experience with the pilots, I moved away from hand-tracking and IK solutions in favour of external tracking with four Valve Base Stations 2.0, one at each corner of a relatively small rectangular area. This was to ensure optimal Base Station visibility for the Vive tracker on the elbow and robust tracking from all angles. Motivated by my observations from a pilot of the final study setup with 4 participants, I also developed the Fine Tuning mode as a final step for the calibration phase. The ability to switch between AR and VR was also developed in this stage. Half of the participants appreciated the ability to switch to the AR view for fine calibration ($N = 6, 50\%$). P1 commented, "*After switching to AR and doing fine-tuning, only then did the virtual arm feel similar to my real arm.*"

However, as discussed in the previous sections, many participants experienced localized registration issues and other artifacts arising from the differences in the pose and shape of the real arm and the parametric virtual arm. These issues could explain why most users ($N = 8, 67\%$) preferred the Mid-Air condition to the On-Arm condition, refuting part of Hypothesis H3. The experimenter tried to guide participants throughout the calibration process to adjust the offsets and make the virtual forearm's shape match as close as possible to their real forearm on all sides. The complexity of these registration issues became evident in the text entry task. As the keyboard extends over a large surface of the forearm, the keys became harder to access in specific areas whenever the forearm model was inadequate in faithfully representing the curvature of the forearm. In particular, the virtual keys were harder to press if placed below the real forearm's surface due to mis-registrations. On the other hand, if the keys were too far above the surface of the forearm, this resulted in accidental key presses and also suffered from inconsistent haptic feedback.

At first glance, a more complex forearm model may seem the best way forward. When asked to compare the calibrated virtual arm with their real arm in the Post Study Questionnaire, P3 noted, "*Some parts were slightly beneath my arm (near the wrist) while others were above my arm (towards my elbow).*" However, a more complex forearm model might have drawbacks. Despite the simple forearm model in the current implementation of VRAMBRACE, some participants identified the calibration process as being (too) long. P12 mentioned that after completing all tasks in Phase 1, "*The calibration took longer than the actual [Phase 1] tasks.*" Any further increase in the complexity of the arm model may thus adversely affect the learnability and intuitiveness of the calibration process.

6.4 Mode-Switching Extensions

Beyond the use of the controller actions as a mode-switching mechanism, in a controller-free implementation, VRAMBRACE could also use finger gestures of the non-dominant hand, such as extending the small finger, to act as a mode-switch when the user is using the VRAMBRACE keyboard, e.g., to serve as a Shift-key. As the user is not holding a controller, the user can then simply extend one or more of the fingers of the non-dominant hand, which can then also afford Ctrl and Alt states and combinations of all such states. I have not implemented this option in my current version of VRAMBRACE, but once fully integrated and reliable bi-manual forearm and hand-tracking through RGBD or RGB sensors is available, it should be easy to implement such functionality.

Chapter 7

Future Work and Conclusion

Motivated by the benefits of body-anchored interfaces, I developed VRambrace, a novel bimanual system control technique that places UI elements like sliders and a virtual keyboard on the user's forearm. I conducted a study with 12 participants to understand the effect of UI Placement on the performance of system control and text entry tasks.

Findings from the study suggest that UI Placement does not play a major role in the time taken to perform short interactions. Some users still preferred VRambrace for the added haptic feedback and sensation of touch between the fingertip and the surface of the forearm. Additionally, some users preferred the On-Arm UI for short tasks as it gave them a sense of better control or accuracy. These findings could motivate future studies investigating on-arm 'quick menus' for VR games [27, pp. 415], as system control tasks (e.g. switching between active weapons in a shooter game) need to be performed quickly and accurately. Additionally, many participants reported the need for a selection "locking" mechanism while interacting with the 2D-Slider in both UI Placement conditions, as the selected value would sometimes change while the participant moved their index finger away from the UI element (due to slight misregistrations between the real and virtual forearm surfaces). The effect of such changes to the interaction could be explored in future studies. Future studies comparing arm-anchored and on-arm UI against spatial / floating UI elements could also clarify the effect of focus switching on performance [27, pp. 382].

On the other hand, I found users preferred and performed better in text entry tasks with the Mid-Air virtual keyboard compared to the On-Arm one. My findings from the in-depth analysis suggest that more complex and longer tasks like text entry would benefit from finer forearm calibration and modelling, with special attention needed for ergonomics to enable use over longer sessions. Future work could thus investigate variations in key sizes, e.g., with smaller keys near the flatter middle sections and larger keys at the periphery. Alternative keyboard layouts that consider letter frequencies, e.g., OPTI [39], might also lead to better performance. A swipe gesture keyboard could also be investigated, which matches single-finger usage well.

Participants were comfortably seated throughout the experiment, resting their arms on the armrests. Changing their posture could affect the task load and possibly their performance for short tasks. Still, lighter and more ergonomic headsets would also be crucial for long and complex interaction tasks like text entry.

A quantitative/objective method to characterize the calibration and modelling accuracy at multiple points along the arm could be a good launch point for future studies investigating the effect of forearm registration. An expert user could then even be made aware of the accuracy at different locations and then customize the placement of UI elements also to suit their needs and preferences. A more complex parametric representation of the arm or using RGBD cameras to directly sense the forearm are potential technical improvements. However, each new forearm tracking method will need its corresponding calibration procedure, for which the calibration technique presented in this thesis could serve as a starting point. Tracking more than the index finger would also expand the set of interaction methods, e.g., to make classic multitouch gestures like pinch-zoom on the forearm a reality. Additionally, the combination of tracked devices used in my prototype (Index/Knuckle controllers with a Vive tracker near the elbow) lays a potential foundation for future explorations of eyes-free interactions, similar to Pub [31]. In future iterations, and similar to Tran et al. [59], the current visual and auditory feedback in VRambrace could be supplemented with vibrotactile feedback, either through nail-mounted vibration motors or handheld VR controllers. Future studies should also explore the effect of VRambrace's AR mode on perceived calibration accuracy, on the performance of system control and text entry tasks, and other design challenges arising from performing such tasks in visually cluttered environments.

Bibliography

[1] Ferran Argelaguet and Carlos Andujar. A survey of 3D object selection techniques for virtual environments. *Computers & Graphics*, 37(3):121–136, May 2013.

[2] Ahmed Sabbir Arif and Wolfgang Stuerzlinger. Analysis of text entry performance metrics. In *2009 IEEE Toronto International Conference Science and Technology for Humanity (TIC-STH)*, pages 100–105, September 2009.

[3] Takumi Azai, Shuhei Ogawa, Mai Otsuki, Fumihisa Shibata, and Asako Kimura. Selection and Manipulation Methods for a Menu Widget on the Human Forearm. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '17, pages 357–360, New York, NY, USA, May 2017. Association for Computing Machinery.

[4] Takumi Azai, Mai Otsuki, Fumihisa Shibata, and Asako Kimura. Open Palm Menu: A Virtual Menu Placed in Front of the Palm. In *Proceedings of the 9th Augmented Human International Conference*, AH '18, pages 1–5, New York, NY, USA, February 2018. Association for Computing Machinery.

[5] Takumi Azai, Syunsuke Ushiro, Junlin Li, Mai Otsuki, Fumihisa Shibata, and Asako Kimura. Tap-tap menu: body touching for virtual interactive menus. In *Proceedings of the 24th ACM Symposium on Virtual Reality Software and Technology*, VRST '18, pages 1–2, New York, NY, USA, November 2018. Association for Computing Machinery.

[6] Anil Ufuk Batmaz, Mayra Donaji Barrera Machuca, Junwei Sun, and Wolfgang Stuerzlinger. The Effect of the Vergence-Accommodation Conflict on Virtual Hand Pointing in Immersive Displays. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, pages 1–15, New York, NY, USA, April 2022. Association for Computing Machinery.

[7] Anil Ufuk Batmaz, Aunnoy K Mutasim, Morteza Malekmakan, Elham Sadr, and Wolfgang Stuerzlinger. Touch the Wall: Comparison of Virtual and Augmented Reality with Conventional 2D Screen Eye-Hand Coordination Training Systems. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 184–193, March 2020. ISSN: 2642-5254.

[8] Anil Ufuk Batmaz and Wolfgang Stuerzlinger. The Effect of Pitch in Auditory Error Feedback for Fitts' Tasks in Virtual Reality Training Systems. In *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*, pages 85–94, March 2021. ISSN: 2642-5254.

[9] Joanna Bergstrom-Lehtovirta, Kasper Hornbæk, and Sebastian Boring. It's a Wrap: Mapping On-Skin Input to Off-Skin Displays. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 1–11, New York, NY, USA, April 2018. Association for Computing Machinery.

[10] D.A. Bowman and C.A. Wingrave. Design and evaluation of menu systems for immersive virtual environments. In *Proceedings IEEE Virtual Reality 2001*, pages 149–156, March 2001.

[11] Frederik Brudy. Interactive Menus in Augmented Reality Environments. Technical Report LMU-MI-2013-1, Ludwig-Maximillian-University Munich, April 2013.

[12] Jeff Butterworth, Andrew Davidson, Stephen Hench, and Marc. T. Olano. 3DM: a three dimensional modeler using a head-mounted display. In *Proceedings of the 1992 symposium on Interactive 3D graphics*, I3D '92, pages 135–138, New York, NY, USA, June 1992. Association for Computing Machinery.

[13] Polona Caserman, Philipp Achenbach, and Stefan Göbel. Analysis of Inverse Kinematics Solutions for Full-Body Reconstruction in Virtual Reality. In *2019 IEEE 7th International Conference on Serious Games and Applications for Health (SeGAH)*, pages 1–8, August 2019. ISSN: 2573-3060.

[14] Raimund Dachselt and Anett Hübner. Three-dimensional menus: A survey and taxonomy. *Computers & Graphics*, 31(1):53–65, January 2007.

[15] Tafadzwa Joseph Dube and Ahmed Sabbir Arif. Text Entry in Virtual Reality: A Comprehensive Review of the Literature. In Masaaki Kurosu, editor, *Human-Computer Interaction. Recognition and Interaction Technologies*, Lecture Notes in Computer Science, pages 419–437, Cham, 2019. Springer International Publishing.

[16] Jorge A. Wagner Filho, Wolfgang Stuerzlinger, and Luciana Nedel. Evaluating an Immersive Space-Time Cube Geovisualization for Intuitive Trajectory Data Exploration. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):514–524, January 2020.

[17] 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, April 2013.

[18] Jens Grubert, Lukas Witzani, Eyal Ofek, Michel Pahud, Matthias Kranz, and Per Ola Kristensson. Effects of Hand Representations for Typing in Virtual Reality. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 151–158, March 2018.

[19] Joseph F Hair Jr, William C Black, Barry J Babin, and Rolph E. Anderson. *Multivariate data analysis*. Pearson Education Limited, 2014.

[20] Chris Harrison, Hrvoje Benko, and Andrew D. Wilson. OmniTouch: wearable multitouch interaction everywhere. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, UIST '11, pages 441–450, New York, NY, USA, October 2011. Association for Computing Machinery.

[21] Chris Harrison, Shilpa Ramamurthy, and Scott E. Hudson. On-body interaction: armed and dangerous. In *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction*, TEI '12, pages 69–76, New York, NY, USA, February 2012. Association for Computing Machinery.

[22] Chris Harrison, Desney Tan, and Dan Morris. Skinput: appropriating the body as an input surface. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 453–462, New York, NY, USA, April 2010. Association for Computing Machinery.

[23] Zhenyi He, Christof Lutteroth, and Ken Perlin. TapGazer: Text Entry with Finger Tapping and Gaze-directed Word Selection. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, pages 1–16, New York, NY, USA, April 2022. Association for Computing Machinery.

[24] Richard H. Jacoby and Stephen R. Ellis. Using virtual menus in a virtual environment. In *Visual Data Interpretation*, volume 1668, pages 39–48. SPIE, June 1992.

[25] Janis G. Jimenez and Jürgen P. Schulze. Continuous-Motion Text Input in Virtual Reality. *Electronic Imaging*, 30:1–6, January 2018.

[26] Luv Kohli and Mary Whitton. The haptic hand: providing user interface feedback with the non-dominant hand in virtual environments. In *Proceedings of Graphics Interface 2005*, GI '05, pages 1–8, Waterloo, CAN, May 2005. Canadian Human-Computer Communications Society.

[27] Joseph J. LaViola, Ernst Kruijff, Ryan P. McMahan, Doug A. Bowman, and Ivan Poupyrev. *3D User Interfaces: Theory and Practice*. Addison-Wesley, 2017.

[28] Irina Lediaeva and Joseph LaViola. Evaluation of Body-Referenced Graphical Menus in Virtual Environments. In *Proceedings of Graphics Interface 2020*, GI 2020, pages 308 – 316. Canadian Human-Computer Communications Society / Société canadienne du dialogue humain-machine, May 2020.

[29] Zhen Li, Joannes Chan, Joshua Walton, Hrvoje Benko, Daniel Wigdor, and Michael Glueck. Armstrong: An Empirical Examination of Pointing at Non-Dominant Arm-Anchored UIs in Virtual Reality. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, pages 1–14, New York, NY, USA, May 2021. Association for Computing Machinery.

[30] Jiandong Liang and Mark Green. JDCAD: A highly interactive 3D modeling system. *Computers & Graphics*, 18(4):499–506, July 1994.

[31] Shu-Yang Lin, Chao-Huai Su, Kai-Yin Cheng, Rong-Hao Liang, Tzu-Hao Kuo, and Bing-Yu Chen. Pub - point upon body: exploring eyes-free interaction and methods on an arm. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, UIST '11, pages 481–488, New York, NY, USA, October 2011. Association for Computing Machinery.

[32] Xueshi Lu, Difeng Yu, Hai-Ning Liang, Wenge Xu, Yuzheng Chen, Xiang Li, and Khalad Hasan. Exploration of Hands-free Text Entry Techniques For Virtual Reality.

In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 344–349, November 2020. ISSN: 1554-7868.

[33] I. Scott MacKenzie and R. William Soukoreff. Phrase sets for evaluating text entry techniques. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '03, pages 754–755, New York, NY, USA, April 2003. Association for Computing Machinery.

[34] Paul Mallery and Darren George. *SPSS for Windows step by step: a simple guide and reference*. Pearson, 2003.

[35] Manuel Meier, Paul Streli, Andreas Fender, and Christian Holz. TapID: Rapid Touch Interaction in Virtual Reality using Wearable Sensing. In *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*, pages 519–528, March 2021. ISSN: 2642-5254.

[36] Mark Mine. ISAAC: A virtual environment tool for the interactive construction of virtual worlds. *UNC Chapel Hill Computer Science Technical Report TR95-020*, 1995.

[37] Mark R. Mine, Frederick P. Brooks, and Carlo H. Sequin. Moving objects in space: exploiting proprioception in virtual-environment interaction. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '97, pages 19–26, USA, August 1997. ACM Press/Addison-Wesley Publishing Co.

[38] Pedro Monteiro, Hugo Coelho, Guilherme Gonçalves, Miguel Melo, and Maximino Bessa. Comparison of Radial and Panel Menus in Virtual Reality. *IEEE Access*, 7:116370–116379, 2019.

[39] Aunnoy K Mutasim, Anil Ufuk Batmaz, Moaaz Hudhud Mughrabi, and Wolfgang Stuerzlinger. Does Repeatedly Typing the Same Phrase Provide a Good Estimate of Expert Text Entry Performance? In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI EA '23, pages 1–8, New York, NY, USA, April 2023. Association for Computing Machinery.

[40] Aunnoy K Mutasim, Anil Ufuk Batmaz, and Wolfgang Stuerzlinger. Pinch, Click, or Dwell: Comparing Different Selection Techniques for Eye-Gaze-Based Pointing in Virtual Reality. In *ACM Symposium on Eye Tracking Research and Applications*, ETRA '21 Short Papers, pages 1–7, New York, NY, USA, May 2021. Association for Computing Machinery.

[41] Mathias Parger, Joerg H. Mueller, Dieter Schmalstieg, and Markus Steinberger. Human upper-body inverse kinematics for increased embodiment in consumer-grade virtual reality. In *Proceedings of the 24th ACM Symposium on Virtual Reality Software and Technology*, VRST '18, pages 1–10, New York, NY, USA, November 2018. Association for Computing Machinery.

[42] Chanho Park, Hyunwoo Cho, Sangheon Park, Young-Suk Yoon, and Sung-Uk Jung. HandPoseMenu: Hand Posture-Based Virtual Menus for Changing Interaction Mode in 3D Space. In *Proceedings of the 2019 ACM International Conference on Interactive Surfaces and Spaces*, ISS '19, pages 361–366, New York, NY, USA, November 2019. Association for Computing Machinery.

[43] Duc-Minh Pham and Wolfgang Stuerzlinger. HawKEY: Efficient and Versatile Text Entry for Virtual Reality. In *Proceedings of the 25th ACM Symposium on Virtual Reality Software and Technology*, VRST '19, pages 1–11, New York, NY, USA, November 2019. Association for Computing Machinery.

[44] W. Piekarski and B.H. Thomas. Tinmith-Hand: unified user interface technology for mobile outdoor augmented reality and indoor virtual reality. In *Proceedings IEEE Virtual Reality 2002*, pages 287–288, March 2002. ISSN: 1087-8270.

[45] Wayne Piekarski and Bruce 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, 2002.

[46] Majid Pourmemar and Charalambos Poullis. Visualizing and Interacting with Hierarchical Menus in Immersive Augmented Reality. In *Proceedings of the 17th International Conference on Virtual-Reality Continuum and its Applications in Industry*, VRCAI '19, pages 1–9, New York, NY, USA, November 2019. Association for Computing Machinery.

[47] J. Preece, H. Sharp, and Y. Rogers. *Interaction Design: Beyond Human-Computer Interaction*. Wiley, 2015.

[48] Kristin Reich, Elisabeth Jäger, Xuan Linh Do, and Katrin Wolf. ArmTouch: The Forearm as Touchpad for VR Control. In *Proceedings of Mensch und Computer 2019*, MuC '19, pages 897–899, New York, NY, USA, September 2019. Association for Computing Machinery.

[49] Katharina Reiter, Ken Pfeuffer, Augusto Esteves, Tim Mittermeier, and Florian Alt. Look & Turn: One-handed and Expressive Menu Interaction by Gaze and Arm Turns in VR. In *2022 Symposium on Eye Tracking Research and Applications*, ETRA '22, pages 1–7, New York, NY, USA, June 2022. Association for Computing Machinery.

[50] RootMotion. Final IK | Animation Tools | Unity Asset Store.

[51] Chris Shaw and Mark Green. Two-handed polygonal surface design. In *Proceedings of the 7th annual ACM symposium on User interface software and technology*, UIST '94, pages 205–212, New York, NY, USA, November 1994. Association for Computing Machinery.

[52] Chris Shaw, Mark Green, Jiandong Liang, and Yunqi Sun. Decoupled simulation in virtual reality with the MR toolkit. *ACM Transactions on Information Systems*, 11(3):287–317, July 1993.

[53] Chris Shaw, Jiandong Liang, Mark Green, and Yunqi Sun. The decoupled simulation model for virtual reality systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '92, pages 321–328, New York, NY, USA, June 1992. Association for Computing Machinery.

[54] Zhaomou Song, John J. Dudley, and Per Ola Kristensson. Efficient Special Character Entry on a Virtual Keyboard by Hand Gesture-Based Mode Switching. In *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 864–871, October 2022. ISSN: 1554-7868.

[55] R. William Soukoreff and I. Scott MacKenzie. Measuring errors in text entry tasks: an application of the Levenshtein string distance statistic. In *CHI '01 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '01, pages 319–320, New York, NY, USA, March 2001. Association for Computing Machinery.

[56] R. William Soukoreff and I. Scott MacKenzie. Metrics for text entry research: an evaluation of MSD and KSPC, and a new unified error metric. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, pages 113–120, New York, NY, USA, April 2003. Association for Computing Machinery.

[57] Marco Speicher, Anna Maria Feit, Pascal Ziegler, and Antonio Krüger. Selection-based Text Entry in Virtual Reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 1–13, New York, NY, USA, April 2018. Association for Computing Machinery.

[58] Paul Streli, Jiaxi Jiang, Andreas Rene Fender, Manuel Meier, Hugo Romat, and Christian Holz. TapType: Ten-finger text entry on everyday surfaces via Bayesian inference. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, pages 1–16, New York, NY, USA, April 2022. Association for Computing Machinery.

[59] Peter Khoa Duc Tran, Purna Valli Anusha Gadepalli, Jaeyeon Lee, and Aditya Shekhar Nittala. Augmenting On-Body Touch Input with Tactile Feedback Through Fingernail Haptics. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, pages 1–13, New York, NY, USA, April 2023. Association for Computing Machinery.

[60] Dong-Bach Vo, Eric Lecolinet, and Yves Guiard. Belly gestures: body centric gestures on the abdomen. In *Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational*, NordiCHI '14, pages 687–696, New York, NY, USA, October 2014. Association for Computing Machinery.

[61] Cheng-Yao Wang, Wei-Chen Chu, Po-Tsung Chiu, Min-Chieh Hsiu, Yih-Harn Chiang, and Mike Y. Chen. PalmType: Using Palms as Keyboards for Smart Glasses. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '15, pages 153–160, New York, NY, USA, August 2015. Association for Computing Machinery.

[62] Robert Xiao, Teng Cao, Ning Guo, Jun Zhuo, Yang Zhang, and Chris Harrison. LumiWatch: On-Arm Projected Graphics and Touch Input. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 1–11, New York, NY, USA, April 2018. Association for Computing Machinery.

[63] Caglar Yildirim and Ethan Osborne. Text Entry in Virtual Reality: A Comparison of 2D and 3D Keyboard Layouts. In Constantine Stephanidis, Jessie Y. C. Chen, and Gino Fragomeni, editors, *HCI International 2020 – Late Breaking Papers: Virtual and Augmented Reality*, Lecture Notes in Computer Science, pages 450–460, Cham, 2020. Springer International Publishing.

[64] Difeng Yu, Qiushi Zhou, Tilman Dingler, Eduardo Velloso, and Jorge Goncalves. Blending On-Body and Mid-Air Interaction in Virtual Reality. In *2022 IEEE International*

Symposium on Mixed and Augmented Reality (ISMAR), pages 637–646, October 2022. ISSN: 1554-7868.

[65] M. Al Zayer, P. MacNeilage, and E. Folmer. Virtual Locomotion: A Survey. *IEEE Transactions on Visualization and Computer Graphics*, 26(6):2315–2334, June 2020.

Appendix A

Questionnaires

A.1 Pre-Study Questionnaire

1. What is your age?
2. What is your gender?
3. Which hand do you use as your primary hand for day-to-day tasks?
4. How familiar are you with Virtual and Augmented Reality Systems?
5. How frequently do you use 3D / CAD tools or software (for example, AutoCAD, Unity)?
6. How frequently do you play computer games?

A.2 Phase 1

1. Ease of use
2. Calibration for Color panel (only for on-arm condition)
3. Calibration for Touch panel with textures (only for on-arm condition)
4. Calibration for 1D-Slider to change height (only for on-arm condition)
5. Ease of learning to use the system
6. NASA TLX
7. Any comments about the sliders and the touch panel

A.3 Phase 2

1. Ease of use
2. Ease of learning to use the system
3. NASA TLX
4. Any comments about the keyboard

A.4 Post-Study Questionnaire

1. Preference (highest at the top)
2. Why did you prefer one over the other? Compare your experience using UI components like sliders, swipe touch panels and a keyboard floating near your arm with those directly on your arm. How was it different?
3. Comments about switching to Augmented Reality for (a) calibration and (b) exploration session (at the end)
4. How do you feel about having interfaces and menu systems on the arm? (in contrast to menus on screens floating in front of you, similar to the video shown of Quest 2)
5. Did the virtual arm model feel similar in proportions to your real arm after the calibration process? Any other comments about the calibration process
6. Any other comments

Appendix B

Themes in Comments

In this chapter, I present the themes I identified and the cluster of participants that made them. Each table corresponds to the data from one of the free-form text boxes in the study, except for Overall Preference, which was a ranking question.

Table B.1: Phase 1 Comment Box

Theme	Participants	Count
Sliders need better selecting/locking mechanism	P2, P3, P6, P7, P11, P12	6

Table B.2: Phase 2 Comment Box

Theme	Participants	Count
Typing felt like poking	P5	1
Need a bigger keyboard	P3, P4, P8, P9	4
Felt constrained by arm surface	P8, P9	2
Equipment heavy	P2, P7	2

Table B.3: Post Study Questionnaire – Overall Preference

Theme	Participants	Count
Preferred Mid-air	P2, P3, P4, P5, P7, P8, P10, P11	8
Preferred On-arm	P1, P6, P9, P12	4

Table B.4: Post Study Questionnaire – Justify preference

Theme	Participants	Count
On-arm haptic feedback	P1, P2, P6, P9, P12	5
Discomfort touching, poking	P3, P5, P7	3
Prefer on-arm for short, fine interactions	P1, P6, P9, P12	4
Lesser need for calibration in mid-air	P4, P7, P10	3

Table B.5: Post Study Questionnaire – AR Calibration

Theme	Participants	Count
Calibration in AR helped	P1, P3, P6, P9, P10, P12	6

Table B.6: Post Study Questionnaire – Screen vs. Arm

Theme	Participants	Count
Prefer arm anchored	P2, P3, P5, P9, P10, P11, P12	7

Table B.7: Post Study Questionnaire – Proportions of virtual arm after calibration

Theme	Participants	Count
Calibration problems	P2, P3, P7, P9, P12	5
Problems near wrist and elbow	P3	1
Problems with finger tracking	P2, P6, P12	3