Multimodal user interaction methods for virtual reality headsets

Pallavi, Mohan

2020

Pallavi, M. (2020). Multimodal user interaction methods for virtual reality headsets.
Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/137005>

<https://doi.org/10.32657/10356/137005>

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0
International License (CC BY-NC 4.0).

Downloaded on 26 Apr 2023 16:29:44 SGT

MULTIMODAL USER INTERACTION METHODS FOR VIRTUAL REALITY HEADSETS

MOHAN PALLAVI

School of Computer Science & Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

2020

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

20 Aug. 2019

.....

Date

.....

MOHAN PALLAVI

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

20 Aug. 2019

.....

Date

.....

A/P. Goh Wooi Boon

Authorship Attribution Statement

This thesis contains material from [3 number] paper(s) published in the following peer-reviewed journal(s) / from papers accepted at conferences in which I am listed as an author.

Chapter 3 is published in [1] as [Xiaoqi Yan, Chi-Wing Fu, Pallavi Mohan, and Wooi Boon Goh. 2016. CardboardSense: Interacting with DIY Cardboard VR Headset by Tapping. In Proceedings of the 2016 ACM Conference on Designing Interactive Systems \(DIS '16\). ACM, New York, NY, USA, 229-233. DOI: 10.1145/2901790.2901813](#)

The contributions of the co-authors are as follows:

- Dr.Xiaoqi Yan created the initial system, the iPhone-based application to retrieve tap patterns on Cardboard, trained the system, conducted user evaluations and helped write the paper.
- Prof. Chi-Wing Fu provided the initial project direction, guided the work closely, validated results and wrote major parts of the paper.
- I created an Android-based application to retrieve vibration patterns due to tapping on the Cardboard, and another to recognise the different gestures. I also used an SVM classifier to train the system to recognize and distinguish between left, right, top and bottom taps, as well as double taps, and extracted performance metrics from all of these, as shown in the paper. I helped to write some sections of the paper, and created a video for it too.
- A/P Goh Wooi Boon guided the work closely, helped with the video, and helped write the paper too.

Chapter 4 is published in [2] as [P. Mohan, W. B. Goh, C. Fu and S. Yeung, DualGaze: Addressing the Midas Touch Problem in Gaze Mediated VR Interaction, 2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct \(ISMAR-Adjunct\), Munich, Germany, 2018, pp. 79-84. DOI: 10.1109/ISMAR-Adjunct.2018.00039](#) The contributions of the co-authors are as follows:

- I developed the interaction design, coded the applications, conducted user evaluations, analysed the results, made the video and wrote the paper.
- A/P Goh Wooi Boon provided the initial work direction, guided the work closely, and edited and helped write the paper too.
- Prof. Chi-Wing Fu guided the work, and edited and revised the paper.
- A/P. Sai-Kit Yeung guided the work, and edited and revised the paper.

A section of Chapter 6 is published in [3] as Pallavi Mohan, Wooi Boon Goh, Chi-Wing Fu, and Sai-Kit Yeung. 2019. Head-Fingers-Arms: Physically-Coupled and Decoupled Multimodal Interaction Designs in Mobile VR. In The 17th International Conference on Virtual-Reality Continuum and its Applications in Industry (VRCAI '19). Association for Computing Machinery, New York, NY, USA, Article 28, 1–9. The contributions of the co-authors are as follows:

- I developed the three interaction designs, coded the applications, conducted user evaluations, analysed the results, made the video and wrote the paper.
- A/P Goh Wooi Boon provided the initial work direction, guided the work closely, and edited and helped write the paper too.
- Prof. Chi-Wing Fu guided the work, and edited and revised the paper.
- A/P. Sai-Kit Yeung guided the work, and edited and revised the paper.

20 Aug 2019

.....

Date

.....

MOHAN PALLAVI

Acknowledgements

Firstly and most importantly, I would like to thank my supervisor, Assoc. Prof. Goh Wooi Boon, for his patient guidance and continuous support. He has been unfailingly helpful and has chaperoned me with much-needed wisdom and advice throughout the course of my research. I would also like to thank my co-supervisor Assoc. Prof. Sai-Kit Yeung from HKUST, as well as Prof. Chi-Wing Fu from CUHK for their constant encouragement, insights and discussions regarding my research.

Special thanks is in order to my mother, Namita Mohanbabu, who has been my backbone of support throughout these four years, encouraging me to push through without giving up at every difficult juncture. She has believed in me, inspired me and pushed me to succeed in ways I am endlessly grateful for. I am also grateful to the rest of my family: my brother, Aditya Mohan, for his cheer, support and wisdom beyond his age; my grandparents, Achutha Menon and Bhama Menon, for their relentless optimism and comfort during difficult times. They always believed I would make it.

I would like to thank my friends and family in Singapore - they have been my systems of support throughout. In particular, I thank my cousins, Dr. Prathibha Menon and Asuthosh Nair, for being my family away from home. Lastly, I thank my friends in NTU, who have been no less than family to me. With their support, research hurdles cease to seem insurmountable.

“Books are the original virtual reality.”

—Marushia Dark

To my dearest father, Manghat Mohanbabu

Abstract

Despite the popularity of VR, there is a lack of research in interaction methods for VR HMDs. In their confined setting, conventional VR headsets provide only limited input modalities. Commonly-supported interaction methods include tracking the user’s head orientation, external controllers and buttons, and more recently, gaze interaction by eye tracking. This research aims to widen the currently available range of input modalities in virtual reality, by developing novel interaction designs and exploring multimodal interaction in mobile VR.

The first work in this research aims to add a novel interaction method for a DIY cardboard VR headset that enfolds a commodity smartphone. A set of novel tap-gesture-based inputs have been developed that take advantage of the motion sensing capabilities available in smartphones. Experimental results have shown promising potential for this system to extend the interactivity of future cardboard-based VR headset applications. The second work in this research explores eye gaze tracking as a feasible interaction modality in VR, and understands some of its challenges. It presents DualGaze, a novel gaze-based interaction method developed to address the Midas Touch problem for gaze mediated VR interaction. In this, users perform a distinctive two-step gaze gesture for object selection. A user study was conducted to compare the accuracy and selection speed of DualGaze and the popular gaze fixation method on a simple gaze-typing task. The results show that DualGaze is significantly more accurate while maintaining a comparable selection speed that was observed to improve with familiarity of use.

The next part of the research explores hand and finger gesture-based interaction in VR using the smartphone camera inside a VR HMD, and vision-based tracking methods. The research also explores the effects of combining these modalities together with head gaze-based user input in VR, and proposes a few novel interaction designs. The first interaction design developed in this modality is a novel two-handed gesture interaction design that is invariant to head motion, called Twin-Fingers. The technique provides an interactive 2D cursor control in

VR, where the user slides two overlapping splayed fingers over two from the other hand to perform the gesture. In addition, a conventional single-hand gesture design is also considered and its performance is compared with Twin-Fingers on an onscreen cursor-based pointing task that involves both head movement and finger-gesture interaction. Our user study shows that most users prefer the simplicity and convenience of a single-hand gesture for tasks without requiring head movement. However, with limited practice, users performed better with Twin-Fingers when tackling tasks that required both head movement and finger gestures.

When it was found that Twin-Fingers required a considerable learning curve, another interaction design was developed in a similar vein, called X-Fingers, which involved only one (index) finger from the dominant hand sliding over the index finger of the non-dominant hand. X-Fingers required less effort to perform the gesture by the user, while retaining Twin-Fingers' properties of invariance to hand and head motion; this led to the idea that this interaction design can be coordinated with the movement of the user's arms or head to provide an additional input modality. The incorporation of the arms or the head provides physically-coupled and physically-decoupled multimodal interactions respectively. Given these two design options, user studies were conducted to understand how the nature of the physical coupling of interactions influences the user's performance with tasks of varying degrees of coordination between the modalities. The results show that physically-decoupled interactions designs are preferred when the degree of coordination is high within the multimodal interaction. Moreover, experiments were also conducted to investigate user performances in object positioning tasks of the coupled interaction modality with arms together versus a decoupled arms-apart modality. Results show that user performances and preferences depend on the target size and task type for each interaction modality. Combining the findings from these user evaluations, the research presents guidelines for VR interaction design for task types and coordination levels based on users' performances and preferences.

By presenting designs for hand and head interactions along with the previous work on tap and eye-based interactions, this research broadens the existing input modalities available in VR HMDs, thus allowing for richer user experiences in mobile VR.

Contents

Acknowledgements	ix
Abstract	xiii
List of Figures	xix
List of Tables	xxv
Symbols and Acronyms	xxvii
1 Introduction	1
1.1 Motivation	3
1.2 Objectives	5
1.3 Major Contributions of the Thesis	6
1.4 Organization of the Thesis	8
2 Literature Review	9
2.1 Introduction	9
2.2 Virtual Reality - A Background	9
2.2.1 Early VR Systems	10
2.2.2 VR Environments	11
2.3 VR Headsets	12
2.3.1 Google Cardboard	13
2.3.2 Google Daydream	15
2.3.3 Playstation VR	16
2.3.4 Oculus Quest	17
2.4 Human Input-Based Interaction in VR Headsets	17
2.4.1 Head Gaze-based Interactions	18
2.4.2 Eye Gaze-based Interactions	18
2.4.3 Hand and Finger-based Interactions	19
2.4.4 Other Human Input-based Interactions	19
2.5 Interacting with VR Applications	20
2.5.1 Text and Numerical Input	21
2.5.2 Object Selection	22

2.5.3	Object Manipulation	22
2.6	Summary	24
3	Tapping Interactions in VR Headsets	25
3.1	Introduction	25
3.2	Related Work	28
3.3	Interaction Design	30
3.4	Implementation and Results	36
3.4.1	Hardware	36
3.4.2	Software	36
3.4.3	Experimental Conditions	36
3.5	Results	37
3.5.1	Evaluation: Performance	37
3.5.2	User evaluations: Recognition Accuracy	37
3.5.3	User Subjective Feedback	38
3.5.4	Discussion	38
3.6	Basic Interaction Design Guideline	40
3.6.1	Enhancing Interactivity of Potential Cardboard Apps	41
3.6.2	Activation of Buttons and Selection of Menu Items	41
3.7	Conclusion	43
4	Gaze-based Interactions in VR Headsets	45
4.1	Introduction	45
4.2	Related Work	48
4.2.1	Gaze-based Selection in VR	49
4.3	The DualGaze Interaction Method	50
4.3.1	Designing the DualGaze Interaction	52
4.3.1.1	User Study on DualGaze Design	53
4.3.2	A Comparative Gaze-based Interaction Method	54
4.4	User Study	55
4.4.1	Experimental Design and Set-Up	55
4.4.2	Participants and Procedure	56
4.4.2.1	Participants	56
4.4.2.2	Procedure	56
4.5	Results	56
4.5.1	Accuracy	56
4.5.2	Selection Speed	57
4.5.3	Temporal Performance Changes	58
4.6	Discussion	60
4.6.1	Accuracy	60
4.6.2	Selection Speed	62
4.7	Conclusion	63
5	Single-Input Interactions with Fingers	65

5.1	Introduction	65
5.2	Related Work	67
5.3	Interaction Design S1: Twin-Fingers (TF)	68
5.3.1	Interaction Design	69
5.3.1.1	Calibration and Scale Invariance	71
5.3.1.2	Head Motion Invariance	72
5.3.1.3	Hand Motion Invariance	73
5.4	Interaction Design S2: Bunny-Fingers (BF)	74
5.5	Evaluation: TF and BF	75
5.5.1	Experimental Design and Set-Up	76
5.5.1.1	Participants and Procedure	77
5.5.1.2	Results and Discussion	78
5.6	Interaction Design S3: X-Fingers (XF)	80
5.6.1	Interaction Design	80
5.7	Evaluation: BF and XF	82
5.7.1	Experimental Design and Set-Up	83
5.7.1.1	Participants and Procedure	84
5.7.1.2	Results and Discussion	84
5.8	Conclusion	86
6	Multi-Input Interactions in VR Headsets	89
6.1	Introduction	89
6.2	Related Work	91
6.3	Mapping Task and Physical Interaction Design in Bimanual Multi-Input Interactions	92
6.3.1	Interaction Design M1: ArmX	93
6.3.2	Interaction Design M2: Double Bunny-Fingers	95
6.3.2.1	Experimental Set-Up and Design	97
6.3.3	Participants and Procedure	99
6.3.4	Results and Discussion	101
6.3.5	Summary	104
6.4	Multimodality in Hand Gestures: Exploring Physically Coupled And Decoupled Modalities	104
6.4.1	Introduction	104
6.4.2	Interaction Design M3: HeadX	106
6.4.3	Evaluation	107
6.4.3.1	User Task	107
6.4.3.2	Experimental Set-up and Design	109
6.4.3.3	Participants and Procedure	110
6.4.3.4	Results	111
6.4.4	Discussion	114
6.4.5	Summary	116
6.5	Applications for Compound Multi-Input Interactions	117

6.5.1	Gameplay Design	117
6.5.2	Participants and Procedure	118
6.5.3	Results and User Impressions	119
6.6	Conclusion	122
7	Conclusions and Future Work	125
7.1	Summary	125
7.2	General Insights and Conclusions	127
7.3	Limitations and Future Work	129
	List of Author's Awards, Patents, and Publications	133
	Bibliography	135

List of Figures

1.1	Headsets	1
1.2	Interaction Flow	2
1.3	Three components of interaction paradigm	7
2.1	Early VR headsets	10
2.2	Early HMDs	11
2.3	CAVE	11
2.4	VR HMD	12
2.5	Cardboard	13
2.6	Interaction in Cardboard	14
2.7	Google Daydream View	15
2.8	PlayStation VR with an external controller [32].	16
2.9	(a) The Oculus Quest headset. (b) Oculus Touch, its two handheld 6DOF controllers. [34].	17
3.1	CardboardSense: A finger tap-based interaction method for DIY cardboard headsets. Finger taps are tracked by smartphone motion sensors to provide discrete inputs into a VR application.	26
3.2	Left: Cardboard components. Right: potential tap locations.	27
3.3	GestureWrist: Wristband-type input device. Modified from [94].	28
3.4	TiltType prototype for text entry. (a) Labels indicate characters that can be entered by tilting in that direction. (b) TiltType is small enough to be worn on the wrist. (c) Both hands are required to enter text. Modified from [102].	29
3.5	In Stage 1, potential gestures were explored using this real-time plotting tool to see accelerometer (top) and gyroscope (bottom) signal.	31
3.6	Gyroscope and accelerometer signals for four different simultaneous left-and-right taps; we can see that the patterns are unstable.	32
3.7	Example signature samples (aligned) extracted from the raw signal (e.g., leftmost) using the methods presented in Stage 2. RGB plots refers to signals in XYZ, respectively, and the horizontal axis is time in seconds; note also that to show the temporal variations more clearly, the frontal part of the signatures is zoomed into, rather than showing all the 45 data samples in the signature samples. Each signature sample ranges approximately 0.9 sec.	33

3.8	Example non-gesture signature samples collected in different head-/body movement situations (B2-B6), note the time scale in the plots.	34
3.9	User subjective feedback of various aspects of their experience using CardboardSense.	39
3.10	Mean gesture accuracy for various tap locations as a result of the user evaluation, averaged for 16 participants.	39
3.11	(a) Horizontal button layouts for intuitive tap gesture selections. Handling more than 3 menu items using (b) translucent sliding selection window or (c) the enlarged icons technique employed on Apple iOS desktops.	42
3.12	An improved interface to extend the interactivity of panoramic video applications using Cardboard Sense gestures. (a) A specific tapping gesture (L/R) done will cause the corresponding slider bar to appear in order to provide visual feedback of the adjusted level. (b) Context-sensitive buttons like Exit can appear at the top when the video has stopped in order to communicate its activation based on the top-mid tap gesture.	42
3.13	An improved interface to extend the interactivity of first person shooter game using CardboardSense gestures.	43
4.1	DualGaze: A, eye gaze-based interaction method for headsets with embedded eye trackers such as FOVE[44]. This method addresses the Midas Touch problem in gaze-based interaction designs.	46
4.2	Two comparative interaction designs for selecting a target with one's gaze in a gaze-mediated VR user interface. The DualGaze method (left) and the Fixed Gaze method (right).	47
4.3	The stages of the DualGaze interaction method, with the red eye denoting the user's gaze point, involving a user's gaze travelling through buttons 1 and 2 to confirm the user's desired choice, button 3. The positioning of the confirmation flag does not obstruct the gaze trajectory and forces one to gaze back to confirm	52
4.4	(a)Accuracy of three different gaze lengths. (b) Selection Speed for the three lengths. Statistical significance is denoted by asterisks (**= $p < 0.01$)	54
4.5	Left: The physical setup, with the FOVE HMD used in this work. Right: A sample layout of the UI displayed on the FOVE HMD, where users have to gaze-select ten numbers on the keypad shown.	55
4.6	(a)Accuracy of both methods. (b) Selection Speed for both methods, with Standard Error bars. Statistical significance is denoted by asterisks (**= $p < 0.01$)	57
4.7	Eye gaze trajectory denoting travel time and selection time as gaze moves from one answer to the next.	58
4.8	Temporal performance changes in selection speed, with Standard Error bars.	59
4.9	Temporal performance changes in accuracy.	60

4.10	Survey responses to users' preferred selection method in use-case scenarios	62
5.1	Single-input finger-based interactions explored in this chapter, all using a smartphone's RGB camera.	65
5.2	The three proposed interaction techniques: (a) Technique S1 : Twin-Fingers (b) Technique S2 : Bunny-Fingers; (c) Technique S3 : X-Fingers.	66
5.3	(a)The FistPointer interaction method employed by Ishii et al [51] that relies on absolute hand position tracking. (b) The Twin-Fingers interaction method that relies on tracking the relative movement of negative space between both pairs of fingers.	68
5.4	(a)The three areas of negative space formed between four fingers that are used to determine cursor position; (b) Fingers of the dominant hand move down to up vertically over the other hand; (c) Progressions of length of negative space with time, (d) Movement of screen coordinates (the y-axis starts at 0 from the top of the screen), both during the movement in (b).	69
5.5	(a) Graph of changing areas of the left, right and center gaps as scales change. (b) Graph of stable ratios of left and right areas with respect to the center gap's area	72
5.6	Absolute XY Positions of the user's hands (C_x and C_y respectively, as seen from a moving viewpoint) with respect to the stability of the cursors(P_x and P_y); this demonstrates that Twin-Fingers is invariant to head motion in a VR headset.	73
5.7	Absolute XY Positions (C_x and C_y respectively) of the user's moving hands with respect to the stability of the cursors; this demonstrates that Twin-Fingers is invariant to hand motion.	74
5.8	The Bunny-Fingers interaction gesture, with the user's thumb and forefinger pinched together. The negative space between fingers, Zone A, is tracked by the smartphone camera, and the rightmost tip of the zone controls cursor position.	75
5.9	(a) The Twin-Fingers interaction, with the 3 gaps that are tracked (A_x, A_y and A_c). (b) The Bunny-Fingers interaction, a comparable alternative to Twin-Fingers that is variant to head motion, tracked by a single gap (Zone A)	76
5.10	Screens displayed for both tasks in the user study. (a) Numbered grid with direct selection for User Task 1; (b) Numbered grid surrounded with numbered bubbles to burst with slight head movements, for User Task 2; (c) A user performing a task using TF. . . .	76
5.11	The X-Fingers (XF) interaction design: Sliding the dominant index finger horizontally or vertically moves an onscreen 2D cursor.	80

5.12	Tracking X-Fingers (a): Image of user's crossed fingers as captured by camera; (b) Segmented fingers; (c) Convex hull of fingers in pink, detected fingertips (T_1 and T_2) and intersection point I ; (d) The varying lengths of L_x and L_y , used to calculate the x - and y -positions of the onscreen cursor.	81
5.13	Screens displayed for both tasks in the user study. (a) Numbered grid with direct selection for User Task 1; (b) Numbered grid surrounded with numbered bubbles to burst with slight head movements, for User Task 2; (c) A user performing a task using Technique S3.	83
5.14	: Mean task completion time in both interaction techniques, across 2 user task types, with Standard Error bars. Statistical significance is denoted by asterisks (*= $p < 0.05$).	85
6.1	Multi-input multimodal interactions explored in this chapter, all using a smartphone's RGB camera.	90
6.2	(a)ArmX: Bimanual non-identical interaction (b) Double Bunny Fingers (DBF): Bimanual identical interaction	93
6.3	ArmX: Bimanual non-identical spatial position control with finger (Input 1) and arm (Input 2) movement.	94
6.4	Double Bunny Fingers (DBF): Bimanual identical interaction gesture with two inputs.	96
6.5	Task Assignment Types. (a)Spatially asymmetric positioning task with a shaded circle nested inside a large shaded square. (b)Spatially symmetric positioning task with two disparate shaded shapes of the same size.	98
6.6	Coordination levels for each task assignment type, induced by dynamic outlines.	99
6.7	A participant performing the multi-input tasks using (a)ArmX interaction method, and (b)DBF interaction method.	100
6.8	Mean task completion time in both interaction techniques, across 2 user task types and 2 coordination levels, with Standard Error bars. Statistical significance is denoted by asterisks (*= $p < 0.05$; **= $p < 0.01$).	101
6.9	User preferences of technique for asymmetric and symmetric task types, from 11 participants.	103
6.10	(a)ArmX: Multimodal spatial position control with finger and arm movement (physically-coupled); (b) HeadX: Multimodal spatial position control with finger and head movement (physically-decoupled).	105
6.11	The three user tasks (L1, L2 and L3) with increasing levels of coordination between the two interaction modalities. Increasing coordination is achieved through the introduction of dynamic target outlines that the user needs to track during interaction.	107

6.12	(a) Participant performing the user study; (b) Sample interface of the application with the local cursor controlling the blue circle, and global cursor controlling the purple square. The boundary is denoted by the blue box; dimensions are C_h and C_w	109
6.13	Mean task completion time in both interaction techniques, across 3 different levels, with Standard Error bars. Statistical significance is denoted by asterisks (**= $p < 0.01$).	111
6.14	Subjective feedback from participants: User perceptions on various aspects of each technique on a 5-point Likert scale with Standard Error bars. Statistical significance is denoted by asterisks (**= $p < 0.01$)	113
6.15	Subjective feedback from participants: User preferences of technique for each level.	114
6.16	Sample screens from 'Box That Pizza!' (a) The purple user-controlled box tries to 'catch' the pizza. (b) Pizza toppings and cardboard box appears immediately once user-controlled box encloses the pizza . .	118
6.17	User task performance per level in 'Box that Pizza!', with Standard Error bars. Statistical significance is denoted by asterisks (*= $p < 0.05$)	119
6.18	Subjective user evaluation: User rating of playability, enjoyment and ease of use for both interaction methods , with Standard Error bars. Statistical significance is denoted by asterisks (*= $p < 0.05$; **= $p < 0.01$)	120
6.19	Mock-up of a VR paint tool with one 2D spatial input modality controlling the brush tool (C2; represented by the brush icon) and the other input modality simultaneously controlling the brush's color wheel (C1; represented by the white crosshair).	122
7.1	A list of major novel interaction methods developed in this research, along with their interaction task context in VR.	126
7.2	Continuous input on Cardboard-based headsets by sliding fingers over an affixed serrated strip.	130

List of Tables

3.1	Cross-validation rates of the remaining feasible gestures in % . . .	35
3.2	Gesture Recognition Accuracy (%): 10 new and 6 old participants. .	38

Symbols and Acronyms

Acronyms

VR	Virtual Reality
HMD	Head-Mounted Device
NUI	Natural User Interface
IPD	Inter-Pupillary Distance
USB	Universal Serial Bus
IR	Infrared
LED	Light Emitting Diode
CAVE	Cave Automatic Virtual Environment
NUI	Natural User Interface
TF	Twin-Fingers
XF	X-Fingers
BF	Bunny Fingers
DBF	Double Bunny Fingers
P-C	Physically-Coupled
P-D	Physically-Decoupled
M	Mean
SD	Standard Deviation

Chapter 1

Introduction

The onset of Virtual Reality (VR) has opened in its wake exciting potential for users and developers alike. The rapid rise in popularity of low-end VR headsets such as Google Cardboard has seen VR getting accessible to a wide range of users and applications - in fields as diverse as medicine, gaming and military [4].

Essentially, VR replicates an immersive environment that simulates a virtual world on a display for a user. The physical environment of the user is occluded, and they are presented with an immersive 3D virtual environment, which they can then navigate and control. VR in itself is a potential medium for users to engage with one another and share experiences [5].

FIGURE 1.1: (a) Examples of VR headsets. (b) A stereoscopic VR application screen. (c) A user wearing a VR headset

Immersive virtual environments can be provided to the user through several ways, and a widely-used VR display is the head-mounted display (HMD), which is a head-attached VR display that moves in tandem with the user's head. The most popular example of a head-worn display is a VR headset, with either an attached display or a slot for a smartphone to be embedded in it, as shown in Figure 1.1(a). Users wearing such a headset are presented with a stereoscopic screen on the display, as in Figure 1.1(b), which they can then interact with, as in Figure 1.1(c).

Interaction essentially is the process through which a human, or user, communicates with a device. In the context of VR, many VR applications require interactivity between the headset and the user. However, traditional PC or console interaction methods (such as handheld controllers, keyboard and mice), or smartphone-type direct touch interactions are not always practical as these physical peripherals are not visible or readily accessible to users wearing HMDs. This motivates the design of other novel interaction methods through which users can communicate with a VR headset without using external peripherals. The interaction paradigm in VR follows a basic workflow: A human input (eg. finger) is tracked by a sensing technology (eg. motion sensor), which forms a recognizable interaction gesture (eg. tapping gesture), that is used to communicate with applications in VR(eg. video player). This workflow is illustrated in Figure 1.2.

The research presented in this thesis explores and evaluates several novel ways in which users can employ different parts of their own body as human input or channels of interaction with the VR headset. It also addresses several challenges associated with interacting with the different input requirements of application scenarios in VR headsets.

FIGURE 1.2: The flow of interaction in VR: human inputs are tracked by sensing technology, forming interaction gestures that interact with VR applications

1.1 Motivation

An emerging trend in the field of VR is the rise of portable VR headsets. Essentially, VR headsets are Head-Mounted Devices (HMDs) that consist of either an inbuilt stereoscopic display or a slot for a smartphone to act as a display. Companies today have taken advantage of VR's popularity to develop and market new, portable, and lightweight headsets that offer VR experiences[6].

However, despite their ubiquity, VR headsets pose several environmental constraints on users in terms of headset build, by closing users off from the outside world and disabling direct contact with the smartphone or display in use. In effect, this disables further touch-based interaction with the smartphone, and thus without interaction, the VR headset is limited by the experiences it can offer the user. Especially in interaction-heavy domains such as gaming, this is a major setback as such kinds of applications require precise and accurate user controllability. Due to these unique challenges, VR designers are required to rethink new solutions for interacting in the VR domain.

While the VR headset's rise is linked with its accessibility, interaction techniques developed for it should ideally be contained within the VR headset's environment and not require external peripherals or controllers; this benefits the VR experience since the use of natural, embodied interaction methods can lead to a better VR experience and immersion than using external controllers such as a joystick or mouse [7][8]. However, a survey of existing literature indicates that while there has been ongoing research in this area, most of them depend on extraneous, attachable and expensive hardware such as external controllers, to achieve broadened interaction design.

Moreover, while single-input interactions are the norm in VR HMDs, there are also examples of VR applications that require two or more 2D inputs to be controlled by the user in performing multiple coordinated tasks, such as simultaneously manipulating and positioning objects [9]. Multimodal interaction in VR simultaneously employ the behaviour of various parts of the user's body such as the eye gaze, head rotation, hand movement and speech to interact with VR HMDs. It has also been found that users have a strong preference to interact multimodally rather than unimodally; this preference, however, is most pronounced in spatial application domains [10].

There has been research that has focused on novel, unconventional human-based input methods. A prominent discrete interaction modality available to VR users that uses human input elements is voice. Voice commands can be employed as a discrete interaction modality in VR; this requires no direct touch-based interaction onscreen and only a smartphone's microphone for detection. However, the use of voice as a discrete input modality does suffer from limitations, the primary one being a loss of privacy in interaction. Another disadvantage is a lack of accuracy and intuitive mapping of commands for spatial input controls that are typically required in such discrete input applications. Due to this nature of voice, this research explores modalities that involve gaze, hands, head and fingers - human gestural elements that provide input modalities that suffer less disadvantages than voice.

These prominent gaps in available interaction methods reveal potential for novel single and multi-input interaction techniques and design to be developed for VR headsets that are natural, accessible and usable. This is the primary motivation of this research. A brief survey of interaction methods available today in VR headsets are as follows:

- **External Trigger:** The first-generation Google Cardboard headset allows the user to slide an attached magnet on the left-hand side to trigger an event using the phone's magnetometer. Other headsets follow this by introducing an external push button that attempts to substitute major touch events onscreen, the obvious limitation being that it leaves users with a finite set of buttons to make a choice, depending on the particular hardware and build of the HMD or headset.
- **Head Tracking:** Head tracking is a relatively simple way to change the look-at direction of the current stereoscopic view in the virtual world. This method maps the user's head orientation sensed by the smartphone's built-in accelerometer to the look-at direction in the 3D virtual world. The earlier version of Cardboard SDK offered a gaze-and-trigger feature that triggers an event associated with a virtual object that the user is gazing at (sensed through head orientation) over a prolonged period or during the magnet button activation. This research explores the combination of head and hand gestures as an efficient possibility for multimodal interaction in VR headsets.

- **Eye Tracking:** Eye tracking in virtual reality is not completely novel, and primitive versions of it was implemented as far back as 2002[1], as a nascent possibility. However, with recent innovations and the rise of the HMD, eye tracking has taken on more significance than ever before. However, gaze-based interfaces in VR face several challenges such as the Midas Touch Problem, that remain problematic issues in mobile VR environments, and is further investigated in this thesis. [11].
- **Hand Gestures:** Hand gesture-based interaction methods are also on the rise. However, a majority of these gestures require the attachment of external depth cameras, and mostly offer single-input modalities. This research designs vision-based hand gesture interaction techniques, and also explores multimodal approaches to hand gesture interaction designs in mobile VR.

Given the limitations of current interaction methods in mobile VR, this research aims to widen the currently available range of input modalities by exploring the combinations of various affordances of human gestures, hardware sensors and interaction scenarios to develop novel interaction techniques for use with VR headsets.

1.2 Objectives

The objectives of the research presented in this thesis are to investigate and evaluate novel, effective and usable interaction methods for mobile virtual reality applications.

Interaction paradigms in VR headsets are essentially made of three major components:

1. **Human Gestural Input Element:** the interaction modality, or the particular way in which users utilize their body to communicate with the VR headset through the means of gestures and movement: e.g. eyes, fingers, hands, head.
2. **Sensing Technology:** the technology that recognizes and translates these human gestures to input for the mobile VR system.

3. **Application Input Design:** the type of input design required in applications, that respond to the interaction techniques by performing actions.

The research presented in this thesis considers the various affordances that different components that make up interaction paradigms provide. Firstly, it considers the various parts of a human body that can be used as modalities of interaction, such as eye gaze, head movement, finger gestures and hand gestures. Next, it considers the available sensing technology by which these human inputs can be tracked. As a general guiding principle of our research, as far as possible, the inclusions of external or extraneous hardware or controllers are kept to a minimum in this research. While tracking eye gaze requires IR cameras for precise and accurate tracking, finger taps can be detected on the periphery of headsets through vibrations detected via the embedded smartphone's motion sensors. Moreover, the designs of hand and finger gestures presented in this research have been such that they can be detected and used as input via the embedded smartphone's rear RGB camera, as this allows the interaction to be done without additional sensors such as a depth camera. An overall depiction of the various affordances of sensing technology, human gestural input and application input designs in VR is charted in Figure 1.3. The figure also outlines the chapters in the thesis that cover the interaction designs implemented using particular combinations of components.

The objectives of this research can be summarized as follows:

- Develop novel interaction techniques in VR headsets that employ human input channels tracked using available sensing technology for common interaction scenarios in VR.
- Investigate multimodal and multi-input approaches to interacting with VR headsets using aforementioned interaction designs.
- Evaluate the comparative performance of these interaction techniques with baseline methods through formal user evaluations and subjective feedback.

1.3 Major Contributions of the Thesis

The major contributions of the thesis are summarised as follows:

FIGURE 1.3: The three major components of the interaction paradigm in VR headsets, and the combination of their various affordances that this research investigates. Coloured lines represent the chapters in this thesis that address the particular combinations.

1. A novel set of tap-based gestures that can be detected on the periphery of a cardboard-based VR headset called CardboardSense has been proposed, developed and evaluated through user evaluations.
2. Eye gaze-based interfaces and their associated challenges have been investigated in the environment of a VR headset. A novel eye gaze-based interaction design called DualGaze that addresses these challenges has been designed, implemented and evaluated in a user study, and has been found to outperform an existing popular solution in a numeric keypad selection application scenario.
3. A novel set of vision-based single-input interaction methods using hand and finger gestures called Twin-Fingers and X-Fingers have been designed, developed and evaluated through a series of user studies and found to be more usable and effective when compared with a baseline method for numeric keypad selection applications.
4. Extending the single-input interaction methods developed, a set of bimanual multi-input interaction designs have been designed, developed and evaluated. A study of the mapping between the interaction task design and the physical

interaction design, and its effect on user performance in different types of interaction tasks are detailed.

5. A novel physically-decoupled multi-input interaction design involving the combination of head and fingers is designed, developed and evaluated together with a physically-coupled interaction design. The findings from this evaluation provide insights on the effect of physical coupling on user performance on tasks requiring coordination of multiple inputs.

1.4 Organization of the Thesis

The thesis is organized into 7 chapters, as follows:

- **Chapter 1** introduces the basic research problem and scope of this research, motivation, and objectives. A brief description of available interaction methods in VR headsets is also discussed.
- **Chapter 2** presents an in-depth survey of literature in the fields of VR interaction, gaze-based interaction, bimanual interaction as well as multimodal interaction.
- **Chapter 3** presents a novel technique for interacting with a DIY cardboard-based VR headset based on tap gestures. The stages of its development as well as experimental results obtained are discussed.
- **Chapter 4** explores eye gaze-based interaction in VR headsets, and presents a novel gaze-based interaction method to address particular challenges that such interfaces face.
- **Chapter 5** investigates single-input interaction using vision-based hand and finger gesture recognition in VR headsets.
- **Chapter 6** explores multi-input finger, hand and head gestures in VR headsets and presents a set of novel multi-input interaction techniques involving them.
- **Chapter 7** concludes the research by summarising the various interaction methods developed in this research for particular application task scenarios, and summarises the major findings of this research.

Chapter 2

Literature Review

2.1 Introduction

The research presented in this thesis focuses on designing novel interaction methods for VR headsets. A study of emerging research in VR would set the concepts presented in this thesis in context, and also outline the relevance and novelty of this research. Detailed literature surveys relevant to each of the interaction method proposed in this research are presented separately within the context of each chapter.

This chapter is organized as follows: it starts by shedding light on the history of VR systems, environments, displays and applications. Further, it discusses the emerging trends in VR displays, and the rise of smartphone-based VR headsets. Next, the history and kinds of interaction methods available in VR headsets today are discussed in detail, elucidating some of the challenges of designing interaction for VR, and the need for novel interaction methods as well. The chapter is concluded with a discussion of contemporary applications in VR headsets that require the design of novel interaction methods.

2.2 Virtual Reality - A Background

Virtual Reality (VR) existed as far back as in 1965, when Ivan Sutherland presented the first concept of it: a system to make a virtual world look, sound and feel real,

and respond realistically to the actions of the viewer [12]. However, it was Jaron Lanier, founder of VPL Research, who first coined the term VR [13]. VR has come a long way since then, and research over the past decades have taken Lanier and Sutherland's idea of an absolute Virtual Reality closer than ever. This section analyses the emergence of VR systems and environments. [14][13].

2.2.1 Early VR Systems

FIGURE 2.1: (a) Sensorama, the earliest Virtual Reality system [15]. (b) The Sword of Damocles, the earliest VR hardware and HMD [16].

- **Sensorama (1962)**

The first Virtual Reality system created was the Sensorama, in 1962, patented by Morton Heilig [15]. It was a multi-sensory system that paved the way for future VR systems. The Sensorama allowed users to enter in a manner similar to an interactive cinema[17].

- **The Ultimate Display (1965)**

Sutherland defined the concept of VR by proposing a system with interactive graphics, and sight, smell, sound and force feedback in a virtual, or artificial environment. He called such a system The Ultimate Display [12].

- **The Sword of Damocles (1968)**

Often known as the first VR hardware, Sutherland constructed a VR system that was also the first instance of a Head-Mounted Display (HMD), in 1968. The HMD supported head tracking and a stereo view that updated according to the user's head position as well as orientation [16]. Figure 2.1 shows the system.

- **DataGlove (1985) and the EyePhone HMD (1988) by VPL**

One of the earliest instances of an interactive VR system was the DataGlove, used as an input device, in tandem with the EyePhone, which was an immersive HMD. These were manufactured and created by a company called VPL [18]. The system is pictured in Figure 2.2.

FIGURE 2.2: The EyePhone HMD and DataGlove by VPL [18].

2.2.2 VR Environments

FIGURE 2.3: A CAVE environment with four projection walls or surfaces [19].

Immersive virtual environments can be provided to the user through several ways, and can be categorized into two basic types. The first, and the earliest kind is a

stationary display. Such a display is fixed in location, although the world being rendered on this display moves in relation with the user's bodily presence. An early example of such a system includes the popular CAVE (CAVE Automatic Virtual Environment) systems [19]. A CAVE system consists of four projection surfaces formed by projecting stereoscopic images onto the four walls of a room, as shown in Figure 2.3. One of these walls is a door. The CAVE system allows users to look around in all directions, thus allowing them to interact with full immersion in a virtual environment [20] [21].

The second kind is the head-mounted display (HMD), which is a display that is moving in tandem with the user's head, and is usually attached to the user's head [22]. As mentioned in the previous section, Sutherland's The Sword of Damocles was the earliest example of an HMD. While VR was hyped in the 1980s, it did not survive into the 1990s, with the rise of internet [23]. It was not until 2012 that VR regained favour in the public consciousness, when Oculus released the Oculus Rift HMD, heralding a new wave of advancements in VR technology and research [24].

Thus, today, the most popular example of a head-worn display is a typical VR headset, with either an attached display or a slot for a smartphone to be embedded in it, as shown in Figure 2.4.

FIGURE 2.4: An example of an HMD, a typical VR headset.

2.3 VR Headsets

A literature survey of the current developments in the field of VR interactions alone would be incomplete without an analysis of the VR headsets available in the market,

since most interaction methods in VR today are tightly coupled with the headsets that offer them. Technology giants such as Samsung [25] [26], HTC [27][28], Google [29] [30] and Facebook (Oculus)[24] have released their own headsets. With the exception of Google, however, most of these headsets employ headset-specific technology or hardware for interacting with them. For example, Samsung Gear VR includes a touchpad and back button on the side of its headset which allow users to perform basic selection tasks onscreen[25], while Oculus Rift has a positional tracking system, called "Constellation", performed by a USB stationary infrared sensor that is picking up light that is emitted by IR LEDs, in its headset.

This section attempts to compare and detail interaction methods available in a few of the popular headsets of today.

2.3.1 Google Cardboard

FIGURE 2.5: Google Cardboard. Left: Pre-assembly Right: Post-assembly [30]

Google Cardboard [30] is an innovative and affordable VR headset designed by Google Inc., where commodity users can experience VR simply by using their smartphones, as shown in Figure 2.5. The cardboard itself costs less than five dollars, and it can be easily assembled by folding the prefabricated Cardboard components and fixing a smartphone inside. By then, we can mount the Cardboard in front of our eyes with the help of a rubber band and experience an immersive VR world with stereoscopic viewing.

Google Cardboard was first released in June 2014 at the Google I/O developers conference. After just a few months in October 2014, more than half a million Cardboard units have been sold. The growing popularity of Cardboard has given rise to several third-party VR applications available to use on smartphones, e.g.,

Titans of Space, where users can explore a 3D simulation of our Solar system, Roller Coaster VR, which simulates the experience of riding a roller coaster, etc. Moreover, Google has released several SDKs to aid the development of Cardboard applications on Android and iOS, as well as providing a plugin for the Unity game engine.

However, there are very limited input methods for Cardboard. The most obvious one is to employ the built-in motion sensor to detect user's head orientation and then apply the tilt information to control the look-at direction in the virtual world. Other than this, the first-generation Cardboard provided a magnet-based control, where the user can slide a magnet on the left side of the Cardboard, upon which the phone's magnetic sensor can detect the changes in the magnetic field and trigger it as an event. The second-generation Cardboard in 2015 unveiled a small lever button, by which the user can make an indirect contact on the phone's display and generate a touch event. Although these two mechanisms offer certain degree of input capability, they are mainly used like an on-and-off button on the Cardboard for starting and stopping applications, as practised in existing Cardboard applications. Other than the above mechanisms, the Cardboard SDK offers a gaze-and-trigger solution, which uses the player's gaze (head orientation) to select or activate objects in the 3D virtual world. For example, after the player focuses on a certain object in the virtual world over a certain period of time, or he/she slides the magnet button during the gaze, the Cardboard API can cast a ray in the 3D

FIGURE 2.6: Examples of interaction methods in Cardboard: (a) Magnet sliding switch, (b) Head gaze toggle, (c) Head gaze menu, (d) Environment interaction using head gaze [31]

virtual world to find the associated virtual object in the player's view and trigger an associated event. Although this gaze-based method offers extra input capability, the overall capability for us to interact with Cardboard-based applications is still rather limited.

The work by Yoo and Parker shown in Figure 2.6 presented an initial analysis of the existing interaction methods with the Cardboard [31]. No new interaction methods were presented nor suggested in their work. Chapter 3 details the proposal and evaluation of a new interaction method designed specifically for this kind of DIY cardboard-based VR headset.

2.3.2 Google Daydream

FIGURE 2.7: Google Daydream View with a wireless controller [29].

Daydream was Google's next, more sophisticated venture into the VR arena. Daydream is essentially a VR platform, while Daydream View is the name of the VR headset that runs on the Daydream platform, shown in Figure 2.7 [29].

Announced on October 2, 2016, Daydream-ready smartphones can be placed in the front compartment of the Daydream View and then viewed in VR through the headset's two lenses. The major differences from previous headset builds include its lightweight exterior, capacitive nubs and an NFC chip to aid the set-up process.

The major interaction method introduced by Daydream View is a wireless controller, as pictured in Figure 2.7. This controller can be used for interacting with the virtual world through button presses or through waving the device. On-board sensors are used to track the orientation of the controller and approximate the position of the user's hand. The controller has a touch pad, two circular buttons

(one functioning as a home button and one functioning as an app-specific button), and two volume buttons, along with a status light. The Daydream Controller essentially works like a lightweight TV remote, and almost every interaction with the screen inside the headset depends on this controller.

2.3.3 Playstation VR

FIGURE 2.8: PlayStation VR with an external controller [32].

PlayStation VR shown in Figure 2.8 is developed by Sony Interactive Entertainment, is its most sophisticated yet, and is the biggest player in the VR gaming field [32].

The PlayStation VR has a 5.7-inch OLED panel, with a display resolution of 1080p. The headset also has a processor box which enables the Social Screen video output to the television, as well as nine positional LEDs on its surface for the PlayStation Camera to track 360 degree head movement [33]

Interaction with the PlayStation VR is, once again, through an external controller, as pictured in Figure 2.8. This controller packs more interaction options than Daydream View's minimalist controller, however - with five buttons and a motion tracker as well. Unfortunately, this headset also falls back to dependence on external interactive controllers. Moreover, it has a built-in screen, and the content is limited to PlayStation titles.

2.3.4 Oculus Quest

FIGURE 2.9: (a) The Oculus Quest headset. (b) Oculus Touch, its two handheld 6DOF controllers. [34].

Oculus released its latest headset, Oculus Quest in May 2019 [34], pictured in Figure 2.9(a). Oculus Quest is a completely standalone VR headset, which means it is not tethered to a PC through wires and is completely mobile.

The system runs on Snapdragon 835. The display uses two diamond Pentile OLED displays, and also allows for physical IPD adjustment. For spatial headset tracking, Oculus Quest uses an inside-out tracking system that was previously used in Rift, called Oculus Insight. The system consists of four wide-angle cameras placed at each corner of the headset for accurate spatial tracking.

Oculus Quest consists of two handheld 6DOF controllers (called Oculus Touch), much like its predecessor Oculus Rift S[35], shown in Figure 2.9(b).

2.4 Human Input-Based Interaction in VR Headsets

VR headsets require interaction methods for users to engage with VR content on their headsets effectively. As discussed in Chapter 1, interaction methods that employ the user's various body parts as input elements are required. This section discusses existing literature on human input-based interaction methods that have been developed. Some of these interactions require or work only in conjunction with external controllers.

2.4.1 Head Gaze-based Interactions

Since all VR headsets support head tracking, interaction methods that rely on head movement alone have been developed and deployed, such as the work by Yu et al. which explores the possibility of text entry by head movement in VR headsets [36]. The work by Atienza presents a detailed study on head gaze-based interaction in VR and some commonly encountered issues[37]. The work by Sheng et al. explores a novel menu interaction method in VR using Samsung Gear VR headset; interaction methods for menu selection are head gaze and Samsung Gear's touchpad controller [38]. Foxlin and Harrington have built a system called WearTrack, that combines sourceless head orientation tracker with a tracking device that is worn on the user's head that tracks a 3D beacon on the user's hand [39]. The work presented in Chapter 5 and Chapter 6 present interaction methods that combine head gaze with hand and finger movement for spatial object positioning in VR.

2.4.2 Eye Gaze-based Interactions

The concept of eye tracking in virtual reality is not altogether new. Tanriverdi et al. explored this concept in 2000 when they worked with an ISCAN eye tracker to compare the advantages of eye tracking as an interaction method in VR with respect to arm-extension grasping [40]. More recent studies have shown a marked progression in the field, such as the work of Geiselhart et al. in 2016 [41]. In this work, they present a reasonably cheap VR eye tracker using an off-the-shelf Raspberry Pi-based IR camera coupled with a Samsung Gear VR headset - initial tests do show feasibility.

Orlosky et al. have built a hybrid AR-VR system using Oculus Rift with extraneous hardware attachments [42]. They have used multiple IR cameras for this system. In addition, they have used 6 IR embedded LEDs around each of the lenses for the cameras to illuminate the eye. For eye tracking, they have used an available API, the SMI eye tracking API, to track the eye, as opposed to devising a novel approach. They devised their own algorithm to detect blinks and squints of the eye, rather than tracking. A significant disadvantage would be that they have used LEGO blocks to affix the various cameras around the system, which weigh the headset down considerably. They also have a separate pair of telescopic lenses to facilitate an AR experience as well.

Eye gaze tracking is gaining traction as an input modality in VR headsets. A recent study by Luro et al.[43] demonstrates the advantage of gaze based pointing over pointing through external controllers. Of late, the FOVE headset, a VR headset that supports eye tracking through an attachable IR camera, has become commercially available and retails for about USD599 [44]. Chapter 4 explores some challenges faced by eye gaze-based interfaces in VR headsets, and presents a novel interaction design that addresses these issues.

2.4.3 Hand and Finger-based Interactions

There have been several works which present mid-air pointing methods such as the works of Cockburn et. al[45] and Vogel et. al [46]. Such hand gesture interactions have also long been viewed as a natural way to interact with VR headsets. However, most of these methods require the use of external controllers [47][48][49]. A recent interaction method, TORC, is developed by Lee et al[50]. It is a haptic VR controller for high-dexterity finger interaction. It employs the use of an attachable external trackpad, which users slide their thumb over to interact with virtual objects.

A notable exception among these works is FistPointer, which employs hand gestures to navigate in a virtual environment without requiring any additional controllers for VR headset users [51]. This technique allows the user to select a target by means of a mid-air interaction through the smartphone's rear camera. Fistpointer is designed to support a single 2D spatial input control by the movement of the user's fisted thumb. More hand and finger-based interactions in literature are discussed in Chapter 5 and Chapter 6, along with the particular challenges associated with them. These chapters also present novel vision-based hand and finger interaction methods for VR headsets that address some of these issues, in particular the possibility of hand and finger-interactions that are capable of supporting two instead of just one 2D spatial input control.

2.4.4 Other Human Input-based Interactions

Apart from head, eye and hand-based gestures, existing literature also presents interaction methods that use more unconventional human-input gestures to enhance

the interactivity of the VR headset. However, these are somewhat limited. For example, there have been interaction methods that allow touch-based interaction on the face of the headsets, as in the works of Gugenheimer[52] [53], and in the works of Lee et al. [54][55]. A Hand-to-Face interaction method for HMDs has also been proposed by Serranno et al[56]. Cheek gestures have also been used as interaction methods in VR, as in the work of Yamashita et al [57]; so have other facial gestures [58][59]. Foot gestures have been employed as an input modality in the work by Saunders et al.[60], as well as Lopes et al.[61], detected by attached sensors. The preliminary work done by Wu et al[62] prototypes an interaction method that recognizes distinct eye expressions as input for VR devices. However, its feasibility remains unknown. Recently, Surale et al.[63] proposed finger touch-based gestures on a multi-touch tablet to interact in VR. These are all interaction methods that employ the use of external devices and peripherals. Given the prevalence of VR headsets that have embedded smartphones, the survey of existing literature shows a dearth of interaction designs that minimize the use of external controllers and focus solely on exploiting only the available sensing technologies of the smartphone. A notable exception is Fistpointer, as discussed before.

2.5 Interacting with VR Applications

Interaction methods are often designed around applications that necessitate them. From a survey of literature, one can see that VR headsets have applications that require certain kinds of input repeatedly, such as text/keypad entry, and menu selection. In fact, applications in mobile VR have been compared with classical WIMP (Window, Icon, Menu and Pointers) interaction methods and systems [64]. While WIMP interactions are static, VR environments provide continuously changing options for input, such as scene navigation, etc.

Input methods can generally be classified as discrete inputs or continuous inputs. However, some interaction scenarios require inputs that are designed around certain interaction tasks. Poupyrev et al [65] identified a set of basic interaction tasks that can be combined to form any complex manipulation actions. These include positioning, orientation, selection, text input and numerical input. According to Poupyrev, scaling is a task that can be achieved with a combination of selection, positioning and orientation [23].

This section further classifies the interactions described by Poupyrev into three kinds:

1. Text and Numerical Input
2. Object Selection
3. Object Manipulation (including positioning, orientation and scaling)

These interactions have been surveyed in the context of VR. Major works that propose interaction methods for such tasks have been presented as follows.

2.5.1 Text and Numerical Input

Text and number input in VR via keypad and keyboard entry have often been researched in VR, but most of the research has centred around the use of physical keyboards or external devices used in conjunction with VR HMDs in virtual environments. For example, the works done in ([66],[67],[68],[69],[70],[71]) all use connected physical keyboards for text entry in VR.

An early work by Rosenberg and Slater proposes a chording glove as a text input device, wherein a chord keyboard is mounted on a glove's fingers [72]. The study by Speicher et al evaluates various interaction methods for text input, all using external controllers and devices [73]. PizzaText is another text entry system in VR that uses dual thumbsticks of a handheld game controller to input text [74]. DigiTouch is a thumb-to-finger text input interaction method for HMDs, developed by Whitmire et al [48], that uses a touch-sensitive glove as input. However, all of these interaction methods to type in VR involve external handheld controllers and devices.

There have also been methods for text entry using head gaze, such as in the works of Yu, Gu, Yang et al [36]. Eye gaze is also explored as an input technique for number and text entry on a keypad. This has been explored in the works of Speicher et al, Hansen et al, Koesling et al, and Apel et al [73] [75] [76] [77]. A system by Ma, Yao, Wang et al combines Brain-Computer Interface and eye gaze for text entry in VR [78]. However, most designs fail to address certain usability challenges associated with gaze-based selection in VR, such as the Midas Touch Problem.

2.5.2 Object Selection

Object or target selection has repeatedly been identified as a primary task for interactions in VR [79] [80]. Selection of an object is the action of indicating a target for subsequent processing such as manipulation and text entry. In VR HMDs that present virtual objects or buttons, users are required to interact with such objects by navigating to them and selecting them for further action. Such tasks require object pointing and selection methods in VR. Ray casting is the earliest and most popular technique for object pointing; however, it suffers from several limitations such as low accuracy [81] [82][80][83].

Since head gaze-based interaction is commonly embedded within VR headsets, there have been works that explore navigation and target selection using head gaze [84]. Some works also combine eye gaze-based selections with head-gaze, such as in the work of Qian and Teather [85].

Menu selection is a common task in VR that requires object selection methods. Various menu selection methods using external controllers have been surveyed in a work by Kim et al [86]. Another menu selection method has been proposed for smartphone-based VR headsets by Sheng et al [38]. It combines eye and head gaze tracking with a Samsung touchpad. A recent work proposes a hand-based gestural selection method in VR called FistPointer which employs hand gestures to navigate in a virtual environment without requiring any additional controllers for VR headset users. This approach supports only one-degree motion - the movement of the hand alone[51]. The task of target selection also faces a few challenges, one of which is the Midas Touch problem. The Midas Touch Problem happens when an object is selected accidentally in the course of navigating through it. This has been investigated in the context of eye gaze-based interaction in the work explored in Chapter 4.

2.5.3 Object Manipulation

Object manipulation includes object grasping, spatial positioning and manipulating of other parameters of objects such as size, rotation etc. Typically, input methods for such kinds of applications require continuous input control. The dimensions of continuous parametric control will depend on the nature of the manipulative tasks.

For example, re-sizing a stationary rectangle would require a single continuous 2D-input control of the x and y dimensions, much like a typical mouse control. Whereas, the task of simultaneous re-sizing and positioning a rectangle on a 2D canvas will require dual continuous 2D-input control. This research explores both these continuous input requirements.

There have been several works that propose interaction methods for object manipulation based on handheld external controllers or tracked gloves. A work by Davis et al explores freehand object manipulation; however, it requires the attachment of an external LeapMotion controller on the user's forehead via a headband [87]. Tangible object manipulation in VR has also been explored by Wu et al [88]. Mapes and Moshell [9] have introduced a novel input device called ChordGloves, which emulates a pair of mice and keyboard, to translate, rotate, scale and align objects in VR environments. TurkDeck [89] is also a VR system that is prop-based, i.e., users manipulate objects in VR by manipulating a corresponding tangible object in the physical world. All of these interaction methods have necessitated the use of handheld controllers, tangible objects or gloves as opposed to natural human inputs or gestures in VR.

However, a study by Dias and Eliseu has found that users perform object manipulation tasks in VR better when the interaction is gesture-based, rather than controller-based [90]. This crucial finding has motivated the research's guiding principle of controller-less interactions that overcome limitations posed by other existing controller-less methods. There have been attempts to design such gestures for object manipulation in VR [91]. M. Achibet has presented a system for multi-finger interaction in VR environments; this involves users controlling a virtual hand by gesturing over a tactile tablet [92]. A study by Lin et al.[8] evaluates parameters to consider while designing hand-based gestures for object manipulation in VR environments; these have been studied using LeapMotion sensors to detect hand poses. The study was limited to a specific dice positioning task across a very short distance but demonstrated the usefulness of hand gestures as effective means for object manipulation in VR.

2.6 Summary

In summary, this chapter has discussed the history of VR systems and environments, describing the re-emergence of VR and the VR headset into public consciousness. It has outlined details of popular VR headsets available for users today, and the kinds of interaction or input devices that they entail, describing the need for more novel ways to interact with these VR headset, especially in ways that can avoid the use of external controllers and exploit existing sensing technologies already available in the embedded smartphone commonly found in many low cost VR headsets. Also discussed were some recent interaction methods developed for the VR headset platform and types input methods required by different VR applications.

The following chapters describe the various contributions of the thesis by presenting and evaluating novel interaction designs and methods for VR headsets.

Chapter 3

Tapping Interactions in VR Headsets

3.1 Introduction

The first investigation in this research explores a way to expand the interactivity of a simple DIY Cardboard headset (such as Google Cardboard) through human gestural input. Finger-based gestures are common and familiar forms of human gestures for user input in a natural user interface system[93]. Hence, a novel finger gesture-based interaction method for cardboard-based VR headsets named CardboardSense was designed, that provides a set of discrete inputs to the headset (see Figure 3.1). CardboardSense is a set of finger tap-based gestures that are detected within the embedded smartphone’s gyroscopic signals¹.

DIY cardboard-based VR headsets such as the Google Cardboard are growing in popularity as they can bring immersive VR experience to smartphone users for less than \$10. They have also spawned many third-party VR applications, e.g., Titans of Space, where users can explore a 3D simulation of Solar system, and Roller Coaster VR, which simulates the experience of riding in a roller coaster, etc. Unfortunately, the Cardboard has limited input capabilities. An obvious input control is to map the user’s head orientation sensed by the smartphone’s built-in accelerometer to the look-at direction in the 3D virtual world. In addition, the first-generation Cardboard allows the user to slide an attached magnet on the

¹The research presented in this chapter has been published in [1]

FIGURE 3.1: CardboardSense: A finger tap-based interaction method for DIY cardboard headsets. Finger taps are tracked by smartphone motion sensors to provide discrete inputs into a VR application.

left-hand side to trigger an event using the phone’s magnetometer. A small lever button that makes indirect contact with the phone’s touchscreen was featured in the second generation Cardboard. Additionally, the Cardboard SDK offers a gaze-and-trigger feature that triggers an event associated with a virtual object that the user is gazing at over a prolonged period or during the magnet button activation.

This chapter explores finger tapping as an input modality for interacting with the Cardboard. It requires no additional physical implement since it takes advantage of the distinctive vibration patterns detected by the phone’s built-in motion sensors when a user taps on different locations on the Cardboard. As such, the proposed technique can also be applicable to other VR headsets provided the enclosure is rigid to support the signal transmission, e.g., other cardboard HMD like Microsoft VR Kit or better quality plastic versions like Samsung Gear VR. By analyzing and differentiating these different temporal signatures in real-time, Cardboard applications can now employ tapping gestures as an additional form of user input, as shown in Figure 3.2. A summary of this chapter is given in a short video at www.dropbox.com/s/kk84h7f0o4hhn5c/paper230.mp4?dl=0

This technique was developed by first exploring all tapping gestures that can be detected distinctively by the motion sensors. The investigation started by deriving methods to extract temporally-aligned signature samples from the continuous signal and progressively analyze the similarity among different gestures through

FIGURE 3.2: Left: Cardboard components. Right: potential tap locations.

a machine learning method. After determining a set of feasible gestures, further experiments were performed to evaluate the detection performance and reliability of the gestures.

This chapter makes several important contributions. First, this interaction method has extended the input capabilities of the low-cost cardboard VR headset without requiring any physical changes to the device. It uses the phone's built-in gyroscope and is computationally efficient. Second, six different tapping gestures can be reliably distinguished. Their spatial distribution over the Cardboard surfaces provides a good degree of input options for interface design. Lastly, the tapping gestures are easy-to-learn and are ergonomically-friendly to perform. The chapter is organized as follows:

- Section 3.2 starts with a survey of related literature in the context of this work.
- Section 3.3 introduces the finger tap-based interaction method CardboardSense and discusses and evaluates its design in various stages.
- Section 3.4 details a user evaluation done to validate the interaction method.
- Section 3.5 presents the quantitative and subjective results of this user evaluation.
- Section 3.6 presents basic interaction design guidelines for integrating CardboardSense as an interaction method in some typical VR applications.
- Section 3.7 finally summarises the findings from the previous sections and concludes the chapter.

3.2 Related Work

The section discusses some of the interaction methods which are based on motion sensors.

Several pioneering works have been developed to explore different applications of motion sensors. Rekimoto [94] developed GestureWrist [Figure 3.3] and GesturePad, which use accelerometer and capacitive sensors to detect hand gestures and forearm movements. Lee and Mase [95] measured acceleration and angular velocity using wearable sensors to estimate the locations of users, their transitions in indoor space, and their activities, e.g., sitting, standing, and walking. Wilson and Shafer[96] developed a wand-based wireless sensor package, which employs accelerometer and magnetometer to detect user gestures. Kela et al.[97] studied accelerometer-based controls as a supplementary/alternative interaction paradigm in a design environment, while Iso and Yamazaki[98] mounted a single three-axis accelerometer on a normal cell phone and developed a method to analyse and classify gaits. While these works have developed methods to capture human motion or gestures, none have explored the capability of an embedded smartphone in a cardboard-like VR headset to expand their input capabilities.

FIGURE 3.3: GestureWrist: Wristband-type input device. Modified from [94].

There have been numerous works investigating the use of motion sensor information to support and enhance interaction applications.

Sensing Spatial Movement: Rahman et al.[99] fused the IR camera and accelerometer information from a phone to estimate the 3D position and orientation of the phone for pointing annotated objects in an indoor environment. Kratz et al.[100] analyzed the effect of combining accelerometer and gyroscope data on recognizing motion gesture with dimensionality constraints. Chen et al.[101] explored

joint interactions with smart phone and smartwatch, and developed user gestures that consider synchronized accelerometer data. CardboardSense, by contrast senses spatial movement around the periphery of the cardboard headset using machine learning.

Text Entry: Discrete text input is an important requirement in many applications. Several researchers have attempted to perform this task using motion-based sensing. Partridge et al.[102] employed tilt direction and angle sensed by accelerometer together with button pressed by user to facilitate text entry on small-sized mobile devices, shown in Figure 3.4. Jones et al.[103] studied the influence of human ergonomics, accelerometer and interface considerations to the design of accelerometer-based text-entry system, while Goel et al.[104] proposed an adaptive system that employs accelerometer information to increase the typing accuracy during walking. Text entry is a spatially sensitive task. On a typical QWERTY keyboard, a sensitive input method that is motion-sensing based could lead to inaccurate or inconvenient typing.

FIGURE 3.4: TiltType prototype for text entry. (a) Labels indicate characters that can be entered by tilting in that direction. (b) TiltType is small enough to be worn on the wrist. (c) Both hands are required to enter text. Modified from [102].

Distinguishing between Users: Iwasaki et al.[105] sensed typing pressure of users on keyboard by using the internal accelerometer in laptop computers, and distinguished users based on their typing patterns. Schmidt et al.[106] detected phone tapping on an interactive surface by an external IR camera and then identified the phone related to an event by checking the incidental acceleration sensed by

the built-in accelerometers in the phones. Hutama et al.[107] distinguished multiple phone contacts on a large multitouch wall display by correlating the temporal variation of the accelerometer data from the phones and the touch action on the display.

Expressive Finger Contacts: Hinckley and Song[108] distinguished a gentle finger swipe from a hard-onset drag on the phone screen by using gyroscope and accelerometer data, while Heo and Lee[109] employed accelerometer data to discriminate a gentle tap (Tap) from a strong tap (ForceTap). Goel et al.[110] combined information from multiple built-in sensors (touchscreen, inertial sensors, and vibration motor) to estimate hand posture and exerted pressure on phone. Serrano[111] proposed the Bezel-Tap gestures for immediate interaction on hand-held tablets; these gestures consist of two successive events: a bezel tap detected by the accelerometer followed a normal screen touch or slide. Zhang et al.[112] employed the microphone, gyroscope, and accelerometer sensors to detect tapping on the back side of a phone. Recently, Oakley et al.[113] explored beats tapping gesture for interacting with smart watches; these are rapid sequences of two-finger touches and releases on the smart watch screen. Rather than using motion sensor, Harrison et al.[114] explored the use of acoustic sensing to detect different forms of taps and touches: pad, tip, knuckle, nail, etc.

Although the above works explore and analyze motion sensing data for detecting different forms of finger actions, the scenarios that they target for are very different from the method discussed in this chapter. None of them has explored finger interaction with setups similar to that of the Cardboard, i.e., head-mounted and phone-based, for VR applications. The following section describes the proposed interaction method in detail.

3.3 Interaction Design

This work involves four research stages. First, various finger gestures were tested to see if they can be visually differentiated using the built-in motion sensors (stage 1). Then, signature samples of different gestures were extracted and analyzed (stage 2), and classifiers were constructed from these samples by machine learning to empirically determine a set of feasible gestures (stage 3), namely: tap left,

tap right, tap top-mid, tap bottom-mid, left double-tap, and right double-tap. A real-time method to recognize the gestures was then developed (stage 4). Lastly, various experiments were also performed to evaluate the results and potential usage scenarios were discussed.

FIGURE 3.5: In Stage 1, potential gestures were explored using this real-time plotting tool to see accelerometer (top) and gyroscope (bottom) signal.

Stage 1: Exploring Potential Gestures

In Stage 1, potential gestures that could be performed were identified, e.g., tapping on different locations (see Figure 3.2), tapping left and right simultaneously, double tapping, sliding on Cardboard, etc. To aid the exploration, a program that continuously plots the motion signals on the phone was developed to see the signal response in real-time as different finger actions were tried out on Cardboard. This has been depicted in Figure 3.5.

From this initial experiment, the simultaneous left-and-right taps were dropped, since the signal patterns are quite unstable, see Figure 3.6. Sometimes, they looked like stationary signals since left and right taps cancel each other out, and sometimes, they looked like a left or right tap. Moreover, since the cardboard surface is rather smooth and hard, the signal responses for sliding were found to be weak, or similar to that of tapping. Hence, the sliding gesture was also excluded.

Stage 2: Collecting Gesture Data

Next, accelerometer and gyroscope signature samples of the following gestures were collected: tap left, right, front-L, front-M, front-R, top-L, top-M, and top-R,

FIGURE 3.6: Gyroscope and accelerometer signals for four different simultaneous left-and-right taps; we can see that the patterns are unstable.

bottom-L, bottom-M, and bottom-R, and double-tap versions of them. Note that for consistency, a bottom tap is considered to be one performed by a finger flick from below. As such, no bottom double-tap is considered since double finger flicks might be difficult for some people to perform.

To collect signature samples of a gesture, the same gesture action was repeated successively twenty times every two seconds, and a long sequence of accelerometer and gyroscope data was recorded. Here, it was found that if the long signal were to be simply chopped into segments centered at the designated action moments (every two seconds), the extracted signature samples may not align well with one another, see Figure 3.7(left-most), because the actions may not be performed exactly at the designated moments. Hence, the first local (absolute) maximum among the signal channels was searched for, within $[-0.2, +0.8]$ seconds around each designated moment, and 10 data samples before the local-maxima sample and 34 samples after it were extracted to form a signature sample of 45 data samples.

For each gesture, the above process was repeated 16 times, thus producing 20×16 -signature samples. To further improve the method's robustness, 6 additional participants (2 male and 4 female; aged 23 to 30) were recruited to perform the above process twice, thus helping to produce an additional 20×2 -signature samples per gesture. Figure 3.7 revealed that different gestures produce different signature samples.

Gyroscope vs accelerometer: From Figure 3.7, we can see that both gyroscope and accelerometer signals reveal certain temporal variation while the accelerometer signals are usually relatively weaker. In addition, accelerometer is more sensitive to the user's static head orientation, which is not advantageous to the data training. Results using gyroscope only (G) are also compared with results that use gyroscope together with accelerometer (A+G). It was found that A+G can improve the accuracy slightly at the expense of an increase in computation - the accuracy was found

FIGURE 3.7: Example signature samples (aligned) extracted from the raw signal (e.g., leftmost) using the methods presented in Stage 2. RGB plots refers to signals in XYZ, respectively, and the horizontal axis is time in seconds; note also that to show the temporal variations more clearly, the frontal part of the signatures is zoomed into, rather than showing all the 45 data samples in the signature samples. Each signature sample ranges approximately 0.9 sec.

to be improved from 99.51% to 99.56%, but the overall gesture detection time can increase from 0.04 (G) to 0.11 (A+G) sec. Computational cost increases by 2.75 times. The above reasons led to the final choice of employing only the gyroscope's signals in gesture training and recognition.

Non-gesture data: Next, to help differentiate user gestures from head/body motion, motion samples of six different situations (B1-B6) were collected, see Figure 3.8 for B2-B6, while B1 is stationary. For the case of B6 (incidental and small head movement), 80 signature samples were collected by the procedure used for tapping, while for the rest, the actions were performed over 80 seconds, and randomly picked and collected 80 signature samples from the long gyroscope signal.

Stage 3: Training Gesture Data

FIGURE 3.8: Example non-gesture signature samples collected in different head-/body movement situations (B2-B6), note the time scale in the plots.

This stage has two purposes: i) analyze the distinctiveness of each gesture (against other gestures as well as B1-B6), and ii) build classifiers for gesture recognition. Here, a machine learning library called LIBSVM and the C-SVC model with radial basis function as the kernel were used[115]. To further optimize the classifier, hyperparameter optimization by the grid search method with 5-fold cross-validation was performed.

First, the cross-validation rates of the three top tap gestures were found to be rather low: tap top-L (81.62%), tap top-M (86.14%), and tap top-R (73.22%), so it is not feasible to differentiate among them. Figure 3.7 also shows that they exhibit very similar patterns. Since tap top-M is slightly more distinctive than others, tap top-L and tap top-R were dropped, and tap top-M was kept. Similarly, the three bottom tap gestures are very similar, and thus only tap bottom-M was retained.

Second, it was found that tap front-L and tap front-R are similar to tap right and tap left, respectively. The cross-validation rate to differentiate tap front-L from tap right is only 75.11% while it is 83.92% for tap front-R from tap left. From the signal plots, it was found that tap front-L (tap front-R) produces incidental motion similar to that of tap right (tap left), so tap front-L and tap front-R had

Device	Single Tap				Double Tap	
	Left	Right	Top-M	Bottom-M	Left	Right
iPod touch	99.76	99.86	99.23	99.19	99.58	99.43
Android	99.26	99.55	99.09	98.96	99.03	99.12

TABLE 3.1: Cross-validation rates of the remaining feasible gestures in %.

to be dropped. Furthermore, the cross-validation rate for differentiating front-M from bottom-M was found to be just 62.75%. Here, tap front-M was ignored and tap bottom-M kept, since front-M was found to be less stable: with the nose as the pivot point, its signature pattern would vary with slight changes on the front-tap location.

Lastly, since Cardboard is mounted on the left and right sides of user's head by a rubber band, double top tap would produce relatively larger vibration compared to others, so its accuracy is relatively low. Thus, double top tap was also dropped.

Table 3.1 shows the cross-validation rates of the remaining feasible gestures. The rates are all very close to 100%. The slight imperfections were mainly due to weak outlier signals resulted from gentle/misplaced tap actions during Stage 2.

Stage 4: Gesture Recognition After training the SVM classifiers, they were employed to recognize gestures in real-time on mobile devices in Cardboard. To improve the recognition efficiency and accuracy, instead of testing every 45-sample time frame in the signal stream, the method and results in Stage 2 were adopted. That is, a local maximum in the signal stream is firstly searched for and studied if the value is near to the local maximum values of the signature samples extracted in Stage 2. If so, a 45-samples time frame is extracted with 10 samples before the local-maximum point and 34 samples after. Then, the SVM classifiers constructed in Stage 3 are used to perform gesture recognition on the time frame. Hence, the computational efficiency can be improved and the testing of partial patterns can be avoided. Moreover, since recognition is done with the gyroscope signal, the user's head orientation needs not be upright, and this method can still detect the tap gestures accurately.

3.4 Implementation and Results

3.4.1 Hardware

Two mobile devices were employed in the experiments: iPod Touch 5 (running iOS 8.4) and Motorola Moto X 1st Gen (running Android 5.1 Lollipop). The gyroscope sensing rate of both devices is 50Hz with iPod Touch being slightly lower (47-50Hz) and the Android phone being slightly higher (50-52Hz). The development environments are Xcode 6.4 and Android Studio 1.3.2, and the gesture training is performed on Alienware desktop computers with Intel(R) Core(TM) i7 CPU 960 3.2GHZ and 12GB memory.

3.4.2 Software

For gesture training, the source code of LIBSVM (version 3.2) was downloaded in C, and compiled for execution on Matlab (version R2015a) using Mex. For gesture recognition, the downloaded C source code in the Xcode and Java source code in Android Studio were directly used.

3.4.3 Experimental Conditions

1. The phone was mounted tightly in Cardboard without phone case to maximize the vibration transmission.
2. Cardboard was mounted firmly against user's head to avoid unwanted movement during tap.
3. Users were not to tap when heads were in motion.

3.5 Results

3.5.1 Evaluation: Performance

Each iteration in training an SVM classifier takes around 11.27 seconds on average, and since there arises a need to optimize the classifier parameters, this requires 1517 iterations in the training, which takes around 4.74 hours on average per classifier. Hence, several desktop computers were employed to train the classifiers in parallel. For gesture recognition, the average time taken to find a time frame by matching a local maximum are 0.0246 and 0.1348 milliseconds on the iPod Touch 5 and Android phone, respectively, while the average time taken to use the SVM classifiers to recognize a gesture are 0.0139 and 0.0231 seconds, respectively. So, the performance is real-time on both devices; here, these response times were averaged over 36 gestures.

3.5.2 User evaluations: Recognition Accuracy

Two user evaluations were performed to evaluate the gesture recognition accuracy with 16 participants: 6 male, 10 female, aged 23 to 31, and six of them (old participants) helped in the training. These user experiments were conducted using the iPhone. Before the experiments, the phone display was synced with a laptop in order to see the detection results when participants perform tapping.

Evaluation 1: In the first experiment, each participant was first given 5-10 minutes to practice the finger-tap gestures while observing the detection results on the Cardboard. Then, they were asked to perform 300 tap actions: 50 actions per gesture in a randomized order. This took at least half an hour, and the participants were allowed to rest whenever they wanted to. Table 3.2 presents the results, showing that both old and new participants can perform gestures with very high accuracy.

Evaluation 2: In the second experiment, each participant was asked to perform any head motion that they could come up with over a period of 30 seconds. This was so that non-tapping actions mis-detected as tapping could be found out. The purpose of such an evaluation was to cross-validate the recognition system's ability

		Left	Right	Top-M	Bottom-M	Left Dbl	Right Dbl
Old	M	100	100	99.33	99.67	98.67	98.00
	SD	0	0	1.63	0.82	1.63	1.79
New	M	99.00	99.80	98.60	99.00	96.00	98.40
	SD	1.41	0.63	1.35	1.70	2.49	2.07

TABLE 3.2: Gesture Recognition Accuracy (%): 10 new and 6 old participants.

to avoid false positives. In this evaluation, no non-tapping gestures that the users performed were mis-detected as tap gestures through the system.

3.5.3 User Subjective Feedback

The 16 users were asked several questions after the trial to gauge their user experience. On a 7-point Likert scale, the average user physical comfort level of their experience was rated at a very comfortable 5.6, where 1 is extremely uncomfortable and 7 is perfectly comfortable. On the question of how much force was used to activate their desired function, the average force was rated at a moderate 3.9, where 1 is lightest hit and 7 is strongest hit. On the question of how tired their fingers and arms were at the end of the trial, an average mildly tired score of 2.7 was noted, where 1 is not tired at all and 7 is extremely tired. Lastly, when asked how intuitive the tapping technique was, an average score of 6.0 was noted, where 1 is non-intuitive and 7 is naturally intuitive. In this context, intuition refers to how natural the interaction method felt. A summary of these results is shown in Figure 3.9.

3.5.4 Discussion

The evaluation was divided into two groups of participants: the first group was familiar with the interaction method, since they had employed it during the training phase. The second group, however, was unfamiliar with the interaction method. The results of the evaluation show that both groups of participants performed comparatively similarly. The left and right taps were most reliably distinguished due to the distinct vibration signatures they develop. The least reliably distinguished gestures were double left and double right. This was due to the varying time duration

FIGURE 3.9: User subjective feedback of various aspects of their experience using CardboardSense.

and patterns of double taps across users. A possible direction to take to reduce the error rate of the double left and right taps will be to collect more training samples for such taps with varying time duration. This could allow the system to detect double left and double right taps even more reliably, although it is already quite reliable with accuracy at 97.67% and 98.15% respectively.

FIGURE 3.10: Mean gesture accuracy for various tap locations as a result of the user evaluation, averaged for 16 participants.

The drop in accuracy for left, right, top and bottom taps were all less than 1% when comparing the new group with the old group, although the results for double taps were slightly different. This was again due to the varying time and speed users took to perform double taps. However, altogether, the differences between

both groups remain small enough to conclude that although CardboardSense has a slight learning curve, it can be adopted quickly by first-time users too. The mean accuracy calculated across participants for each spatial location of the tap is shown in Figure 3.10.

These finds were further bolstered by the subjective feedback received from the participants about their experience using CardboardSense. Fatigue is an integral aspect of physical gesture-based interaction methods such as CardboardSense to minimise. Users rated their fatigue level to be 2.7 out of a 7 point Likert scale where 1 meant not tired at all and 7 meant extremely tired. The rating of 2.7 represents mild fatigue only, and as such does not imply that tapping on the periphery of headsets to communicate with the smartphone is a tiring activity. Similarly, the force required to tap was also rated 3.9, which was a moderate level of force. Conversely, the users rated their comfort level to be 5.6 out of 7. It implies that CardboardSense is a fairly comfortable gesture, which still requires moderate force for the system to reliably distinguish between the tap locations. To mitigate the effect of fatigue further, it is possible to train the system with more samples of feeble taps to ensure robust detection with mild force as well. Users also rated the interaction method to be highly intuitive, which validates CardboardSense as being a natural, usable and comfortable method altogether.

3.6 Basic Interaction Design Guideline

The Cardboard-based tap gestures explored in this chapter are a set of gestures at discrete spatial locations on a cardboard-based headset. As such, they can be mapped to discrete input controls in VR applications.

They can be used to activate buttons or select menu items. Left and right taps can be used to intuitively scroll through a horizontal list of menu items, while double taps enable fast scrolling to subsequent left/right pages. Later, selection can be made by a bottom tap. To aid remembrance of tap gesture assignments, it is recommended that labelled GUI buttons be placed in screen locations (top-mid, top-left, top-right, and bottom-mid) that correspond to their respective tap positions (tap top-M, tap left, tap right, and tap bottom). Since these function buttons may occlude or consume valuable screen space, they can remain hidden

and only appear when the user makes the first tap and then slowly recede to the background when no further taps are detected over a predetermined period.

3.6.1 Enhancing Interactivity of Potential Cardboard Apps

One of the major limitations of current mobile applications that can run on the Google Cardboard is the lack of input interactions besides the existing use of head orientation to change the view within the interactive application. Here we suggest how the interactivity of some common Cardboard applications and interaction tasks can be enhanced with the use of Cardboard Sense gestures.

3.6.2 Activation of Buttons and Selection of Menu Items

The tap gestures can be intuitively mapped to button activation or menu item selection. However, due to the limited variety of tap gestures and availability of four different spatial positions (i.e. left, top, bottom and right), some changes to typical interface design and widget layout should be considered in order to realize an interaction design that is both intuitive and functionally flexible for used with applications employing Cardboard Sense interactivity. Consider an interface with no more than three buttons or menu items to choose from. These input options should employ a spatially horizontal layout that correspond intuitively to the corresponding tap positions available with Cardboard Sense, as shown in Figure 3.11(a). However, when the number of options exceeds the available six spatial tap gestures, the left and right tap gestures can be used to slide through the menu items, with the top or bottom tap used to make the desired selection, as seen in Figure 3.11(b) and (c).

Watching panoramic videos

Most Cardboard apps for watching panoramic videos do not provide video player controller widgets like Play/Pause, Stop, Fast Forward- Reverse and Volume Control. Figure 3.12 shows a possible interface design for watching panoramic videos using the new interactivity afforded by Cardboard Sense. The 3 button-selectable options (Restart, Play/Pause and Stop) are mapped to the 3 positional tap gestures. The fast forward and reverse functions are intuitively assigned to the sliding

FIGURE 3.11: (a) Horizontal button layouts for intuitive tap gesture selections. Handling more than 3 menu items using (b) translucent sliding selection window or (c) the enlarged icons technique employed on Apple iOS desktops.

forward-backward gestures respectively. The volume up-down control slider is presented vertically to associate its control to the right-left tap gestures. Lastly, the top-M tap gesture could be used to exit the application.

FIGURE 3.12: An improved interface to extend the interactivity of panoramic video applications using Cardboard Sense gestures. (a) A specific tapping gesture (L/R) done will cause the corresponding slider bar to appear in order to provide visual feedback of the adjusted level. (b) Context-sensitive buttons like Exit can appear at the top when the video has stopped in order to communicate its activation based on the top-mid tap gesture.

Exploring 3D Virtual Environments and FPS games

FIGURE 3.13: An improved interface to extend the interactivity of first person shooter game using CardboardSense gestures.

First person shooter (FPS) games within 3D virtual environment are popular applications for VR headsets like Oculus Rift. However, the lack of interactivity has prevented serious adoption of FPS games on Google Cardboard. With the user's head movement controlling the viewing direction, other navigational commands such as moving forward-backward can be assigned to the top-L and top-R tap gestures. Selection of weapons and game options (e.g., exit game) can be done using the left and right tap gestures to scroll through the top horizontal menu in the corresponding direction (see Figure 3.13) with a top-middle tap gesture used to make the final selection. The subsequent firing of the selected weapon can also be done using repeated middle tap gesture on the front of the Cardboard. Like before, the volume control can be mapped to the left-right tap gestures.

3.7 Conclusion

This chapter has explored how users can tap on their headsets using their fingers to interact with their headsets. It presented a novel set of finger tap-based gestures to provide discrete inputs for VR applications on such DIY cardboard-based headsets. It has evaluated the method and proved it to be a feasible interaction modality for such headsets.

The proposed interaction method in this chapter uses only the smartphone’s built-in gyroscopic signals to enhance the interactivity of Cardboard applications with finger-tap input modality. Temporally-aligned signature samples of different potential gestures were extracted and progressively eliminated to a set of reliably differentiable gestures, by observing and analyzing the temporal signatures using cross validation. Robust and computationally efficient methods were developed to recognize six different tap gestures. Experimental results with 16 participants confirmed a high real-time detection accuracy rate of 98.9% on average. Moreover, the findings of the evaluations presented imply that users have found CardboardSense to be a comfortable, intuitive and usable method to interact with cardboard-based VR headsets.

A set of guidelines for integrating CardboardSense as an input modality in applications requiring discrete inputs were also discussed and presented.

The next chapter explores eye gaze as an input modality for VR headsets, and proposes a novel eye gaze-based interaction design for VR headsets.

Chapter 4

Gaze-based Interactions in VR Headsets

4.1 Introduction

With the recent advancements in eye tracking technology, several headsets with embedded eye tracking capabilities have been released. This has led to the adoption of eye gaze interaction as an input modality for VR headsets such as FOVE[44]. While gaze interaction is mostly used as an auxiliary input modality in regular desktop environments, with the current dearth in input modalities for immersive VR setups, it seems like a natural candidate for primary user input control. This chapter explores eye gaze-based interactions in VR headsets using a headset with an embedded IR-based eye tracker called the FOVE, that offer continuous gaze-based input control for VR applications (Figure 4.1)¹.

While exploring possible gaze-based interaction designs, one realizes that new and thoughtful mechanisms are needed to address common problems that the users faced in gaze-based interaction. One such problem is the Midas Touch, where users may unintentionally select a user interface (UI) element, e.g., a button, by accidentally looking at it [116]. Common existing solutions to the problem include the gaze fixation, eye blinking, and action-selection [11]. While gaze fixation and

¹The research presented in this chapter has been published in [2]

FIGURE 4.1: DualGaze: A, eye gaze-based interaction method for headsets with embedded eye trackers such as FOVE[44]. This method addresses the Midas Touch problem in gaze-based interaction designs.

eye blinking often incur errors due to unintentional selections, action-selection typically relies on a separate external trigger, which can be problematic due to many VR headsets having limited or no external input capabilities².

This chapter presents DualGaze, a novel interaction method that makes sole use of gaze to address the Midas Touch problem for gaze mediated VR interaction. It can be readily employed in standard VR menu selection tasks without the need of having any external triggers. DualGaze is designed in such a way that when the user gazes at a selectable UI element, a confirmation flag would pop up right next to the UI element for the user to willfully look at to confirm the selection; see the left column in Figure 4.2 for an illustration of the DualGaze interaction procedure. Instead of arbitrarily positioning the confirmation flag, we strategically adapt its location to the user's gaze trajectory, i.e., we put it at a location that user's gaze just passed through right before entering the UI element's boundary. Hence, to confirm a selection, the user needs to consciously avert their eyes back to the flag. In this way, the chance of unintended or accidental gaze at the confirmation flag is reduced, especially when one gazes at a UI element but has no intention of selecting it.

²A summary of this section is given in a short video at www.youtube.com/watch?v=kD-q1AzIlek

FIGURE 4.2: Two comparative interaction designs for selecting a target with one's gaze in a gaze-mediated VR user interface. The DualGaze method (left) and the Fixed Gaze method (right).

Compared to the popularly-employed gaze fixation method (see the right column in Figure 4.2), DualGaze has higher accuracy in terms of avoiding the Midas touch problem. In addition, the increase in accuracy is obtained while only minimally impacting the interaction responsiveness in terms of selection speed. These results were confirmed by quantitative data collected from a user study, which shows that users of DualGaze can perform simple gaze-typing tasks with significantly better accuracy than gaze fixation. Further analysis shows that towards the end of the tasks, users' increasing familiarity with DualGaze resulted in DualGaze outperforming the comparative method in terms of selection speed. This also suggests that the DualGaze technique is not difficult to master. A concluding survey found most users perceive DualGaze to be their preferred gaze interaction method in scenarios where the selection task at hand is deemed critical and irreversible, such as exiting a game.

The chapter is organized as follows:

- Section 4.2 starts with a survey of related literature in the context of this work.

- Section 4.3 details the proposed interaction method, DualGaze, and discusses its design.
- Section 4.4 details a user evaluation done on DualGaze and another baseline method.
- Section 4.5 presents the quantitative and subjective results of this user evaluation.
- Section 4.6 discusses the implications of the results presented.
- Section 4.7 finally summarises the findings from the previous sections and concludes the chapter.

4.2 Related Work

Of late, several HMDs in the market enable external IR-based eye trackers to be embedded inside, allowing eye tracking to be a whole new input modality in VR [117][44]. Studies combining VR eye tracking with BCI have also been ongoing[118][78]. With the introduction of eye tracking to VR, however, there is still a dearth of thoughtful gaze-based interaction methods, especially designed to work with the constraints of VR headsets. While there has been a considerable amount of research devoted to eye tracking in VR [119][40][42], research in gaze-based interaction in VR is yet to catch up.

A work by Jacob and Stellmach[120] comprehensively explores design considerations for gaze-based interaction methods. The work identifies the Midas Touch problem as a key consideration while designing gaze-based interfaces, and presents studies on interaction approaches to address the problem for specific cases. Later, Huckauf and Urbina[121] presented a follow-up work that takes an alternative approach using antisaccades, which are an explicit eye movement studied in cognitive psychology, for object selection. Users performing tasks involving antisaccades fixate their gaze on an object in the centre of a screen, and an eccentrically located stimulus is presented to them. They make a saccade in the direction opposite to the stimulus, with an amplitude equal to the distance between the stimulus and fixation, to confirm the selection. The work by Istace et al. [122] provides a solution to the Midas Gaze problem that uses eye dwells to select objects as in [123],

called Snap Clutch, but employs different modes of operation, such as Dwell Click mode, and Eye Control Off mode.

The recent work by Kasprowski et. al.[124] sheds light on problems faced by the gaze contingent interfaces today, including the Midas Gaze problem. The work also discusses an experiment, in which mouse, touchpad and gaze contingent interfaces are employed to play a simple shooting game. However, the selection method used in the gaze contingent interface is an action-selection method based on an external touchpad trigger. Moreover, there was a significant number of users who stated that mouse input was more convenient than the proposed gaze-based selection method.

Smooth pursuit eye movements have also been suggested solutions to the Midas Gaze problem, in works such as [125] where gaze-based interaction in smart watches is facilitated by having the user's gaze follow a circularly moving target to confirm selection. This requires a more complicated eye movement sequence than that of other discussed methods since it involves a constantly moving target that the gaze has to follow. Our proposed method, on the other hand, employs a target that is fixed and requires only a simple glance-back to activate. Another work by Bednarik et al.[126] employs machine learning to predict the intended gaze location; the accuracy of the system is limited, at 76%. Likewise, there have been several other works that explore gaze-based action-selection methods, e.g., the work by Velichkovsky et. al.[11], which collects all possible solutions to the problem. However, almost all solutions involve the use of an external trigger or input, such as the work by Kumar et. al.[127], which uses keyboard presses to confirm the gazed-at selections. However, gaze interaction in VR can be made more universally implementable and hands-free if the selection method is independent of an external peripheral.

4.2.1 Gaze-based Selection in VR

A recent work by Piumsonboon et. al.[128] similarly explores such new gaze-based selection methods. While one of them uses smooth pursuit eye movement in VR such as in [129], another of them use Duo-Reticles, a spatiotemporal-based selection method in which a moving average is made to align with the eye's current gaze position. However, since the moving average always remains at a minimum

distance behind the user's current gaze, the second reticle is constantly moving, so it is distracting for the user to gaze back at, as was highlighted in their user study's subjective evaluation. The proposed method in this research attempts to solve this by employing a non-moving, yet trajectory-adaptive confirmation flag, at which users can intuitively glance back to confirm the selection in a more predictable manner, as opposed to consciously matching the current gaze reticle back to the moving reticle.

4.3 The DualGaze Interaction Method

Gaze-based interfaces in immersive VR systems typically involve a menu selection, where users have to scan through a number of options on the screen and then select one of them afterwards. This gives rise to some very particular interaction design challenges as the travelling gaze during the visual scanning cannot be reliably distinguished from a wilful gaze on an identified choice.

There are several existing gaze interaction methods that seek to mitigate this problem by introducing a distinguishing action for confirming a choice. Such solutions in VR mainly involve gaze fixation, blinking, or an external trigger such as a remote switch [11], to confirm a selection once a desired option has been visually targeted. Each of these solutions has their respective limitations, as discussed earlier in the chapter.

In this chapter, we present a novel gaze-mediated interaction method called DualGaze that addresses some of these limitations. Firstly, unlike methods that employ external triggers to confirm selections, DualGaze purely employs user's gaze, thus allowing hands-free immersive VR interaction design.

Secondly, DualGaze employs a single and consistent eye gaze interaction modality to perform both the "choosing" action and "confirmation" action in the selection process. Hence, the approach avoids the many side effects of using blinking to distinguish between the visual search and selection. Frequent switching between gaze and blinking could feel unnatural and repetitive blinking can lead to rapid eye fatigue. In addition, momentary occlusion of the iris and pupil during blinking can cause the eye tracking systems to lose their smooth and continuous gaze tracking capability.

Thirdly, unlike the gaze fixation method, the gaze positions for the "choosing" action and "confirming" action in DualGaze are not co-located. As such, carefully planning successive gaze locations can help distinguish the two different but correlated actions. This addresses the limitation of the gaze fixation method that an accidental prolonged gaze on a UI element may cause an unwanted activation. A typical remedy to ameliorate the Midas Touch problem in gaze fixation is to increase the gaze duration before the confirmation, but this degrades the responsiveness of the interactive experience, especially in situations where multiple selections need to be made in quick succession.

Figure 4.2 (left) shows the basic idea behind our DualGaze interaction design. The activation of a target onscreen choice is achieved using a quick successive two-gaze strategy. After the user's gaze enters the target region (e.g., a "clickable" button), a smaller pop-up box called the confirmation flag would appear right next to the target region. By a second gaze on the confirmation flag, the user can confirm the selection. We design the confirmation flag in the form of a simple box, roughly a quarter of the menu button in size, and coloured distinguishably from the target button. Putting the flag close to the target region reduces necessary gaze travel for the users to complete the selection action.

To address the Midas Touch problem, we need an appropriate placement of the confirmation flag to facilitate intentional and deliberate eye gaze trajectory, and to reduce the chance of accidental confirmation. After a brief delay of 0.3 seconds (an empirically derived value) between the gaze entry and fixation on the target region, the confirmation flag would appear, thus further suppressing the chance of unintentional confirmation. To cancel a selection, users may simply look out of the target region but not towards the confirmation flag, as one would normally do. The process of scanning menu options and finally making a selection using DualGaze is illustrated in Figure 4.3 .

The design goal of DualGaze is to provide a reliable, accurate and responsive gaze-mediated interaction method that can be used in an immersive eye-tracked VR environment for making onscreen option selections, which can be in the form of different menu options or keys of a numeric keypad. The next section describes some preliminary studies done in deriving appropriate options in the DualGaze interaction design.

FIGURE 4.3: The stages of the DualGaze interaction method, with the red eye denoting the user's gaze point, involving a user's gaze travelling through buttons 1 and 2 to confirm the user's desired choice, button 3. The positioning of the confirmation flag does not obstruct the gaze trajectory and forces one to gaze back to confirm

4.3.1 Designing the DualGaze Interaction

In designing the DualGaze interaction, we have studied and evaluated design alternatives using small-scale user studies that involve five to seven participants.

A fundamental design variable in DualGaze is the confirmation flag placement strategy for improving the reliability of gaze interaction. Two placement strategies were evaluated. The first strategy is to position the flag at a fixed place with respect to the selected button every time (in this case, the right side of the button), to allow the user to gaze at a predictable and fixed location. The second strategy is to position the flag at the entry point from which the user's gaze enters a particular selection region. In other words, we adapt the confirmation flag position to the trajectory of the user's gaze. The rationale behind the second strategy is that a conscious effort is required by the user to avert their eyes and return to the point where the user's gaze enters the selection region - such a location is salient, yet does not obstruct the user's gaze path as he/she moves forward.

4.3.1.1 User Study on DualGaze Design

A small-scale user study with seven participants was conducted to study which of the two strategies led to better user performance by using the simple number selection task shown in Figure 4.5 (right). The same task was presented to each design strategy and the task requires the user to gaze-type ten numbers that are presented as five sequential pairs of two-digit target numbers. The temporal order of these ten numbers is labelled Questions 1 to 10. The sequential order of the two strategies was alternated between trials to prevent biases due to asymmetric familiarity and/or exposure-induced eye fatigue.

The VR headset used for the study was FOVE VR developed by a Tokyo-based start-up[44]. FOVE uses an infrared technology for accurate tracking of eye movements with low latency. Embedded sensors within the headset track the users' pupils, while they interact with the objects on the screen. Setting up the FOVE headset involves a calibration of an external positional tracking camera, which should always be able to see a full view of the headset, as well as staying connected to the PC through USB wires; see Figure 4.5 for the physical setup.

Accuracy was measured as the number of correctly selected numbers each user made out of the ten required numeric entries. The one-way ANOVA test was done, and the results showed using a trajectory-adaptive flag placement could produce significantly better accuracy (Mean (M)=9.7, $p=0.038$) when compared to the first strategy, which puts the flag at a fixed position (M=7.3). This result confirms that using the trajectory-adaptive strategy to place the confirmation flag can effectively avoid accidental confirmation as users made significantly less erroneous selections during a trial to select ten numbers out of the buttons.

Task completion time was also found to be lower for trajectory-adaptive positioning (M=23.78s) than fixed positioning (M=37.47s). However, the one-way ANOVA test (with $p=0.081$) does not show this difference to be significant. In any case, one can safely conclude that using the trajectory-adaptive placement strategy could produce a more reliable and accurate interaction design that does not compromise interaction responsiveness; see Figure 4.3 for an example of this flag placement strategy.

From the above user study, we complete the final DualGaze interaction design, which adopts the trajectory-adaptive confirmation flag placement strategy.

4.3.2 A Comparative Gaze-based Interaction Method

Gaze fixation [11] is a popular gaze interaction method, by which users stare at an onscreen target region for a pre-determined period to select and confirm his/her choice on the target. We choose this method as the baseline in our comparative study, since like DualGaze, gaze fixation is also a pure gaze-only interaction design. While gaze fixation has been in widespread use and various aspects of its design have been analyzed [130], we conducted a small-scale user study with six participants using our eye-tracking system based on the FOVE VR headset to determine the appropriate delay-to-activation time setting for the experimental setup, so that we can obtain a fair and uniform comparative evaluation against DualGaze.

FIGURE 4.4: (a) Accuracy of three different gaze lengths. (b) Selection Speed for the three lengths. Statistical significance is denoted by asterisks (**= $p < 0.01$)

Three different durations were used in the study to evaluate their relative accuracy and total time taken to input ten numbers in the same number selection task; their results are shown graphically in Figure 4.4. The implementation and set up are described in 4.3.1.1. The delay durations used were Short (0.5s), Medium (1.0s) and Long (1.5s). The results from the one-way ANOVA tests show that Medium ($M=7.5s$) has significantly better accuracy than Short ($M=3.3s$) with $p=0.002$, while Long ($M=9.3s$) has better accuracy than Medium, though the improvement was not significant ($p=0.072$). As for the task completion time, Short ($M=10.4s$) was significantly faster ($p=0.002$) than Medium ($M=19.4s$) and Medium was significantly faster ($p=0.002$) than Long ($M=30.5s$). As a result, a delay of 1.0 second was chosen to be the best empirical parameter to trade-off between accuracy and task completion time for the method, which we will refer to it as "Fixed Gaze." This duration was adopted in the comparative study with DualGaze.

4.4 User Study

4.4.1 Experimental Design and Set-Up

A within-subjects user study was conducted to investigate and compare the performance of DualGaze with that of Fixed Gaze. The same task was presented to each user and the task requires the user to gaze-type ten numbers that are presented as five sequential pairs of two-digits target numbers. In this study, the temporal order of these ten numbers is labelled Questions 1 to 10. The sequential order of the two methods was alternated between trials to prevent biases due to asymmetric familiarity and/or exposure-induced eye fatigue. In short, a user would firstly gaze-type ten numbers using Method 1, followed by the same ten numbers using Method 2, where Methods 1 and 2 were alternately Dual and Fixed Gaze for each user.

The implementation and set up are described in 4.3.1.1. The tasks given to the users were FOVE-based applications developed on Unity IDE using the FOVE's Unity-based SDK, on a stereoscopic VR view. Two separate DualGaze and Fixed Gaze applications were developed for the study, and the user's accuracy and time performance data were collected through log files at the end of each task. Figure 4.5 (right) shows a sample layout of the UI.

FIGURE 4.5: Left: The physical setup, with the FOVE HMD used in this work. Right: A sample layout of the UI displayed on the FOVE HMD, where users have to gaze-select ten numbers on the keypad shown.

4.4.2 Participants and Procedure

4.4.2.1 Participants

The user study was conducted with 27 participants. The participants' age range from 18 to 30 years, and they were university students and staff recruited through flyers after we obtained an ethical approval for the study (IRB-2017-10-047). While 41% of the participants had used VR headsets before, 86% of them were first-time users of an eye tracking system.

4.4.2.2 Procedure

Once a user signed a consent form, a brief explanation was given to the participant regarding the order of the proceedings and strapped the VR headset comfortably around their head. Then, they were asked to perform a calibration routine, whereby their eyes have to follow a green dot moving around the screen. Once the calibration was successful, the participants were presented the first application, where they were required to input five 2-digit numbers from a numeric keypad (see Figure 4.5 (right)) using the selection method particular to the current application. In all cases, the participant used their gaze to select numbers from the numeric keypad and a beep was sounded every time the participant has successfully selected and confirmed a number. This was repeated for the other interaction method, with a minute's rest for the participant in between each task. Their final score (number of digits they input correctly), and other aspects of their performance such as time spent on each question, were recorded via a log file generated by the program.

After completing both tasks, the participants were asked to fill a survey regarding their experience and preferences for the two different gaze selection methods.

4.5 Results

4.5.1 Accuracy

An important goal of gaze-based interaction is to allow users to perform their selections in a reliable and accurate manner. In this study, the following research

question is asked:

RQ1. *Is DualGaze more accurate than Fixed Gaze?*

Accuracy was computed as the number of correct entries out of the ten digits that the user made by using each of the two interaction methods. Results from the one-way ANOVA tests showed that DualGaze is significantly more accurate than Fixed Gaze ($p=0.0004$). Participants tend to make more errors using Fixed Gaze ($M=7.7$, $SD=1.56$) than when using DualGaze ($M=9.19$, $SD=0.77$). This is shown graphically in Figure 4.6(a).

4.5.2 Selection Speed

Another important goal that affects user experience is the responsiveness of an interaction method. This is directly related to the speed in which the required selection can be made by the user. The next question considered was:

FIGURE 4.6: (a) Accuracy of both methods. (b) Selection Speed for both methods, with Standard Error bars. Statistical significance is denoted by asterisks (**= $p<0.01$)

RQ2. *Is DualGaze faster for users to make a selection than Fixed Gaze?*

To answer this question, the average time that users took to input a correct number was computed. The result is shown graphically in Figure 4.6(b). The time taken to input a correct number begins when the gaze moves from a previously-selected digit to the current selected digit and ends when a correct confirmation is registered. DualGaze ($M=2.13s$, $SD=0.39s$) was found to be slightly faster than Fixed Gaze

FIGURE 4.7: Eye gaze trajectory denoting travel time and selection time as gaze moves from one answer to the next.

($M=2.15s$, $SD=0.42s$) in terms of the average time taken for making a correct answer, but such time difference was not statistically significant ($p=0.868$). In fact, the selection time taken is affected by factors such as the travel time as well as the selection time, and the travel time in turn may depend on successive locations of the digits, so the visual search delay and the repeated incomplete attempts. See Figure 4.7 for a visualization of the eye gaze trajectory as it moves from one number to the next. If external factors such as travel time (T) were eliminated and only the average selection time (S) was considered, then DualGaze ($M=0.65s$, $SD=0.36s$) is found to be significantly faster than Fixed Gaze ($M=1.01s$, $SD=0.006s$) with a ($p=0.00001$). This result implies that the time users spent doing the first gaze on a digit and then the second gaze on its corresponding confirmation flag in DualGaze is still faster than a single fixated gaze in Fixed Gaze.

4.5.3 Temporal Performance Changes

We observed in answering RQ2, users were not significantly faster in making their selections using DualGaze compared to Fixed Gaze when averaging over the entire experiment. However, DualGaze is a novel and conceptually more complicated gazed-mediated interaction method than Fixed Gaze, and users were not familiar

FIGURE 4.8: Temporal performance changes in selection speed, with Standard Error bars.

with its operation at the start of the experiment. As such, we were interested in finding out if users were able to improve their performance in selecting numbers after more practice, as they continued to use DualGaze in the experiment. We then posed the following question:

RQ3. *Is DualGaze faster in making a selection than Fixed Gaze in the latter half of the experimental run?*

To answer this question, the average time that the users took to input a correct number for the last five questions were computed. Results from the one-way ANOVA test showed that users using DualGaze ($M=2.1s$, $SD=0.93s$) were observed to take significantly less time ($p=0.0476$) to make correct selections compared to Fixed Gaze ($M=2.4s$, $SD=1.25s$). This result testifies to DualGaze’s fast learning curve. It can also be seen in Figure 4.8 that the selection speed improvement effect with the users’ familiarity can be observed in DualGaze, but not in Fixed Gaze. While DualGaze started off on a significantly slower than Fixed Gaze, it caught up quickly and consistently surpassed Fixed Gaze with a lower task completion time towards the end.

FIGURE 4.9: Temporal performance changes in accuracy.

4.6 Discussion

4.6.1 Accuracy

Our user study results show that DualGaze has a marked advantage in accuracy over Fixed Gaze. DualGaze was designed with the primary goal of minimizing involuntary selection. The design provides an additional step of confirmation by requiring the users to avert their eyes to the confirmation flag. In addition, the adaptive trajectory placement of confirmation flags also reduces the likelihood that the user’s current gaze direction will accidentally trigger this flag. This resulted in a significantly better accuracy, since users had to consciously confirm a choice by reversing the current direction of their gaze to the confirmation flag. Fixed Gaze on the other hand, uses the same action (i.e. gaze fixation) to select and confirm an onscreen selectable region. This means users will make a selection whenever their eyes rest on one of these selectable regions. The only ways to avoid making a selection are to either look away to non-active regions of the screen or keep one’s gaze roving all the time. The former may be challenging if the visual interface has many large active selectable regions and the latter can result in discomfort and rapid eye fatigue. Interestingly, our quantitative results of the temporal changes in accuracy plotted in Figure 4.9 show that the accuracy for both interaction methods starts to drop as the experimental run progresses. This could be partially due to fatigue or users’ inability to sustain their attention level in the given tasks.

However, what is noticeable is that the drop in accuracy for DualGaze is only marginal compared to that of Fixed Gaze.

Users were asked using a survey questionnaire to rate the accuracy of DualGaze and Fixed Gaze. Their responses showed that only a slightly higher number of users perceive the accuracy of DualGaze to be better than Fixed Gaze despite the undeniable quantitative performance measures indicating DualGaze's significantly better accuracy. Those who rated DualGaze as more accurate felt that DualGaze gave them more assurance of the option they were selecting. According to one user, *"There was more accuracy in the first case (DualGaze) as two stage confirmation is involved."* Another user remarked that *"The selection of an option with its acknowledgement is preferable"*. These remarks seem to mostly concur with users' preferences when they were given various use-case scenarios (see Figure 4.10).

Scenario 1 in Figure 4.10 was designed for single-object selection. It consisted of confirming a single option from an onscreen menu having three options. The catch was that the selected option would move to the centre of the screen and get enlarged. 67% of the users preferred Fixed Gaze over DualGaze for this scenario - since there was only one selectable option, users prioritised ease of use over accuracy. Common reasons included *"Easier eye movement"* and *"Assuming the effort to recover from a wrong choice is minimal, I would choose Fixed Gaze"*. This highlighted the users' preference of choosing the relatively simpler method when the task is easy and reversible, and where accuracy was not a priority.

Scenario 2 involved having to select and confirm a single option from an onscreen multiple-choice quiz having three equally-spaced options. The major concern to address in this use-case scenario was the issue of reliability and accuracy as this was a quiz with multiple options and choosing a correct option would be deemed as a critical decision. It was observed that more users (59%) preferred to use DualGaze than Fixed Gaze for their selection. Reasons given by users included: *"This allows me to confirm my answer."* and *"Getting my answers right in the quiz is very important. It feels like fixed gaze can have a lot of accidental clicks"*.

Scenario 3 in Figure 4.10, presented a VR game, out of which the user had to select a red exit button using one of the two selection methods. It was an interesting scenario with even more interesting answers and implications - 41% of the users felt that Fixed Gaze would do the job just as well, since there was only one button

FIGURE 4.10: Survey responses to users' preferred selection method in use-case scenarios

to select. Such users had reasons along the lines of *"Since it's just an exit option and there is only a single button, fixed gaze is easier."* 59% of the users, however, realized the significance of the Exit button and chose to go with DualGaze, the primary reason being to avoid accidental exits. Users who chose this had comments such as: *"In Fixed gaze I might press the exit option accidentally"* and *"I will lose progress if I exit accidentally"*, which highlighted their sense that DualGaze is a more reliable and accurate interaction method.

4.6.2 Selection Speed

An important feature of the proposed DualGaze interaction method is that it managed to achieve high accuracy without compromising on the selection speed. Unlike the Fixed Gaze method, improved accuracy can only be obtained by increasing the delay-to-activation time (see Section 4.3.2) and thus the corresponding time required to complete the selection task. The temporal changes in selection speed plotted in Figure 4.9 also highlight the potential for improvement in selection speed as users become familiar with the DualGaze interaction design and can quickly anticipate where the confirmation flag is likely to appear based on their current gaze trajectory.

Responses from the survey questionnaire suggest that some users seem to perceive that Fixed Gaze has a slightly speedier selection time than DualGaze. Some of the users who had a false sense of Fixed Gaze having better selection speed were also users with more selection errors due to the higher accidental selection with the Fixed Gaze method. As a result, they felt they completed the given task of selecting 10 numbers much faster with Fixed Gaze, and they apparently did not take into account their high error rates in the tasks (which in turn led to them fast-forwarding through the set of 10 questions quickly). This observation highlighted to us the limited ability of a subjective survey questionnaire in evaluating the true performance of an interaction design during a user study. Quantitative measures are still the more objective assessment tool to assess non-experiential performance metrics like accuracy and selection speed.

In short, the general consensus among the users was that DualGaze would be preferred for the assurance and trust that it gave users while using it, given it felt like an interaction method more akin to a mouse click or a touchscreen tap in traditional interaction modalities.

In summary, both qualitative and quantitative results prove the feasibility of DualGaze as a gaze-based 2D object selection method. The primary advantage of DualGaze is its accuracy and ensuing reliability as a gaze-based object selection method in VR. In use case scenarios that prioritise accurate and reliable object selection, DualGaze has proved to be the reliable and preferred method when compared to the traditional Fixed Gaze method. This advantage of increased accuracy has also been corroborated by the user evaluations conducted.

In future, the design of DualGaze can even be extended as a 3D object selection method. Such a design can be realised through tracking the 3D trajectory of the user's eye gaze, and adapting a suitable flag placement strategy guided by the original design of DualGaze.

4.7 Conclusion

This chapter has presented and evaluated a novel gaze-based interaction method for VR headsets with eye tracking capabilities. The proposed interaction method, DualGaze, has addressed the Midas Touch problem in gaze-based interactions in

VR, proving to be more effective than Fixed Gaze, which is currently a widely adopted eye gaze selection technique for VR applications.

DualGaze is more accurate than the popular Fixed Gaze interaction method as users made significantly less errors in making selections over onscreen buttons. As a result, users also perceived DualGaze to be more reliable as the majority of users picked this method over Fixed Gaze in use-case scenarios where the activation of a button is deemed critical and irreversible. They were also able to perform this selection a little faster using DualGaze. Analysis of the average selection time for DualGaze showed that it is significantly faster than Fixed Gaze. However, this significant speed superiority was not evident in the total task completion time, which took into account other external factors like the time expanded in making repeated incomplete flag confirmation. Preliminary evidence was presented that this task completion time can be reduced as users became more familiar with DualGaze.

The user evaluation of this interaction design presented a number selection task which users completed successfully. VR headsets with eye tracking can employ DualGaze as an accurate gaze-based target selection or number entry method for VR applications.

The next part of this research explores hand and finger based interaction gestures in VR headsets.

Chapter 5

Single-Input Interactions with Fingers

5.1 Introduction

FIGURE 5.1: Single-input finger-based interactions explored in this chapter, all using a smartphone’s RGB camera.

In the previous chapters, the research explored tap-based and eye gaze-based gesture interactions for VR headsets.

The next part of the research aims to widen the currently available range of input modalities in VR by exploring hand and finger gesture-based interaction using

FIGURE 5.2: The three proposed interaction techniques: (a) Technique S1 : Twin-Fingers (b) Technique S2 : Bunny-Fingers; (c) Technique S3 : X-Fingers.

the smartphone camera inside a VR headset. The research proposes three novel interaction designs(see Figure 5.1).

The first interaction design developed with this vision-based tracking modality is a novel two-handed gesture interaction design that is invariant to head motion, called Twin-Fingers. The technique provides an interactive 2D cursor control in VR, where the user slides two overlapping splayed fingers over two from the other hand to perform the gesture.

In addition, a second, more conventional single-hand gesture design (called Bunny-Fingers) is also considered and its performance is compared with Twin-Fingers on an onscreen cursor-based pointing task that involves both head movement and finger-gesture interaction. A user study shows that most users prefer the simplicity and convenience of a single-hand gesture for tasks requiring no head movement. However, with limited practice, users performed better with Twin-Fingers when tackling tasks that required both head movement and finger gestures.

When it was found that Twin-Fingers was not a straightforward and convenient interaction design to perform, a third interaction design was developed in a similar vein, called X-Fingers, which involved only one (index) finger from the dominant hand sliding over the index finger of the non-dominant hand. X-Fingers required less effort to perform the gesture by the user, while retaining Twin-Fingers' properties of invariance to non-dominant arm and head motion. A second user study was conducted where two sets of users performed the same two tasks using X-Fingers as well as Bunny-Fingers. In this study, however, it was found that users performed both tasks faster with X-Fingers, leading to the conclusion that X-Fingers is an interaction design that combines the properties of non-dominant arm and head

motion invariance from Twin-Fingers, while maintaining the relative simplicity of gesture that Bunny-Fingers also offers.

The chapter is organized as follows:

- Section 5.2 surveys recent literature on hand and finger-based interactions in VR.
- Section 5.3 introduces Interaction Design S1 (Twin-Fingers) and discusses and evaluates its design.
- Section 5.4 introduces the baseline method, Interaction Design S2 (Bunny-Fingers) and discusses its design.
- Section 5.5 details a user evaluation done on S1 and S2, presents the results and discusses their implications.
- Section 5.6 introduces Interaction Design S3 (X-Fingers) and discusses its design.
- Section 5.7 details a user evaluation done on S2 and S3, presents the results and discusses their implications.
- Section 5.8 finally summarises the findings from the previous sections and concludes the chapter.

5.2 Related Work

There have been several works which present mid-air pointing methods such as the works of [46] and [45]. Such hand gesture interactions have also long been viewed as a natural way to interact with VR HMDs. The work by Kato et al[131] presents a knuckle-based pointing gesture for HMDs. The system by Guginheimer et al[52] proposes an interface that detects touch gestures on a touch-sensitive surface of the HMD. However, these works all employ the use of external peripherals or sensors. Among these, a recent work is FistPointer [51], which employs hand gestures to navigate in a virtual environment without requiring any additional controllers for VR headset users. This technique allows the user to select a target by means of a mid-air interaction behind the smartphone. However, this approach relies on the

FIGURE 5.3: (a) The FistPointer interaction method employed by Ishii et al [51] that relies on absolute hand position tracking. (b) The Twin-Fingers interaction method that relies on tracking the relative movement of negative space between both pairs of fingers.

absolute position of the user's fist to locate an onscreen cursor, which is susceptible to changes in head movement while wearing a VR HMD. This chapter attempts to design efficient user interaction methods addressing such challenges that are independent of the external controllers, by exploiting smartphone camera-based interaction methods to track finger gestures. The proposed gestures can be used as effective alternatives for controlling onscreen cursors in VR.

5.3 Interaction Design S1: Twin-Fingers (TF)

Twin-Fingers (TF) is a vision-based gesture interaction method for use with VR headsets that uses an embedded smartphone with a rear video camera. The goal of the TF design is to allow the movement of the fingers to convey two smoothly varying parameters that can represent the x and y positions of an onscreen cursor. The gesture-based parameters must be stable and reliably measurable in a scale-invariant manner using computer vision algorithms. More importantly, these parameters must be invariant to the movement of the camera. A pointing technique such as FistPointer[51] relies on tracking the absolute location of the user's hand to convey an onscreen cursor position. This position may change when the user moves their head or VR headset, thus changing the embedded smartphone camera's viewpoint. TF is a gesture guided by this consideration. It tracks a user moving

FIGURE 5.4: (a) The three areas of negative space formed between four fingers that are used to determine cursor position; (b) Fingers of the dominant hand move down to up vertically over the other hand; (c) Progressions of length of negative space with time, (d) Movement of screen coordinates (the y-axis starts at 0 from the top of the screen), both during the movement in (b).

the fingers of their dominant hand relative to their stationary non-dominant hand, unlike *FistPointer* (see Figure 5.3).

Skin detection algorithms involved in hand and finger tracking also come with a few challenges. Due to variations across skin colour, lighting and shadow conditions, skin segmentation algorithms produce varying results across several users and environments. In contrast if the negative space in between or around skin is tracked as opposed to the skin itself, such challenges may be avoided. Such negative spaces can be easily made uniform and consistent by performing the gesture in front of plain-coloured surfaces or tables. This consideration further motivates the design of the TF interaction gesture, that relies on the idea of tracking the relative movement of negative space between users' fingers. The following sections detail the design of the TF gesture.

5.3.1 Interaction Design

The *Twin-Fingers* gesture involves sliding two splayed fingers of the dominant hand over two splayed fingers of the static non-dominant hand. Vertical cursor movement corresponds to the vertical movement of the fingers of the dominant hand and the horizontal cursor movement corresponds to left-right movement of the dominant hand. Such a design allows cursor control over a custom area of a user's screen using

only slight finger movements by the user's dominant hand, rendering the user free to move their head about without affecting the cursor position. Moreover, users are free to use either the back of their hands or front in performing this gesture.

Twin-Fingers employs the principle of tracking the negative space(or *gaps*) in between skin, as opposed to tracking features of the skin itself. The areas tracked between the fingers are used to identify the relative X-Y coordinates of the cursor on screen. The length of the right gap, L_x , is used to compute a stable X position P_x , and the length of the left gap, L_y , is used to compute a stable Y position P_y , as shown in Fig. 5.4 (a) and (b).

$$P_x = \frac{L_x \times ScreenWidth}{L_{totalx} \times Scale_x}; P_y = \frac{L_y \times ScreenHeight}{L_{totaly} \times Scale_y} \quad (5.1)$$

From Equation (5.1), screen coordinates P_x and P_y are calculated to position the onscreen cursor. L_{totalx} and L_{totaly} are the total lengths of the left and right gaps as obtained from the initial calibration process detailed in the next section. $Scale_x$ and $Scale_y$ are also calculated from the data obtained from the range test, as detailed in the section after. $ScreenWidth$ and $ScreenHeight$ are the width and height of the screen space over which the cursor will move, respectively.

Figure 5.4 (c) and (d) illustrates this by showing graphs of the progression of various parameters as a user's right hand moves upwards over their left hand; see Figure 5.4(b). That the distinguishing parameter of movement by one finger is always with respect to the other finger results in an important property: invariance. This is of two kinds.

Firstly, if a user moves their head or headset (thus changing the smartphone camera's viewpoint), TF's design ensures that the onscreen cursor position remains unaffected.

Secondly, changing the position of the hand will not change the cursor position on the screen in any way. This is because of the unique design of TF, where at every point the ratio between the negative space areas will remain unchanged even if the absolute position of the two contacting hands changes in 3D space.

5.3.1.1 Calibration and Scale Invariance

A calibration procedure was found to be necessary to implement the design accurately. This is to detect extremities of each user's left and right gaps, as well as record all possible directions the user can move in. This calibration consists of a 2-second procedure by a user to detect the total length of the gaps, while the user performs the gesture with both fingers at the extremities. This is followed by a 6-second range test, where users are free to slide two fingers of their dominant hand over their other hand in all directions and extremities that they possibly can reach. The range test serves the dual purpose of making users associate their sliding directions with cursor movement on screen, as well as recording all possible extremities and locations that the hand can slide to without losing a view of the gaps, to aid in scale-related cursor position calculations; this way the cursor movement is also invariant to scale changes by the user's hands. This property of scale invariance is shown in Figure 5.5, which depicts graphs showing TF's invariance to scale as a user's hand moves away from and towards the smartphone camera encased in their VR headset. The effective working space of the TF gesture is essentially the length of the gap between the non-dominant fingers, and these hands should maintain their position to be within the smartphone camera's field of view. This is the effective working space.

Equation (5.2) calculates screen coordinates using information on scale gleaned from the range test that users perform, in which the system records possible combinations of area ratios for both right and left gaps, given by R_x and R_y , calculated as follows:

$$R_x = \frac{A_x}{A_c}; R_y = \frac{A_y}{A_c} \quad (5.2)$$

Where A_c , A_x and A_y are the areas of the centre, left and right gaps respectively as shown in Figure 5.4 (a). At every point thereafter, when the combination of R_x and R_y that is unique to a position in TF matches with one from the range test, the scale ratios $Scale_x$ and $Scale_y$ are calculated as a ratio of the current length versus the length at range time.

FIGURE 5.5: (a) Graph of changing areas of the left, right and center gaps as scales change. (b) Graph of stable ratios of left and right areas with respect to the center gap's area

The calibration process detailed above also takes into account individual users' capabilities of exceeding the extremities recorded during the range testing due to different hand positioning or gap widening, and recalibrates accordingly during the application's runtime. Such a procedure is adaptive and tailors itself to suit individual users' finger movement capabilities.

5.3.1.2 Head Motion Invariance

The rear camera that tracks the user's hand is mounted on the user's head, via its HMD. Hence, moving the user's head changes the viewpoint of the rear camera as well.

The hand-based cursor controls available today rely on the absolute location of fingertips or hands to decide on a cursor position. In contrast, because TF relies on the relative position of two fingers of the dominant hand with respect to that of two fingers from the second hand to determine the cursor position onscreen. Since most cursor controls rely on absolute hand positions, moving a user's head around will affect the absolute position of the hand as well, thus affecting the cursor

FIGURE 5.6: Absolute XY Positions of the user’s hands (C_x and C_y respectively, as seen from a moving viewpoint) with respect to the stability of the cursors(P_x and P_y); this demonstrates that Twin-Fingers is invariant to head motion in a VR headset.

position. TF does not suffer this disadvantage due to its design, as a result of which this interaction method is invariant to head motion. An experiment was conducted to prove the same, where a graph depicting varying head movement (denoted by tracking the centroid of the hand C_x and C_y) plotted along with the cursor position was charted in Figure 5.6. It can be seen from the graph that the cursor position is still stable with respect to the moving head position.

5.3.1.3 Hand Motion Invariance

While wearing a VR HMD, it is natural for users to move their hands around, thus changing the hand location in an absolute cursor system. However, due to Twin-Fingers’ design, moving the entire two hands together will not affect the cursor’s position. This was put to test in an experiment where the cursor position (denoted by tracking the centroid of the hand C_x and C_y) was plotted against the hand’s absolute position in the X and Y axes. Figure 5.7 shows the graph, from which it is seen that while the absolute location of the hand is in constant motion (while the fingers are kept constant with respect to one another), the cursor position remains constant and stable.

FIGURE 5.7: Absolute XY Positions (C_x and C_y respectively) of the user’s moving hands with respect to the stability of the cursors; this demonstrates that Twin-Fingers is invariant to hand motion.

5.4 Interaction Design S2: Bunny-Fingers (BF)

Conventional vision-based gesture interaction techniques such as FistPointer[51] employ only a single-handed gesture where some consistently visible feature (e.g. tip of the thumb) is tracked and mapped to the onscreen cursor position. In this work, we devised a similar single-handed gesture that employs the guiding principle used in Twin-Fingers (TF), namely the detection of negative space (or gaps) between the fingers. We called this interaction method Bunny-Fingers (BF) due to the bunny-like shape the fingers form.

It is a simple method that uses the negative space created when users pinch their index finger and thumb together (Zone A) as shown in Figure 5.8. The vision algorithm tracks the inner tip of the gap where the two fingers pinch each other and maps its 2D position to the onscreen cursor position. Its primary difference from the TF design lies in the fact that it relies on absolute finger positions, much like the design of FistPointer.

FIGURE 5.8: The Bunny-Fingers interaction gesture, with the user's thumb and forefinger pinched together. The negative space between fingers, Zone A, is tracked by the smartphone camera, and the rightmost tip of the zone controls cursor position.

$$B_x = \frac{I_x \times w}{C_w}; B_y = \frac{I_y \times h}{C_h} \quad (5.3)$$

Equation (5.3) denotes the calculation of the onscreen XY cursor positions B_x and B_y respectively. The inner corner I is taken as the leftmost point of Zone A. C_w and C_h are the width and height of the frame captured by the camera respectively, while w and h are the width and height of the screen space over which the cursor will move.

This method was subsequently used as the comparative method in the user studies.

5.5 Evaluation: TF and BF

A user study was carried out to evaluate the speed performance of the Twin-Fingers (S1) and Bunny-Fingers (S2) interaction methods for an onscreen cursor-based pointing task; see Figure 5.9. Number entry on a virtual 9-digit keypad was employed to solicit intentional virtual cursor control from the user. User evaluations were done using two different user tasks. While the first does not involve head movement, the second requires a combination of head and finger gesture movement to complete the tasks, which allowed us to investigate the primary rationale for the TF interaction design. Given the novelty and unusualness of the TF gesture, an

FIGURE 5.9: (a) The Twin-Fingers interaction, with the 3 gaps that are tracked (A_x , A_y and A_z). (b) The Bunny-Fingers interaction, a comparable alternative to Twin-Fingers that is variant to head motion, tracked by a single gap (Zone A)

additional study was done with a small subset of participants repeating the same tasks at a separate sitting to observe if practice and familiarity can improve their performance.

FIGURE 5.10: Screens displayed for both tasks in the user study. (a) Numbered grid with direct selection for User Task 1; (b) Numbered grid surrounded with numbered bubbles to burst with slight head movements, for User Task 2; (c) A user performing a task using TF.

5.5.1 Experimental Design and Set-Up

User Task 1: In the first task, the user has to perform is a simple number selection task shown in Figure 5.10(a). It essentially serves as a practice exercise that allows users to familiarize themselves with using TF and BF gestures to navigate an onscreen cursor in the VR environment. A single-digit number is displayed on screen ("Type N"), which the user should then input by navigating the red spot cursor with their fingers to the corresponding number box within the 3×3

digit keypad. The user selects the number by briefly holding the cursor within the number box for 1 second. For each interaction method, the user is required to enter the same fixed set of 10 different numbers twice. The time a user takes to navigate from one correct number to the next is recorded after every trial. For fair performance comparison, users repeat the same fixed set of 10 different numbers for both TF and BF trials.

User Task 2: In this task, an additional four number bubbles appear around the four sides of the square keypad, as seen in Figure 5.10(b). Users make slight head movements to select each numbered bubble, which then gets highlighted. They then "burst" the selected bubble by using their finger gestures to navigate the onscreen cursor to the same number on the 3×3 digit keypad. The time users take from selecting a bubble to navigating to its correct matching number on the keypad is recorded. This was repeated for two rounds per user, thus making the user select a total of 8 numbers using each interaction method. All measured timings for both user tasks exclude the 1 second selection dwell time. Paired T-tests were employed to evaluate the significance of the results.

All mobile applications employed in the user study were developed in Android Studio using Java and OpenCV. The computer vision algorithms used simple color-based skin segmentation to detect the fingers as well as the gaps between them. Users were made to perform the gestures in front of a white background to improve the reliability of the skin segmentation and gap tracking algorithm. The applications ran on an Android smartphone running Android 7.1.1 Nougat, with a rear camera (OnePlus 5, Qualcomm Snapdragon 835 octa-core 2.45GHz, 16MP (IMX398), f/1.7, 1.12-micron pixels, EIS Dual LED flash, 4K 30 fps / 720p 120 fps video), and the VR headset used was a regular plastic headset made by SilverTec [132][133].

5.5.1.1 Participants and Procedure

Ethical approval was obtained before the commencement of all user studies in this chapter (IRB-2018-05-043). The study was conducted with 6 participants whose ages range from 22 to 36 years. They were university students and staff (2 females and 4 males, all right-handed). Participants were first shown an introductory animated PowerPoint presentation which explained both interaction gestures and

the tasks they were required to perform. To avoid sequencing bias, the order of interaction methods was counterbalanced with each participant. Using the first interaction method, each participant completed User Task 1 first followed by User Task 2 before they did the same two tasks using the second interaction method.

5.5.1.2 Results and Discussion

User Task 1 requires users to navigate the onscreen cursor without interference from their head movement. The mean time for making a selection in User Task 1 using BF is ($M=2.3s$, $SD=2.0s$) compared to TF's slower ($M=6.1s$, $SD=10.7s$). Not surprisingly, the more natural one-handed BF interaction gesture being much easier to master resulted in a significantly better time performance ($p<0.0001$) than TF. When comparing the difference in average selection time for the first and second sets of 10 numbers entered, there were significant performance improvements in average selection time in the second set of numbers for both BF ($M=2.9s$, $SD=2.6s$ for 1st set, $M=1.6s$, $SD=0.9s$ for 2nd set; $p=0.0002$) and TF ($M=9.6s$, $SD=14.3s$ for 1st set, $M=3.9s$, $SD=3.9s$ for 2nd set, $p=0.01$). This is clear evidence that users are getting faster in doing the task with increasing usage and familiarity with the gestures and their respective characteristics. The drop in standard deviations also suggests that users are also getting more consistent with their navigation of the cursor.

User Task 2 now introduces slight head movements as users navigated the onscreen cursor towards a number they selected by moving their head (either up, down, left or right). The mean time for making a selection in User Task 2 using BF is ($M=1.9s$, $SD=1.0s$) and TF is ($M=2.2s$, $SD=3.1s$). Unlike the results in User Task 1, with the incorporation of head movement, the performance of BF is no longer significantly ($p=0.26$) better than TF. While performing this task using BF, users found their cursor sliding backwards with every head movement as they attempted to guide the cursor to the matching number. In TF, however, they found the cursor to be stable despite their head movements whilst still keeping their fingers inside the view of the camera. However, it seems that the mean selection time of BF is still superior to TF, albeit not significantly so. When users were questioned about their perceptions of both methods after doing User Task 2, they mentioned that while TF remained stable and thus offered a faster navigational option than BF, some of them still found relating their finger movements to that of the cursor

onscreen tricky and hence took a longer time to figure their way to each number. This suggests that the proposed TF gesture is more difficult to master.

The results from the first set versus second set analysis in User Task 1 do suggest that gesture-based interaction has an inherent learning curve in its mastery for efficient usage. We were therefore interested to study users' comparative performances in User Task 2 if they were allowed further practice on both the TF and BF interaction methods. A subset of the participants (three users) from the previous study was invited to repeat User Task 2 at a different sitting. This time, they were free to practice either method on User Task 1 for as long as they wanted to and only proceed to User Task 2 once they were confident enough about navigating using both methods. This time around, the mean time for making a selection in User Task 2 using TF ($M=1.1s$, $SD=1.3s$) is significantly ($p=0.003$) faster compared to BF ($M=2.0s$, $SD=0.8s$). Results from the subset of repeated users who are familiar with the proposed TF and BF interaction methods show that they were able to perform User Task 2 significantly faster using TF as they were observed to utilize the head movement-invariant design of TF to simultaneously move their heads to select bubbles and quickly navigate to the matching keypad numbers. In the case of BF, they had to re-orientate their cursor after their head movement.

In short, TF may have a steeper learning curve, but its benefits in performing efficient cursor navigation along head movement can be reaped once sufficient mastery has been achieved. In many VR applications, head motion-based viewpoint changes during gameplay is a common feature, and the User Task 2 in the user study is meant to simulate such scenarios where head movement interaction is closely integrated alongside hand gesture-based navigation control on the VR screen. The user study shows that the proposed interaction design has a steeper learning curve than conventional one-handed gestures but with some practice, users are able to attain sufficient mastery of the gesture to achieve cursor navigation performance that are comparable or better than the one-handed gestures in VR environments that integrates simultaneous head motion interactions.

5.6 Interaction Design S3: X-Fingers (XF)

The results from the previous study indicate that while the tracking methodology of TF leads to a stable cursor position that is invariant to head movement, it is a relatively complicated gesture that requires some practice for mastery. This motivated the design of a simpler gesture that retains TF’s invariance to head motion. Employing the concept of keeping a reference finger, a new interaction design was developed called X-Fingers (XF), the concept of which is pictured in Figure 5.11.

Like Twin-Fingers, X-Fingers (XF) is a vision-based gesture interaction method for use with VR headsets that uses an embedded smartphone with a rear video camera. The goal of the XF design is to allow the movement of the fingers to convey two smoothly varying parameters that can represent the x and y positions of an onscreen cursor. The gesture-based parameters must be stable and reliably measurable using computer vision algorithms. More importantly, these parameters must be invariant to the movement of the camera.

5.6.1 Interaction Design

The XF gesture (Figure 5.11) involves sliding the index finger of the dominant hand over the index finger of the static non-dominant hand, in an X-shaped orientation. Vertical cursor movement corresponds to the vertical movement of the finger of the dominant hand and the horizontal cursor movement corresponds to left-right

FIGURE 5.11: The X-Fingers (XF) interaction design: Sliding the dominant index finger horizontally or vertically moves an onscreen 2D cursor.

movement of the finger of the dominant hand. Such a design allows cursor control over a custom area of a user's screen using only slight finger movements by the user's dominant hand, rendering the user free to move their head or hands about without affecting the cursor position. Moreover, users are free to use either the back of their hands or front in performing this gesture. Unfortunately, the negative space tracking feature of TF that allows consistent segmentation has been compromised in the XF design. Skin colour-based segmentation involves several challenges that would vary results across lighting conditions, skin tones and background. To aid in robust colour segmentation in XF, users are made to wear coloured finger coverings made of soft stretchable rubber, since there is no discernible gap that can be tracked like in the bare-handed TF interaction design. This was a necessary step to ensure the fingers of all users in the user study be tracked in an unbiased manner.

XF uses a simple computer vision-based algorithm to locate the tips of the two intersecting index fingers, as well as their intersection point. The algorithm firstly performs color-based image segmentation of the user's fingers (Figure 5.12(b)); then, it takes the hand's contour to get the convex hull of the hands (Figure 5.12(c)). The hull vertices are then narrowed down to the locations of the two fingertips T_1 and T_2 by taking the topmost two hull points. Their intersection is further located by calculating the intersection of the outer contours of the left and right finger, as shown in Figure 5.12(c) and Figure 5.12(d).

The lengths tracked along the fingers, L_y and L_x are calculated as the Euclidean distances of the intersection point I from T_1 and T_2 respectively, shown in Figure 5.12(c) and (d) and delineated in Equation (5.4).

FIGURE 5.12: Tracking X-Fingers (a): Image of user's crossed fingers as captured by camera; (b) Segmented fingers; (c) Convex hull of fingers in pink, detected fingertips (T_1 and T_2) and intersection point I ; (d) The varying lengths of L_x and L_y , used to calculate the x - and y -positions of the onscreen cursor.

$$L_x = \text{EuclideanDistance}(T_1, I); L_y = \text{EuclideanDistance}(T_2, I) \quad (5.4)$$

The length on the left finger, L_x , is used to compute a stable X position F_x , and the length on the right finger, L_y , is used to compute a stable Y-position F_y .

$$F_x = \frac{L_x \times w}{L_{totalx} \times S_x}; F_y = \frac{L_y \times h}{L_{totaly} \times S_y} \quad (5.5)$$

From Equation 5.5, screen coordinates of the local reference, F_x and F_y are calculated to position the onscreen cursor. L_{totalx} and L_{totaly} are the total lengths of the left and right fingers as obtained from an initial calibration procedure.

This calibration consists of a 2-second procedure by a user to detect the total lengths of the fingers, while the user performs the gesture with both fingers at the extremities. Furthermore, S_x and S_y are scale factors, also calculated from the data obtained from the calibration procedure. Lastly, w and h are the width and height of the screen space over which the cursor will move, respectively. This is analogous to the smartphone camera's field of view, which is the effective working space available to the user.

5.7 Evaluation: BF and XF

The evaluation between TF and BF had demonstrated that with some amount of practice, the design of TF was able to outperform BF in terms of task performance by users involving head movement. Since XF was designed to be a simpler gesture to master than TF while retaining its property of head variance, it was decided to have two sets of users perform different tasks in order to evaluate users on their performance without the need for evaluating learning effects for either method. Hence, users were made to perform either User Task 1 or 2 alone. Paired T-tests were employed to evaluate the significance of the results involving BF and XF, while independent-measures ANOVA was used to evaluate the significance of results between this evaluation and the previous one.

5.7.1 Experimental Design and Set-Up

User Task 1: In the first task, the user has to perform is a simple number selection task shown in Figure 5.13 (a). It essentially serves as a practice exercise that allows users to familiarize themselves with using BF and XF gestures to navigate an onscreen cursor in the VR environment. A single-digit number is displayed on screen ("Type N"), which the user should then input by navigating the red spot cursor with their fingers to the corresponding number box within the 3×3 digit keypad. The user selects the number by briefly holding the cursor within the number box for 1 second. For each interaction method, the user is required to enter the same fixed set of 10 different numbers twice. The time a user takes to navigate from one correct number to the next is recorded after every trial. For fair performance comparison, users repeat the same fixed set of 10 different numbers for both XF and BF trials.

User Task 2: In this second user task, four numbered bubbles appear around the four sides of the square keypad, as seen in Figure 5.13 (b). Users make slight head movements to select each numbered bubble, which then gets highlighted. They then "burst" the selected bubble by using their finger gestures to navigate the onscreen cursor to the same number on the 3×3 digit keypad. The time users take from selecting a bubble to navigating to its correct matching number on the keypad is recorded. This was repeated for two rounds per user, thus making the user select a total of 8 numbers using each interaction method. All measured timings for both user tasks exclude the 1-second selection dwell time.

FIGURE 5.13: Screens displayed for both tasks in the user study. (a) Numbered grid with direct selection for User Task 1; (b) Numbered grid surrounded with numbered bubbles to burst with slight head movements, for User Task 2; (c) A user performing a task using Technique S3.

All mobile applications employed in the user study were developed in Android Studio using Java and OpenCV. The computer vision algorithms used simple color-based skin segmentation to detect the fingertips and their intersections. Users were made to perform the gestures in front of a white background while wearing colored finger coverings to improve the robustness of the color segmentation algorithm under varying lighting and shadow cast conditions.

5.7.1.1 Participants and Procedure

The study was conducted with 12 participants in total whose ages range from 22 to 36 years; 6 participants performed User Task 1, while the other 6 performed User Task 2. They were university students and staff (5 female, all right-handed). Participants were first shown an introductory animated PowerPoint presentation which explained both interaction gestures and the tasks they were required to perform. To avoid sequencing bias, the order of interaction methods was counterbalanced with each participant. In the case of Technique XF, users were firstly required to undergo a 2-second calibration procedure where they performed the gesture with both fingers at the extremities. A within-subject design was employed in the experiment, with a single varying factor, the interaction technique.

5.7.1.2 Results and Discussion

The results summarised in Figure 5.14 portray the average task completion time per task for each interaction method and task type for the different sets of users. Quantitative results were computed for statistical significance using paired sample t-tests; the normality of the data was verified with Kolmogorov-Smirnov tests[134].

User Task 1 had users perform simple number entry tasks using either interaction technique. The purpose of User Task 1 was to evaluate the performance of both techniques without the involvement of head motion. The mean time for making a selection using Technique XF is ($M=1.14s$, $SD=0.47s$) and Technique BF is ($M=2.1s$, $SD=1.33s$); this difference in time was found to be significant ($p=0.01$).

Like in the previous evaluation between TF and BF, User Task 2 introduced slight head movements as users navigated the onscreen cursor towards a number they selected by moving their head (either up, down, left or right). The mean time for

FIGURE 5.14: : Mean task completion time in both interaction techniques, across 2 user task types, with Standard Error bars. Statistical significance is denoted by asterisks (*= $p < 0.05$).

making a selection using Technique XF is ($M=1.58s$, $SD=0.27s$) and Technique BF is ($M=3.33s$, $SD=1.12s$). This difference was also found to be significant ($p=0.01$). Moreover, while performing this task using BF, users found their cursor sliding backwards with every head movement as they attempted to guide the cursor to the matching number. In XF, however, they found the cursor to be stable despite their head movements whilst still keeping their fingers inside the view of the camera.

Comparing the user performance of the set of users who performed BF in the earlier study, we found that there is no significant difference in performance between both sets of users in either task, upon using an independent-measures ANOVA test ($p>0.05$). ANOVA was chosen due to the unequal participant count between both evaluations. Hence, from the results of this user study, it was found that X-Fingers is the preferred method of the three single input methods when it came to performance in tasks requiring navigation along with head motion. In contrast, while TF is a bare finger interaction method which tracks the negative space and is also head motion invariant, it does not perform as well as XF in terms of user performance and ease of use.

5.8 Conclusion

The work in this chapter investigates, designs and evaluates efficient user interaction methods that are independent of external controllers, by exploiting smartphone camera-based interaction methods using finger gesture recognition and head movement. The proposed gestures can be employed as an alternative means of controlling onscreen cursors in VR.

In summary, the chapter has introduced and evaluated the designs of three new finger-based interaction techniques, which are used to control an onscreen 2D cursor in mobile VR:

1. S1: Twin-Fingers
2. S2: Bunny-Fingers
3. S3: X-Fingers

These interaction techniques have been tracked and detected through the RGB camera of a smartphone embedded within a VR headset, using classical computer vision algorithms, which have been detailed in the chapter. Each interaction technique was designed and evaluated for their performance in 2D number selection tasks. The results of these evaluations have shown each interaction technique to have its respective advantages and disadvantages.

While a literature survey revealed a lack of head motion-invariant pointing gestures, an interaction gesture was developed that utilised the non-dominant hand of the user as a reference, on which a dominant hand performs a gesture that controls an onscreen 2D cursor in VR, thus leading to the development of interaction technique TF (Twin-Fingers).

A baseline method that employed a simpler one-handed gesture, technique BF (Bunny-Fingers) was developed to evaluate the effectiveness of TF. The evaluation showed that while BF was the technique of choice for number-selection tasks with and without head movement as an embedded interaction modality within the task; this was found to be due to the steeper learning curve that TF entails. As a result, a subset of the users in the primary evaluation obtained further limited practice

on both methods, after which it was found that TF outperformed BF in terms of task performance time.

Hence, this evaluation demonstrated the advantage of a head-motion invariant bare-finger interaction technique to control a cursor when slight head movements are involved in the task, when compared to a traditional absolute onscreen 2D cursor such as that of BF.

Taking the disadvantage of TF's rather complex and non-conventional gesture into account, a simpler version called X-Fingers (XF) was designed, which retained TF's primary advantage of head motion-invariance. However, since XF did not use gap tracking like TF, users were made to wear coloured finger coverings for robust tracking across all users and lighting conditions. The design of XF was detailed, and it was evaluated against the baseline technique BF with the same tasks as in the previous evaluation. This time, it was found that users performed better in both tasks using XF, compared to BF. These findings strongly suggest that XF is a viable head motion-invariant interaction design that is usable as well as effective, and is preferred over traditional one-handed input techniques such as BF in 2D onscreen cursor control tasks.

The next chapter looks at multi-input interactions that extend some of the interaction methods presented in this chapter.

Chapter 6

Multi-Input Interactions in VR Headsets

6.1 Introduction

The previous chapter looked at the various methods through which users can provide direct input to a VR headset using different finger gestures. X-Fingers provided a continuously moving onscreen 2D cursor for VR interactions. However, it only provides control to a single 2D cursor in VR.

The focus of this chapter is to devise interaction designs that control two 2D input spaces. This is motivated by our goal of realizing popular arcade games that plays with two joysticks console (e.g. Xbox-360 and PS3) on a low-cost console-free VR headset. The stereoscopic affordance of VR headsets provides opportunity for VR game designers to introduce interesting depth dimension to such games. Moreover, there are also examples of VR applications that require two or more 2D inputs to be controlled by the user in performing compound tasks, such as simultaneously manipulating and positioning objects [9]. This is generally achieved by the use of compound or multiple external controllers. This work, in contrast, presents interaction designs that employ the camera of the embedded smartphone alone to detect finger, arm and head movement of the user in order to control two 2D spatial input dimensions. This is shown in Figure 6.1.

FIGURE 6.1: Multi-input multimodal interactions explored in this chapter, all using a smartphone’s RGB camera.

The designs of such multi-input interaction methods need to be sensitive to the context of the interaction task at hand. They also require usable coordination between all inputs when they are to be employed simultaneously. This chapter explores the designs of multi-input interaction designs in terms of task context and coordination level between modalities or inputs. In short, the chapter evaluates the effects of the following with respect to the novel multi-input interaction designs proposed:

- Symmetry of bimanual hand gestures *on* task assignment symmetry
- Physical coupling of interaction modalities *on* user’s coordination between inputs.¹²

The chapter is organized as follows:

- Section 6.2 surveys recent literature on multi-input and multimodal interactions in VR.
- Section 6.3 evaluates task assignment and gesture symmetry, and evaluates their effects on user task performance.

¹A summary of this section is given in a short video at <http://bit.ly/MultimodalVR>

²The research presented in this chapter has been published in [3].

- Section 6.4 evaluates the effects of physically coupled and decoupled multimodal interaction designs on user performance in asymmetric multi-input tasks requiring increasing levels of coordination.
- Section 6.5 outlines and evaluates a gameplay experience designed to utilize multiple and simultaneous spatial controls, to demonstrate the playability and usefulness of the methods proposed.
- Section 6.6 concludes the chapter by summarizing the novel interaction designs proposed, and their effects on task assignment type, modality coupling and coordination levels in VR tasks.

6.2 Related Work

While single-input interactions are the norm in VR HMDs, interactions that combine different modalities together - i.e. multimodal interactions - are seen as more effective methods to enhance user experience. Multimodal interaction in VR simultaneously employ the behavior of various parts of the user's body such as the eye gaze, head rotation, hand movement and speech to interact with VR HMDs. The combination of such disparate modalities together make for a more natural user experience in VR. It has also been observed that users have a strong preference to interact multimodally rather than unimodally, especially in spatial application domains [10].

There have been comparative evaluations between several interaction modalities in VR as well as AR. However, research that investigate the combinations of various interaction modalities for user tasks in VR is comparatively less. The work by [135] combines speech output to mid-air hand gestures in AR in tasks where users control a virtual dog. A combination of both modalities was found to be effective; however, the hand gesture required the use of a LeapMotion controller, and thus requires a custom-made HMD device. The VR interface designed by Kok et al [136] is a multimodal interface to VTK on a virtual environment, in which head tracking is used for camera control, pedals for clutching and speech input for system control. However, the system relies heavily on external controls that the user controls individually, in addition to a head-tracked VR HMD, involving pedals, a microphone and external cameras. The system developed by Tsai et al [137] further integrates

multimodal interaction sensors (smartphone, Kinect v2, and Myo armband) and streaming technology in VR, which again employs the use of peripheral sensors and controllers. Pinpointing is a technique developed in the work by Kytö, et al.[84] which combines head motion and eye gaze direction to facilitate accurate object selection in AR; this is again dependent on a Pupil Labs eye tracker attached to Microsoft HoloLens. A system that also employs the combination of head gaze with another input modality in VR is WearTrack[138], which combines a source-less head orientation tracker with a head-worn tracking device that tracks hand gestures. This requires a wireless 3DOF ring pointer, as well as the portable head tracker, to be worn by the user.

By contrast, this chapter proposes multi-input multimodal interactions requiring the use of a single RGB camera of an embedded smartphone in a VR headset. While previous work for multi-input interactions in VR has talked about asymmetric interactions involving handheld controllers in conjunction with one hand[139], this section investigates human gesture-based interaction methods for multi-input systems. The multimodal interaction designs proposed involve the novel combinations of the physically-decoupled head+finger movement, and the physically-coupled arm+finger movement on mobile VR.

6.3 Mapping Task and Physical Interaction Design in Bimanual Multi-Input Interactions

When designing for multi-input interactions, one has to be mindful of the nature of the physical and cognitive interdependency of the multiple tasks being done by the user. In particular, multi-input positioning tasks can vary in terms of spatial ranges and coordination between inputs. This section looks at multi-input interaction designs that cater to symmetric and asymmetric task assignments. The work by Casalta, Guiard et al [140] talks about a symmetric and asymmetric division of labour for bimanual interactions based on the task given to each hand. The symmetry of task assignment in this case refers to the spatial positioning ranges given to each human input element. If both inputs are given the same spatial range, the task assignment to each input is said to be symmetric. Otherwise, it is said to be asymmetric. Besides the symmetry in the task assignment, the research here also

explores the effect of symmetry in the bimanual designs, and how it relates to such tasks.

FIGURE 6.2: (a)ArmX: Bimanual non-identical interaction (b) Double Bunny Fingers (DBF): Bimanual identical interaction

This section proposes two bimanual (involving both hands) interaction methods in VR for two-input object positioning tasks. This is shown in Figure 6.2. The first method, ArmX, is a *non-identical* input approach where Input 1 is controlled by finger movement, and Input 2 is controlled by the movement of the user's two arms in tandem. The second method, Double Bunny Fingers (DBF), is an *identical* approach where Input 1 is controlled by the unconnected dominant hand alone, and Input 2 is controlled by the unconnected non-dominant hand alone (both inputs are performed through similar gestures).

The major difference in these two multi-input bimanual interaction designs is the similarity or symmetry of the bimanual gestures and the way the two input interactions are executed. A user evaluation is done to study the effects of the physical designs of these two interaction methods on tasks of varying symmetry and coordination levels. The interaction methods, evaluation, results and their implications are discussed in the following subsections.

6.3.1 Interaction Design M1: ArmX

An interesting observation from exploring single-input interaction designs was that the 2D position inputs elicited from both Twin-Fingers (TF) and X-Fingers (XF) interactions are invariant to the movements of the user's arms and head; since XF is the superior method, by this very virtue, an additional dimension of interactive

input control can be obtained if the users keep the non-dominant left hand stationary and only slides the index finger of the dominant right hand (or vice versa for left-handed users). Tracking the position of the tip of the non-dominant hand provides this additional dimension of 2D spatial control.

The 2D spatial coordinates from the X-Fingers interaction is measured with respect to the relative position of the sliding index finger of the dominant hand against the static index finger of the non-dominant hand. If placement of the crossed index fingers is maintained, then changes in the user's arm position will not affect the local 2D coordinates of X-Fingers. This spatial invariance property of the X-Fingers design enables it to be combined with an additional interaction modality.

FIGURE 6.3: ArmX: Bimanual non-identical spatial position control with finger (Input 1) and arm (Input 2) movement.

This property of hand movement invariance of the X-Fingers gesture was exploited to provide an additional input modality in VR. ArmX is an interaction design we propose that separates the user's two hands as a separate entity that provides a modality of interaction that is different from the one provided by X-Fingers. This is tracked by computing the tip of the non-dominant finger and mapping it to a cursor onscreen, as shown in Equation (6.1) and Figure 6.3.

$$A_x = \frac{T_2x \times w}{C_w}; A_y = \frac{T_2y \times h}{C_h} \quad (6.1)$$

Equation (6.1) calculates the screen coordinates of the global reference, A_x and A_y to position the onscreen cursor when the user moves their arms together. This is irrespective of the position of the local finger-based coordinates F_x and F_y , since

it takes into account the absolute position of the non-dominant fingertip T_2x and T_2y (see Figure 6.3). C_w and C_h are the width and height of the frame captured by the camera respectively, while w and h are the width and height of the screen space over which the cursor will move.

Hence in ArmX, while the movement of the dominant index finger over the non-dominant index finger dictates the onscreen position of the finger-based cursor, moving the user's two arms together as a single entity while maintaining their relative finger position in XF controls a second, unconnected onscreen cursor. The property of hand movement invariance of the X-Fingers gesture was exploited to provide an additional input modality in VR.

6.3.2 Interaction Design M2: Double Bunny-Fingers

Bunny-Fingers (S2) was developed primarily to incorporate the simplicity of a one-handed gesture. In a similar vein, we devised a double-handed version of the S2 gesture that also employs the detection principle used in TwinFingers (S1), namely the detection of gaps between the fingers. We called this interaction method Double Bunny-Fingers (M2)(DBF) due to the bunny-like shape the fingers from both hands form, while encircling green circular discs between their thumb and forefinger. It is a simple method that uses the gap created when users pinch their index finger and thumb together as shown in Figure 6.4. The vision algorithm tracks the inner tips of the green-coloured disc in each hand where the two fingers pinch each other and maps its 2D position to the onscreen cursor position.

$$D_1x = \frac{I_1x \times w}{\frac{C_w}{2}}; D_1y = \frac{I_1y \times h}{C_h} \quad (6.2)$$

$$D_2x = \frac{I_2x \times w}{\frac{C_w}{2}}; D_2y = \frac{I_2y \times h}{C_h} \quad (6.3)$$

Equations (6.2) and (6.3) calculate the screen coordinates of both inputs for DBF, Input 1: D_1x and D_1y , and Input 2: D_2x and D_2y to position the onscreen cursor

when the user moves each hand encircling each disc. It takes into account the absolute position of the rightmost and leftmost tips of each disc I_2x and I_2y . C_w and C_h are the width and height of the frame captured by the camera respectively, while w and h are the width and height of the screen space over which the cursor will move.

FIGURE 6.4: Double Bunny Fingers (DBF): Bimanual identical interaction gesture with two inputs.

The DBF interaction design works much like two identical bimanual gestures whereby one hand controls one cursor, and the other hand, the second cursor. When the interaction design requires cursors with differing ranges, the dominant hand is assigned control of the global cursor, while the non-dominant hand is assigned control of the local cursor. Hence, DBF is referred to as an identical bimanual interaction method.

With the interaction techniques M1 and M2, we were interested to know the effects of task symmetry, coordination level and interaction technique on task completion time performance. The techniques in question are both multi-input; however, M1 (ArmX) employs the disparate body parts of hand and fingers as connected inputs (i.e. non-identical bimanual design), whereas M2 involved the two separate arms of a single user as unconnected but identical inputs (i.e. identical bimanual design). The evaluation of these two methods has been outlined in the following sections.

6.3.2.1 Experimental Set-Up and Design

After designing the two sets of bimanual gesture interaction techniques (M1 and M2), we were interested to investigate the effects of task symmetry, coordination level as well as technique on user task performance. Task symmetry in this context refers to the spatial ranges of the objects to position with each input. A user study was conducted where users were evaluated for their speed in completing an asymmetrically distributed task as well as a symmetrically distributed task, both with increasing coordination levels, using each of the two interaction designs.

We employed a within-subject design with 3 factors:

1. Interaction technique (M1, M2)
2. Task assignment type (Asymmetric and Symmetric)
3. Coordination level (L1 and L2)

1. Interaction Technique

Both interaction techniques M1 and M3 are bimanual and multi-input; however, in the non-identical bimanual technique M1 (ArmX), Input 1 is controlled by moving the index finger of the dominant hand over the index finger of the non-dominant hand, and Input 2 is controlled by moving the user's arms together (Figure 6.3). In the identical bimanual technique M2 (DBF), moving the user's dominant hand controls Input 1, while moving their non-dominant hand controls Input 2, as in Figure 6.4.

2. Task Assignment Type

The tasks, which were essentially basic 2D object positioning tasks, were further divided into two types based on the individual task assignment for both Input 1 and Input 2 for both bimanual gestures. This is shown in Figure 6.5.

The first task assignment type was asymmetric, which meant that each input control of the multi-input interaction technique was assigned a different task. The difference between the two object positioning tasks lies in the range provided to position each object; thus in the asymmetric task type, Input 1

FIGURE 6.5: Task Assignment Types. (a) Spatially asymmetric positioning task with a shaded circle nested inside a large shaded square. (b) Spatially symmetric positioning task with two disparate shaded shapes of the same size.

would control a small circle nested within a large square, while Input 2 would move the large square around the screen.

In the symmetric task, both inputs control two separate, unconnected shapes of similar dimensions, sharing the same range. Hence, in this task, Input 1 controls a large circle around the screen, and Input 2 controls a large square around the screen. These tasks were designed as such to investigate the effects of task mapping to physical gesture mapping, in particular the similarity of the task assignments and similarity in the bimanual gesture design.

3. Coordination Level

The tasks were designed with increasing degrees of coordination required between each of the input modalities as the user progresses across the three different task levels, as shown in Figure 11. Each of the tasks is detailed as follows and shown in Figure 6.6:

L1 - Low Level Coordination: In this user task, both the square and circular target outline remain static onscreen. Users complete the task by positioning both shapes inside their respective outline, in any order of their choice. Each subtask is considered complete if the entire shaded shapes reside within their respective target outline. This applies for both asymmetric and symmetric task assignments.

FIGURE 6.6: Coordination levels for each task assignment type, induced by dynamic outlines.

L2 - High Level Coordination: In this user task, the square target outline slowly moves back and forth across the screen, while the circular outline remains static within the shaded square. Users complete the task by fitting the small circle into the static circular target outline while keeping the big shaded square within the slowly moving square target outline continuously for a duration of 2 seconds. This also applies for both asymmetric and symmetric task assignments.

The purpose of varying the levels of coordination in the user study is to understand how the level of task coordination between the two inputs can affect the performance of the non-identical and identical bimanual gesture interactions.

6.3.3 Participants and Procedure

Ethical approval was obtained before the commencement of all user studies in this chapter (IRB-2018-05-043). The study was conducted with 11 participants (6 female, all right-handed). Participants were first shown an introductory animated PowerPoint presentation which explained both interaction gestures and the tasks they were required to perform. To avoid sequencing bias, the order of interaction

FIGURE 6.7: A participant performing the multi-input tasks using (a)ArmX interaction method, and (b)DBF interaction method.

techniques as well as task assignment type were counterbalanced with each participant. After the instructions, users proceeded to have a trial session of the tasks using Technique 1, followed by the actual session using Technique 1. Within these tasks, the order of task assignment type was counterbalanced across all participants.

After a short break, users had a trial session of both task assignment types using Technique 2, followed by the actual session using Technique 2. Each session consisted of ten tasks altogether, with five tasks from each level. The levels always appeared in order - i.e. a user performed five tasks from L1, followed by five tasks from L2. Altogether, users performed $5 \text{ tasks per level} \times 2 \text{ coordination levels} \times 2 \text{ task assignment types} \times 2 \text{ techniques} = 40 \text{ tasks}$.

Users were asked to complete a questionnaire at the end the sessions, where they were asked to compare their preferences for which interaction technique in each of the two different coordination levels and task assignment types. The entire procedure took about twenty minutes. A picture of the environment in which the user study was conducted is shown in Figure 6.7.

FIGURE 6.8: Mean task completion time in both interaction techniques, across 2 user task types and 2 coordination levels, with Standard Error bars. Statistical significance is denoted by asterisks (*= $p < 0.05$; **= $p < 0.01$).

6.3.4 Results and Discussion

For each interaction technique, we were interested in knowing how users performed both asymmetric and symmetric tasks that required increasing levels of coordination. Therefore, during the study, participants were evaluated on the basis of the time taken to complete each task they were given, for a total of 40 tasks per person. After the study, users were also asked to provide subjective feedback on their perceptions of both interaction techniques. Quantitative results were computed for statistical significance using paired sample t-tests; the normality of the data was verified with Kolmogorov-Smirnov tests[134].

Figure 6.8 shows the results of the evaluation based on the factors tested for using both interaction techniques. All differences were also found to be significant ($p < 0.05$). This leads to several interesting observations.

Firstly, it is observed that when the input control is asymmetrically distributed to either hand, users are seen to perform significantly faster (p -values for L1=0.02; L2=0.001) using the technique that employs a non-identical bimanual gesture design such as ArmX (L1: $M=3.52s$, $SD=1.02s$; L2: $M=4.79s$, $SD=1.87s$), for both levels of coordination. This is in contrast to their performance while using an identical bimanual gesture design, DBF (L1: $M=4.79s$, $SD=1.87s$; L2: $M=13.74s$,

SD=5.06s) While increasing levels of coordination lead to significantly increasing task performance time for both techniques, users still perform better when employing M1 as their interaction technique. This suggests that when the task type is different (or asymmetric) for each input, it is easier and more natural for users to map these tasks to different forms of input control, such as sliding fingers for Input 1, and moving arms for Input 2 (as in ArmX). In addition, the assignment of the smaller movement of the finger for Input 1 to the smaller movement of the local cursor control and the gross arm movement of Input 2 to the large global cursor movement provided a cognitively more coherent input gesture-to-task mapping scheme.

However, this trend is seen to reverse in the second task assignment type, where the task assignment is the same for both input controls. In the symmetric tasks, users are seen to perform significantly faster (p-values for L1<0.001; L2=0.001) using the technique that employs an identical gesture such as DBF, for both levels of coordination. Similar to the previous trend, while increasing levels of coordination lead to significantly increasing task performance time for both techniques, users performed better when employing DBF (L1: M=4.06s, SD=0.74s; L2: M=7.09s, SD=3.70s) as their interaction technique in both levels. In contrast, they performed symmetric tasks poorer with ArmX (L1: M=9.90s, SD=2.64s; L2: M=4.06s, SD=0.74s). These findings again lend credence to the importance of providing a coherent mapping of the spatial nature of the the two coordinated tasks to that of the bimanual gesture movement and design. In other words, the symmetric nature of the task distribution could be more speedily performed with an identical bimanual gesture of DBF that shares similar spatial movement constructs for both hands than that the non-identical bimanual design of ArmX, where the spatial movement of the fingers and arms differ substantially.

A post-evaluation questionnaire was conducted when users rated their preference for each interaction method for each task type(Figure 6.9). For the asymmetric task, 8 out of 11 participants felt ArmX was easier to use than DBF. Participant P10 commented, *"It was quick and seamless to control the shapes using the (ArmX) technique than the (DBF) technique."* P4 noted that *"For (asymmetric) task, (ArmX) method felt easier to coordinate."* Confusion of task assignment to both hands in the DBF method also played a part in their poorer performance in the asymmetric task with DBF. P8 noted that *"Using the (DBF) method I tend to move*

FIGURE 6.9: User preferences of technique for asymmetric and symmetric task types, from 11 participants.

both hands simultaneously which sometimes becomes confusing." P9 commented, *"There was difficulty distinguishing between the tasks for both hands with DBF"*. There was a minority who did prefer DBF for asymmetric tasks however. For example, P1 felt that *"In DBF, both hands were free of each other and could do their assigned tasks more easily, without bothering about coordination of both hand movements"*. This comment seems to suggest that a minority of users find the physical coupling of fingers and arm in the bimanual gesture design to be restrictive and difficult to coordinate. This could be due to the novelty of the ArmX gesture, since a majority of users still performed better and preferred ArmX for asymmetric tasks, due its more coherent asymmetric task-to-gesture mapping.

In the symmetric task assignment, 7 out of 11 users chose DBF over ArmX as the easier method to use. P4 noted that *"For (symmetric) task, DBF method was easier to coordinate"*. P10 commented that *"I think it was easy for me to register the tasks in DBF method since we had separate tasks for each hand to move them around and match the outlines to the shapes."* P2 noted: *"(ArmX) was more difficult when both shapes were moving in the (symmetric) task"*. P10 said: *"In case of (ArmX), I got confused and mixed up the shapes and the fingers"*. Such comments suggest that the task to input gesture mapping needs to be designed in a coherent manner to avoid confusion, which can subsequently lead to poor performance.

The quantitative results as well as subjective findings from the post-survey questionnaire has provided clear evidence that the mapping coherency of the VR task requirements to the way the bimanual input gestures are designed can strongly influence the performance and preference of the user interaction.

6.3.5 Summary

Given the possibility of two different multi-input bimanual interaction designs, we set out to study how the symmetry of bimanual hand gesture design affects user task performance when the tasks are differently assigned, and of increasing coordination levels. Our results show that when the task assignment is asymmetric, users perform tasks better using the non-identical ArmX gesture, while when the task assignment is symmetric, they perform tasks better using the identical gesture DBF. This leads one to conclude that the design of the physical interaction method has an influence on the task design.

In summary, these findings lead to the conclusion that a non-identical gesture design, such as in ArmX, is recommended when the task assignment is asymmetric, while an identical gesture design, such as in DBF, is recommended when the task assignment is symmetric in spatial positioning tasks in VR.

The following section looks at multimodality in asymmetric multi-input interactions in VR.

6.4 Multimodality in Hand Gestures: Exploring Physically Coupled And Decoupled Modalities

6.4.1 Introduction

The 2D spatial coordinates from the X-Fingers interaction is measured with respect to the relative position of the sliding index finger of the dominant hand against the static index finger of the non-dominant hand. If placement of the cross index

fingers are maintained, then changes in the user's arm position will not affect the local 2D coordinates of X-Fingers. This is similarly true of the movement of the built-in camera of the VR headset if the user moves his or her head. This spatial invariance property of the X-Fingers design enables it to be combined with an additional interaction modality, which could either be the user's arm movement or head movement. This give rise to two multimodal interaction designs, ArmX (M1) and HeadX (M3). ArmX has been detailed in the previous section, and HeadX is detailed in the following sections.

FIGURE 6.10: (a)ArmX: Multimodal spatial position control with finger and arm movement (physically-coupled); (b) HeadX: Multimodal spatial position control with finger and head movement (physically-decoupled).

Unlike the X-Fingers 2D spatial input, users can "control" the 2D position of fingertip by either moving both their arms, as shown in Figure 6.10(a) or moving their head, as shown in Figure 6.10(b). We called the former a physically-coupled (P-C) multimodal interaction, since the arm and the fingers are physically connected when the user performs both dimensions of input control. The latter is a physically-decoupled (P-D) multimodal interaction as users moves the finger and their head independently.

This interaction design scenario gives rise to several interesting research questions that are investigated in the following sections. Firstly, do physically-coupled and physically-decoupled multimodal interaction designs influence users' performances and preferences? Secondly, how do the different degrees of coordination between these co-interactions of different input modalities influence the users' performance?

6.4.2 Interaction Design M3: HeadX

While interacting with VR headsets, it is natural for users to move their heads, thus changing the viewpoint of the embedded smartphone's rear camera. Since most cursor controls rely on absolute hand positions, moving a user's head around will affect the absolute position of the hand as well, thus affecting the cursor position. In contrast, X-Fingers (XF) does not suffer this disadvantage due to its design, as a result of which this interaction method is invariant to head motion. Because XF relies on the relative position of the index finger from each hand with respect to each other to determine the local 2D reference position onscreen, changing the smartphone camera's viewpoint with head movement does not affect the computed intersection point of the fingers.

This property of invariance to head motion was utilized to design an additional input modality on top of X-Fingers. The second interaction design is called HeadX, where the movement of the HMD attached to the user's head act as another 2D spatial input control. Unlike traditional head gaze interaction methods in HMDs, this method uses the changing camera viewpoint to detect the absolute location of the tip of the non-dominant finger using vision-based techniques detailed above. Hence, slight head movements by the user, while keeping the user's hands in view, control the global 2D spatial position, while the relative movement of fingers in X-Fingers controls local 2D spatial position, as in the previous interaction design.

$$H_x = \frac{(C_w - T_2x) \times w}{C_w}; H_y = \frac{(C_h - T_2y) \times h}{C_h} \quad (6.4)$$

Similar to Equation(6.1), Equation (6.4) calculates the screen coordinates of the global reference, H_x and H_y to position the onscreen cursor when the user moves their head while wearing the HMD, thus changing the camera viewpoint. This is also irrespective of the position of the local cursor, since it again takes into account the (inverted) absolute position of the non-dominant fingertip T_2x and T_2y . C_w and C_h are the width and height of the frame captured by the camera respectively, while w and h are the width and height of the screen space over which the cursor will move.

6.4.3 Evaluation

After designing the two sets of multimodal interaction techniques, we were interested to investigate which of the two multimodal interaction designs lead to better task performance. A user study was conducted where users were evaluated for their speed in completing the given tasks using each of the two interaction designs. The tasks involved two different forms of 2D spatial positioning controls. The local spatial control was assigned to the X-Fingers interaction and the global spatial control was performed using either the arms or the head. These tasks were designed as basic 2D object positioning tasks.

6.4.3.1 User Task

FIGURE 6.11: The three user tasks (L1, L2 and L3) with increasing levels of coordination between the two interaction modalities. Increasing coordination is achieved through the introduction of dynamic target outlines that the user needs to track during interaction.

In all the tasks, the global 2D spatial controls the position of a large shaded square, while the local 2D spatial controls the position of a small shaded circle nested within the shaded square. While the local 2D spatial position is controlled by X-Fingers for both interaction methods, the global 2D position of the large shaded square is controlled by either the user's arm movement in the ArmX technique or the user's head movement in HeadX technique. The underlying aim of every task is to place the large square inside a square outline within the screen space, and the circle inside a circular outline nested within the large shaded square. The tasks were designed with increasing degrees of coordination required between each of the

input modalities as the user progresses across the three different task levels, as shown in Figure 6.11. Each of the tasks is detailed as follows:

L1 - Low Level Coordination: In this user task, both the square and circular target outline remain static onscreen. Users complete the task by positioning both shapes inside their respective outline, in any order of their choice. Each subtask is considered complete if the entire shaded shapes reside within their respective target outline.

L2 - Medium Level Coordination: In this user task, the square target outline slowly moves back and forth across the screen, while the circular outline remains static within the shaded square. Users complete the task by fitting the small circle into the static circular target outline while keeping the big shaded square within the slowly moving square target outline continuously for a duration of 2 seconds.

L3 - High Level Coordination: In this final user task, the square target outline moves across the screen as in L2. Additionally, the small circular target outline also moves back and forth within the large shaded square. Users have to not only place the colored shapes inside their respective outlines, they also have to continuously ensure that the shaded shapes stay within their respective target outlines for 2 seconds.

The L1 task has no dynamic targets. As such, users are likely to complete each of the subtask one after the other. There is therefore little need for coordination between the two different input modalities. The L2 task has one static and one dynamic target. Since the static circular target requires local 2D spatial control with X-Fingers, users are likely to perform this task first and then hold this position steady by keeping their fingers in place while they use the other 2D spatial input control modality to move the shaded square into the moving target and maintain synchronized motion with the arms (ArmX) or head (HeadX) movements. This has introduced some need for coordination between the two input modalities through the need to maintain one modality of input control steady while the other input control is changing. The task L3 requires the concurrent maintenance of two shapes within the two target outlines moving with different speeds and directions. This task requires a high degree of simultaneous coordination between the two different input modalities. The goal of this user study is to investigate the effects of these varying levels of input modality coordination on user task performance

for the ArmX and HeadX multimodal interaction designs. This experiment is designed to obtain comparative quantitative performance measure in terms of speed in completing the task for each of these techniques.

6.4.3.2 Experimental Set-up and Design

We employed a within-subject design with 2 factors:

1. Interaction technique (ArmX, HeadX)
2. Task Level (L1, L2, L3)

For each of the two interaction techniques, participants completed five tasks from each of the three coordination levels. As a result, each participant is required to do 30 trials each (2 techniques $\times$ 3 levels $\times$ 5 tasks). To perform a comparative analysis of the interaction designs, we collected the time taken for each participant to complete each task. The user study was conducted in a quiet room against a white background. To minimize variations of hand across participants, they were made to wear colored finger coverings over their two index fingers, as shown in Figure 10(a). For each interaction technique, participants completed five tasks

FIGURE 6.12: (a) Participant performing the user study; (b) Sample interface of the application with the local cursor controlling the blue circle, and global cursor controlling the purple square. The boundary is denoted by the blue box; dimensions are C_h and C_w .

from each level, for a total of 2 techniques $\times$ 3 levels $\times$ 5 tasks = 30 trials each. To analyze the interaction design, we collected the time taken for each participant to complete each task. The user study was conducted in a quiet room against a

white background. To minimize variations of skin tone across participants, they were made to wear colored finger coverings over their two index fingers as shown in Figure 6.12(a).

All mobile applications employed in the user study were developed in Android Studio using Java and OpenCV. The computer vision algorithms used simple color-based skin segmentation to detect the fingertips and their intersections. Users were made to perform the gestures in front of a white background while wearing colored finger coverings to improve the robustness of the color segmentation algorithm under varying lighting and shadow cast conditions. The applications ran on an Android smartphone running Android 7.1.1 Nougat, with a rear camera (OnePlus 5, Qualcomm Snapdragon 835 Octa-core 2.45GHz, 16MP (IMX398), f/1.7, 1.12-micron pixels, EIS Dual LED flash, 4K 30 fps / 720p 120 fps video), and the VR headset used was a regular plastic headset made by SilverTec [132][133].

6.4.3.3 Participants and Procedure

Ethical approval was obtained before the commencement of the user study. The study was conducted with 13 participants (5 females, mean age 28 years, $SD = 7$, all right-handed). Participants were first shown an introductory animated PowerPoint presentation which explained both interaction gestures and the tasks they were required to perform. To avoid sequencing bias, the order of interaction techniques was counterbalanced with each participant. Firstly, users were required to undergo a 2-second calibration procedure where they performed the gesture with both fingers at the extremities. After this, users proceeded to have a trial session of the tasks using Technique 1, followed by the actual session using Technique 1. After a short break, users had a trial session of the tasks using Technique 2, followed by the actual session using Technique 2. Each session consisted of fifteen tasks altogether, with five tasks from each level. The levels always appeared in order - i.e. a user performed five tasks from L1, followed by five tasks from L2, followed by five tasks from L3 for each technique.

Users were asked to complete a questionnaire at the end of the sessions, where they were asked to compare their preferences for which interaction technique in each of the three different coordination levels. The entire procedure took about twenty

minutes. A picture of the environment in which the user study was conducted and a screenshot of what is seen by the user during the study is shown in Figure 6.12(b).

6.4.3.4 Results

For each interaction technique, the evaluation attempted to discover how users performed tasks that required increasing levels of coordination. Therefore, during the study, participants were evaluated on the basis of the time taken to complete each task they were given, for a total of 30 tasks per person. After the study, users were also asked to provide subjective feedback on their perceptions of both interaction techniques. Quantitative results were computed for statistical significance using paired sample t-tests; the normality of the data was verified with Kolmogorov-Smirnov tests[134].

Quantitative Performance

FIGURE 6.13: Mean task completion time in both interaction techniques, across 3 different levels, with Standard Error bars. Statistical significance is denoted by asterisks (**= $p < 0.01$).

Users performed five tasks for each interaction technique for every coordination level (L1, L2, and L3). The mean task completion time for each level is shown in Figure 6.13.

L1: In this level, where both target outlines are non-moving, and requires the least amount of coordination to complete, it was found that users took slightly less time to complete each task using HeadX (M=4.61s, SD=1.4s) than using ArmX (M=4.83s, SD=2.41s). This difference, however, was not a significant one ($p=0.74$), since there were some users who managed to complete the task faster using ArmX than HeadX. This signifies that both interactions take a similar amount of time to perform separately when the task requires little to no coordination.

L2: In this level, users performed five tasks per interaction technique; this time, however, the large square target outline moved across the screen slowly, and users were required to fit their shaded square inside the moving target outline continuously for 2 seconds. In this medium coordination level, users took less time to perform each task using HeadX (M=6.14s, SD=1.93s) than ArmX (M=10.25s, SD=5.06s). This difference was proven to be significant ($p=0.004$).

L3: In this level, users performed five tasks per interaction technique; this time, both the large square target outline as well as the nested circular target outline moved across the screen together, and users were required to fit both shaded shapes into their respective moving outlines continuously for 2 seconds. In this level as well, users took less time to perform each task using HeadX (M=9.67s, SD=4.15s) than ArmX (M=20.02s, SD=12.99s). This difference was also shown to be significant ($p=0.004$).

To avoid ordering bias, the order of technique used was alternated with every user. It was found that evaluating groups of users separately according to the order in which they performed each technique led to no significant differences in the results. This suggests that there are no significant learning effects influencing the users' performances between one session and the next.

User Survey

After the user study, participants were asked to fill in a questionnaire about their perceptions of their performance and preferences for the two multimodal interaction techniques coordination levels. The resulting 5-point Likert scores and computed statistical significance based on the paired sample t-test are summarized in Figure 6.14. The three aspects they were asked to rate were fatigue, the ability of users to differentiate the task given to each modality and the ease of control for each

technique. Overall, users reported a low level of fatigue altogether after completing all tasks on a Likert scale where 1 signified minimum fatigue and 5, maximum fatigue. ($M=2.36$, $SD=1.06$). This shows that users found both techniques reasonably comfortable to use.

FIGURE 6.14: Subjective feedback from participants: User perceptions on various aspects of each technique on a 5-point Likert scale with Standard Error bars. Statistical significance is denoted by asterisks (**= $p<0.01$)

The next few questions attempted to compare the participants' perceptions of various factors pertaining to both techniques such as fatigue, coordination and ease of use, and hence users were asked to rate both techniques comparatively (Figure 6.14). When it came to comparing fatigue levels between the two techniques, users reported that ArmX ($M=3.15$, $SD=1.16$) was significantly more tiring than HeadX ($M=1.77$, $SD=0.8$; $p=0.002$).

In the second question, users were asked to rate their agreement towards their ability to distinguish between the task given to either modality in both cases, on a Likert scale from 1 to 5. While users rated HeadX to be slightly easier to distinguish ($M=4.69$, $SD=0.6$) than ArmX ($M=4.23$, $SD=0.8$), this difference was not significant. This lack of a major difference in user preferences and the high mean scores showed that users had no trouble distinguishing between different modalities when performing the tasks.

The final question asked participants to rate the two techniques on their ease of control, with 1 being extremely easy to control and 5 being extremely hard to control. Users rated HeadX ($M=2.54$, $SD=1.15$) to be much easier to control than ArmX ($M=4.23$, $SD=1.12$). This difference was found to be significant ($p=0.017$).

6.4.4 Discussion

This study investigates the differences in performance and preference between physically-coupled (ArmX) and physically-decoupled (HeadX) multimodal interaction designs when performing coordinated spatial control tasks. The proposed coordinated multimodal interaction requires the users to combine the control of their index finger with either the movement of their two arms (ArmX) or their head (HeadX) to complete the tasks. Results from the questionnaire indicate that users in general do not have issues assigning the relevant control tasks to each of the modality of interaction. This is true for both P-C and P-D multimodal interaction designs, as there is no significant differences between them ($p=0.082$), suggesting both these coordinated interaction designs with different spatial control assigned to different modality do not lead to confusion among users.

FIGURE 6.15: Subjective feedback from participants: User preferences of technique for each level.

However, when it came to the issue of ease of control, the physical decoupling of the two coordinated input modalities does make a significant difference to the user. Users rated the P-D HeadX technique to be much easier to control than the

P-C ArmX technique ($p=0.017$). This preference for physical decoupling was also observed to be dependent on the level of coordination between modalities required to accomplish the task. When the level of coordination between the two modalities is low (i.e. task L1), it was observed that users tend to use each modality in turn to perform the required task in a sequential manner. They usually completed the local spatial positioning of the circular target using their finger interaction before moving on to the global positioning of the square target using either their arm or head movement. With the L1 task, 62% of the users indicated that they preferred the physically-decoupled HeadX technique; see Figure 6.15. However, when it came to their quantitative speed performance, their mean task completion time with HeadX is only slightly less than ArmX, and there was no significant difference between the P-C and P-D multimodal interactions ($p=0.74$). As the level of coordination starts to increase and users have to perform dynamic tracking with one (task L2) or both (task L3) of the interaction modalities, the preference for HeadX increases to above 84%. With both these tasks (L2 and L3), the mean task completion time for HeadX is significantly lower than ArmX ($p=0.004$). The fact that users clearly performed more effectively with HeadX in the medium and high coordination levels indicates that the advantage of a physically-decoupled interaction design increases with the increasing need to coordinate between the different input modalities. This observation is also borne out in the increasing user preference for HeadX with users making comments such as: *"It was easier to do one task per body part."* and *"I had more difficulty with coordination while using the (ArmX) interaction method."*

Some users who did prefer ArmX for task L1 commented that *"It was easier with hand when both the square and circle were still."* Others commented that physical decoupling of the coordinated interaction took some initial adjustment, *"It took me some time to figure out the amount of movement required for the head in moving the square. Moving and controlling the square by hands was easier."* However, the majority of the users surveyed still preferred HeadX, with several saying, *"It was easier to do HeadX simultaneously than ArmX"* and *"Moving hands with fingers felt unnatural, especially with the fingers in this position"*.

Interestingly, though not directly related to the study of physical coupling between different input modalities, researchers in an earlier study which evaluated both hand and head gestures using a multimodal vision-based interface came to similar

conclusions [141]. In their experiment, users were tasked to perform interactive tasks such as selecting app icons on a traditional PC display with a screen-based GUI and then control the functions of an application like the VLC media player. They found that the combination of head and hand gestures resulted in greater task performance accuracy compared to using just the hand or head alone. They also observed in their usage scenario, combining both hand and head gestures resulted in more number of input gestures and these are also gestures that are more intuitive and user friendly.

In addition, when it came to comparing fatigue levels between the two techniques, users reported that the ArmX interaction was significantly more tiring than HeadX ($p=0.002$). Reasons given by users for their preference included: "Doing two tasks with the hands made me feel a little more conscious of my arms hurting", "ArmX interaction required much more concentration and focus compared to HeadX interaction" and "HeadX interaction took less effort compared to ArmX interaction". These observations suggest that the distribution of coordinated physical workload between physically-decoupled modalities that operate relatively independently (e.g. finger and head) can go some ways to reduce fatigue, both physically and cognitively.

In summary, our findings show that for multimodal user interactive tasks that involve substantial coordination between the different input modalities, users tend to prefer and perform better with physically-decoupled interaction designs. While some initial adjustment is needed by the user to figure out how the decoupled modalities coordinate together, this does not seem to impact their subsequent performance. Physical-decoupling also appears to reduce the rate of fatigue during interaction.

6.4.5 Summary

Given the possibility of two different multimodal interaction designs, the research attempted to study how physical-coupling between the different input modalities affected the users' performance and preference in asymmetric multi-input object positioning tasks. The results show that when the required degree of coordination between the two input modalities increases, users started to perform more effectively with a physically-decoupled interaction design. They also indicated strong

preference for this decoupling as they found it more natural to divide the coordinated task to different independent parts of their body. Interaction with this physical decoupling design was also perceived to be significantly less tiring by most users. In summary, the findings from this study makes a strong case to consider physically decoupling the different input modalities when designing multi-input multimodal interaction for mobile VR systems.

6.5 Applications for Compound Multi-Input Interactions

Motivated by the results of the previous study, a game experience was designed that utilized multiple and simultaneous spatial controls to 'box' several pizza pieces onscreen - called 'Box That Pizza!', inspired by popular arcade games where users collect fruits in resizable baskets. We got users to play the game and provide their impressions on the game's playability and engagement levels, when played using both ArmX and HeadX interaction designs. This evaluation served two purposes. First, to evaluate if the proposed multimodal interaction designs can be successfully employed to play simple VR headset games. Second, to validate if the findings from the earlier study are consistent in the VR game.

6.5.1 Gameplay Design

The game consisted of a resizable pizza 'box', whose length and breadth are controlled by the X-Fingers interaction method. The box can be moved using either the user's arms or head; essentially employing either the ArmX or the HeadX interaction design respectively. The aim of the game is to enclose every pizza inside the user-controlled rectangular box as fast as possible. Each fit is a success once each pizza's topping and box appear on it. Two sample screens from the application is shown in Figure 6.16.

The game consists of three levels with increasing coordination required between modalities, akin to the three levels in the earlier study. The increasing levels of coordination is achieved by varying the location and size of the pizza. Level 1 consists of four static pizzas at random locations on the screen, and users are

FIGURE 6.16: Sample screens from 'Box That Pizza!' (a) The purple user-controlled box tries to 'catch' the pizza. (b) Pizza toppings and cardboard box appears immediately once user-controlled box encloses the pizza

required to position and resize the box to enclose each pizza in turn. In Level 2, the pizzas are no longer static; they move within a boundary. In this level, users have to 'catch' the pizzas as fast as they can by positioning and resizing their box quickly. In the final Level 3, besides moving about, the pizzas' sizes will grow and shrink, thus requiring the greatest coordination between both interaction modalities.

6.5.2 Participants and Procedure

This study was conducted with 8 participants (2 females, mean age 27.2 years, SD = 2.2, all right-handed). Participants were first shown an introductory animated

PowerPoint presentation which explained both interaction gestures and the game-play design and rules. To avoid sequencing bias, the order of interaction techniques was also counterbalanced with each participant. Firstly, users were required to undergo a 2-second calibration procedure where they performed the gesture with both fingers at the extremities. After this they proceeded to play each level using both interaction techniques. After each level, users were questioned on their interaction design preference. After users played all three levels using both interaction techniques, they were asked to rate their overall experience in terms of gameplay engagement and enjoyment, and also outline their impressions on the technique and game. All data sets were found to be normal, and statistical significance was computed with paired t-tests.

6.5.3 Results and User Impressions

Besides the subjective evaluations, quantitative measures were also obtained in comparing both the ArmX and HeadX interaction designs. The quantitative evaluation measured the time users took to correctly fit each pizza into a box, in seconds. These results are presented as follows, and shown in Figure 6.17.

FIGURE 6.17: User task performance per level in 'Box that Pizza!', with Standard Error bars. Statistical significance is denoted by asterisks (*= $p < 0.05$)

In Level 1, users performed slightly faster using HeadX ($M=12.2s$, $SD=5.4s$) than ArmX ($M=13.95$, $SD=9.1s$); however, this difference was not significant enough

($p=0.65$). In Level 2, however, users performed faster using HeadX ($M=9.45s$, $SD=4.9s$) than ArmX ($M=19.06s$, $SD=11s$), and this difference was found to be significant ($p=0.011$). This trend continues to Level 3 as well, with users performing better using HeadX ($M=10.84s$, $SD=5.47s$) than ArmX ($M=25.3s$, $SD=15.1s$) significantly ($p=0.028$). These results, delineated in the graph in Figure 6.17, supports the results of the earlier study and corroborates the principle finding that as the level of coordination required in tasks increases, users perform better with physically decoupled multimodal interaction techniques.

FIGURE 6.18: Subjective user evaluation: User rating of playability, enjoyment and ease of use for both interaction methods, with Standard Error bars. Statistical significance is denoted by asterisks (*= $p<0.05$; **= $p<0.01$)

Additionally, a post-game survey required participants to rate each technique on three factors: playability, enjoyment and ease of use, shown in Figure 6.18. Playability in this evaluation referred to how controllable each interaction technique was in the VR game. On a Likert scale from 1 to 5 where 1: very low and 5: very high, users gave the following scores: Users rated HeadX ($M=4.25$, $SD=0.71$) as significantly more playable than ArmX ($M=2.88$, $SD=0.99$, $p=0.02$). P1 commented: *"Interacting with the game seemed natural using HeadX"*. Another participant P5 commented: *"For the third level, although challenging, I found HeadX movement to be more controllable than ArmX"*. Users also rated HeadX ($M=4.5$, $SD=0.53$) made the gameplay significantly more enjoyable than ArmX ($M=3.25$, $SD=1.16$,

$p=0.01$); P3 noted that *"I enjoyed the game and the interaction method well. It was easier to move the head than arms to navigate"*. Lastly, they rated HeadX ($M=4.38$, $SD=0.52$) significantly easier to use than ArmX ($M=2.88$, $SD=0.99$, $p<0.01$). One participant, P7, commented: *"I felt it was easier to move my head along with my fingers more. I was finding it a bit difficult to move arms along with fingers."* Such insights were found to be similar to participants' feedback on both techniques in the previous user evaluation. Furthermore, this evaluation placed increased emphasis on participants' general impressions of the gameplay and interaction experience, since their comments also provided further insights into their subjective feedback on aspects of the two interaction designs. Their overall impressions were generally positive, and several participants commented on the high engagement levels of the gameplay, alluding to the fact that the novelty of interaction techniques did not pose much hindrance to the gameplay experience. P5 commented: *"I had fun. Found(sic) the gesture natural and I'm excited to play more such games!"*. P8 noted the increasing challenges that each level posed, commenting: *"Quite challenging when the pizza slices start moving around, but interesting to play at the same time. Also, it was fun to coordinate and control two inputs simultaneously."* Participants also commented on their varying preferences for both interaction methods; P4 noted, *"In the first level, I felt like using arms was easier, then as the complexity of the task increased, I felt it was easier to get coordination with one body part controlling one input so that the effort is equally distributed between the body parts."* P2 commented similarly: *"Distinguishing motion from resizing was easier with HeadX"*. This is in line with our findings and comments from the previous user evaluation, where it was found that distributing different tasks or input modalities across disparate or physically decoupled body parts reduces the cognitive and physical load on the user.

The findings of these studies have implications in the way multimodal interactions can be designed for VR applications. As an example, Tilt Brush by Google is one such application in VR that employs external controllers to draw in a VR environment using seamlessly dynamic brushes whose properties (e.g. color) can change by controlling the relevant external joysticks [142]. The multimodal HeadX and ArmX interaction designs proposed in this chapter can be used to remove the need for the external joysticks in a smartphone-based VR headset. A typical mock-up screen of such a VR application is shown in Figure 6.19. Given the need for tight coordination between the simultaneous control of brush position and color,

the findings presented suggest that the HeadX multimodal interaction design will be most appropriate choice. HeadX can provide good positioning control of the brush using the X-Fingers interaction, while allowing the user to change line colors dynamically using head movement to navigate the color picker onscreen. Note also that in this usage scenario, the head movements do not affect the user's screen gaze location since the HMD is firmly worn on the user's head. Moreover, the 2D input controls introduced in this chapter can also be extended to 3D interactions in future. While the length and breadth of the X-Finger are used to calculate a 2D cursor position onscreen, the addition of depth can prove to be a third parameter to measure, which could be mapped to add a third dimension to the onscreen cursor. This is a possible future extension of these interaction designs.

FIGURE 6.19: Mock-up of a VR paint tool with one 2D spatial input modality controlling the brush tool (C2; represented by the brush icon) and the other input modality simultaneously controlling the brush's color wheel (C1; represented by the white crosshair).

6.6 Conclusion

This chapter has looked at various designs of hand, finger and head-based multi-input interaction methods, and evaluated their performance and usability in different types of interaction tasks. Through the evaluations and their subsequent results, the chapter has identified some key recommendations for multi-input and multimodal interaction design according to the design of the interaction tasks.

The first part of this chapter has investigated multi-input hand-based interactions for two kinds of compound tasks: asymmetrical and symmetrical. Given the possibility of two different multi-input bimanual interaction designs (M1 and M3), we set out to study how the similarity of bimanual hand gesture design affects user task performance when the tasks are differently assigned, and of increasing coordination levels. Our results lead to the conclusion that a non-identical gesture design, such as in ArmX, is recommended when the task assignment is asymmetric, while an identical gesture design, such as in DBF, is recommended when the task assignment is symmetric.

The next section has focused on multimodal interaction designs. Arising from the interaction design of X-Fingers was the ability to incorporate another input modality via the tracking of the user's fingertip that can be controlled either by the movement of the physically-coupled arms or the physically-decoupled head, which were the two multimodal interaction designs presented. Next, given the possibility of these two different multimodal interaction designs, we set out to study how physical-coupling between the different input modalities affected the users' performance and preference. Our results show that when the required degree of coordination between the two input modalities increased, users started to perform more effectively with a physically-decoupled interaction design. They also indicated a strong preference for this decoupling as they found it more natural to divide the coordinated task to different independent parts of their body. Interaction with this physically-decoupled design was also perceived to be significantly less tiring by most users.

The next section describes a sample application designed using HeadX and ArmX, a gameplay experience that was found playable and engaging by users. It also corroborated our findings from the user evaluation on multimodal interaction design. In summary, the findings presented in this section has also provided VR game-based application evidence highlighting the benefit of physically decoupling different input modalities when designing multimodal interaction designs for multi-input tasks in mobile VR that are highly coordinated.

In conclusion, the chapter has investigated and presented multi-input hand-based interaction designs that address particular interaction task types. It also investigates the addition of head movement as a modality with X-Fingers, and evaluates how the physical decoupling of different modalities affect user performance and

gameplay. The findings of these studies have implications in the way we can design multi-input and multimodal interactions for VR applications using our hands, fingers and head as human gestural input elements.

Chapter 7

Conclusions and Future Work

7.1 Summary

The research presented in this thesis has widened the range of available interaction modalities for VR headsets. This was done by investigating and evaluating novel, effective and usable interaction methods for mobile virtual reality applications on VR headsets suited to common interaction tasks in VR. The research has explored the various affordances that different components that make up interaction paradigms provide. Firstly, it has explored the various parts of a human body that can be used as modalities of interaction, such as eye gaze, head movement, arm movement and finger gestures. Next, it has developed methods by which these human inputs can be tracked. This was done by utilizing available sensing technologies such as the gyroscope and RGB camera in a smartphone, or an IR sensor. Lastly, it has investigated common application scenarios in mobile VR, and mapped the developed interaction methods to interaction tasks that are suited to them. Such tasks are either discrete input-based, like menu selection, or continuous input-based, like object manipulation (spatial object positioning and resizing). An overview of the human gestural input elements, interaction methods developed through them and the interaction task context in which they were evaluated is depicted in Figure [7.1](#).

The following novel interaction methods for VR headsets were developed and evaluated in this research using various modalities:

FIGURE 7.1: A list of major novel interaction methods developed in this research, along with their interaction task context in VR.

1. **CardboardSense:** A set of finger tap-based gestures detected on a DIY cardboard-based VR headset by the embedded smartphone's gyroscope. These can be used for a discrete set of inputs on the periphery of such headsets.
2. **DualGaze:** An eye-gazed based interaction method on an eye-tracking enabled VR headset, designed to address the Midas Touch problem in gaze-based object selection tasks in VR.
3. **Twin-Fingers:** A bare finger-based gesture to control an onscreen 2D cursor in VR that utilizes the gap created between fingers to track the gesture through an embedded smartphone's RGB camera.
4. **X-Fingers:** A covered finger-based gesture to control an onscreen 2D cursor in VR that is tracked through an embedded smartphone's RGB camera. This is invariant to head and hand motion.
5. **Double Bunny Fingers:** A multi-input bimanual gesture that tracks the space between fingers to control two simultaneous inputs in VR that are spatially disparate.
6. **ArmX:** A multi-input bimanual gesture that combines X-Fingers with whole arm movement, used to control two simultaneous inputs that are spatially nested.

7. **HeadX:** A multimodal gesture that combines X-Fingers and head movement to control two simultaneous inputs requiring coordination between them in VR.

By developing these novel interaction methods, the research presented in this thesis has broadened the existing interaction capabilities of commodity VR headsets available today. The findings in this research can also guide further interaction design for the mobile VR environment. This has the potential to make VR headsets more usable and open the possibilities of deploying more interactive applications with stereoscopic experiences.

7.2 General Insights and Conclusions

Evaluating the various interaction methods developed through user studies has led to interesting insights on the nature of human interaction with a VR system. Some of these were elicited through users' subjective feedback, and others were implied through quantitative results. The insights identified may be used as a set of recommendations for interaction design in VR headsets, and are detailed as follows.

C1. Interaction designs should match the nature of the interaction task closely.

Through various interaction methods, it was observed that there is no singular multi-purpose interaction method that suits all kinds of different tasks in a VR environment. Interaction tasks that require physical and cognitive interdependency should have interaction methods of a similar nature designed for them. For example in multi-input tasks, it was found that when the task assignment was symmetric, a physically identical two-input interaction design such as DBF was preferred to the non-identical design of ArmX. This preference did not persist when the task assignment was asymmetric; rather, it was the physically non-identical design of ArmX that users preferred for such tasks. This pattern was corroborated through quantitative results too. Another example is DualGaze, which was perceived to be a reliable and accurate object selection method. Users preferred DualGaze most in scenarios where the interaction task was deemed critical and irreversible, such as

exiting a game or answering a quiz question. Whereas the simplicity of Fixed Gaze was preferred when the task involved a reversible and non-critical action. These observations lead one to the conclusion that interactions should be designed with consideration for the perceived nature of task to be performed.

C2. Familiarity and ease of use are important considerations in interaction design.

While there are several factors determining user preference of one interaction method over the other, the observations from this research have been that the simplicity of gesture precedes other influencing factors. Interaction methods that are convenient and easy to use are preferred by users over more complex interactions, despite other advantages. For example, most of the general comments obtained from users regarding their impressions on various interaction methods involved justifications based on comfort, ease of use and simplicity. In another observation, while Twin-Fingers offered a faster onscreen navigational option, some users preferred the simplicity of the Bunny-Fingers interaction due to TF's more unfamiliar nature (see Section 5.5). Hence, immediate ease of use and user comfort are major considerations to note while designing novel interaction methods. Even though users can adopt to more complex gestures with time and training, the widespread adoption of any interaction design depends very much on first impression and immediate usability.

C3. Understanding the influence of coordination levels between inputs is an important consideration in interaction design.

The various multi-input interaction designs presented in this research were found to suit various interaction task scenarios. Certain tasks were performed better using certain interaction methods, due to their particular designs, which showed that the nature of the task influenced the interaction design as discussed in C1. But such task-to-interaction method mappings need to also consider the degree of coordination within the multi-input tasks. For example, for simple compound tasks, it was observed that performance differences using either HeadX or ArmX interaction design were not significant (see Section 6.4). However, as the level of coordination between both modalities increased, the preference for HeadX became more pronounced and the quantitative performance on those tasks was also significantly superior to ArmX. Understanding the level of coordination between the

task can help us choose the most appropriate multi-input interaction design. A similar trend was also observed in the study of the effects of task symmetry on physical interaction design (see Section 6.3). While the general finding from the evaluation was that task assignment symmetry and physical interaction design are codependent, this codependence was strengthened with an increase in coordination levels for both task assignment types.

C4. Coordinating disparate parts of the human body during multi-input interaction can work well.

It was observed that users have a preference to assigned different parts of their body to control different dimensions of a coordinated multi-input task. The higher the coordination level of these tasks, the greater the preference for these parts of the body to be more independently controllable. For example, the comparative study of ArmX and HeadX interaction designs showed that the physically-decoupled HeadX design, which allows the user's head and fingers to be controlled independently resulted in increasingly better performance as the level of coordination between the tasks increased. The results from the "Box the Pizza" game further support this insight. The notion that people will always find it difficult to coordinate disparate parts of their body does not bear out with our findings, at least not with the HeadX interaction design. They may find it unnatural initially, but once cognitive coherence of the gesture to task mapping has been established, their performance proved superior due to the affordance of physical parallelism.

7.3 Limitations and Future Work

While not comprehensive, this thesis has looked at major input elements of the human body and how available sensing technologies can be exploited to employ such elements as interaction methods for applications in VR. The continuation of this research would be a step further into establishing a Natural User Interface (NUI) framework for smartphone-based VR headsets. However, there are ways in which the proposed interactions can be made more natural and less reliant on external sensors. These come with a few challenges that still need to be addressed.

The future directions of this research include addressing the following limitations.

- Continuous input for Cardboard-based VR headsets:** The idea of smartphone motion sensors detecting movement on the periphery of cardboard-based headsets can be extended beyond the set of discrete tap gestures proposed through the interaction method CardboardSense. Currently, such a discrete set of tap gestures makes it difficult for users to perform continuous input onscreen, such as drawing or navigation. A possible means of extracting continuous input is to investigate if sliding one's fingers on a short flexible stick-on strip of serrated 'teeth' could generate vibration patterns that can be instantaneously localized for continuous spatial control. A depiction is shown in Figure 7.2. Such strips could vary in length and breadth to control the spatial ranges offered by the input.

FIGURE 7.2: Continuous input on Cardboard-based headsets by sliding fingers over an affixed serrated strip.

- Robust eye gaze tracking across users using RGB front camera of smartphone:** DualGaze is a gaze-based interaction method that is based on an eye tracking-enabled VR headset. Thus far, eye tracking has been limited to being dependent on extraneous IR sensors, and complex algorithms such as Daughman's algorithm[143]. Eye tracking has also been largely used in ideal environments with ample lighting, no obstructions, high resolution cameras and completely visible irises. The VR headset imposes several constraints on the environment: ambient lighting is generally low due to the enclosed build of the headset, and, unlike most iris localization algorithms, this deals with standalone eye images as viewed through the headsets lens, which means facial feature recognition cannot be used to localize the iris position. Moreover, smartphone cameras generally have low-resolution cameras at the front,

which makes the captured frames of the user's eye low in quality and often out of focus. These challenges need to be addressed in order to implement a real-time eye tracking application within an embedded smartphone in a VR headset, using only the phone's front-facing camera. The eye gaze of the user can be tracked through using deep learning on a suitable dataset. This would completely remove the dependence of eye tracking-enabled headsets, and offer eye gaze interaction as an input modality for smartphone-based VR headsets.

- **Robust bare finger tracking across users with RGB rear camera of smartphone:** Bare skin tracking comes with a few challenges, including sensitivity to lighting conditions, background noise and skin colour. In future, machine learning-based vision methods can be explored to robustly track hands and fingers with a single RGB camera. In such a design, finger coverings will not be needed for interaction methods such as X-Fingers, and users can perform such finger and hand-based interactions using their bare fingers and hands with a smartphone-based VR headset.
- **The effects of combining eye gaze with hand and head-based interactions in VR:** To support further multimodality in VR headsets, the effects of combining eye gaze-based interactions with head, hand and finger-based interactions have to be evaluated. These various modalities can act in concert to create a multimodal interaction system in VR, with each input modality controlling a different interaction type, guided by the findings from this research. Such combinations require further evaluation of feasibility and application scenarios, and such evaluations can provide further insights into the nature of multimodality in the VR interaction paradigm.

Thus, addressing these research directions in the future can be seen as a step forward to establish an NUI system for smartphone-based VR headsets.

List of Author's Awards, Patents, and Publications¹

Conference Proceedings

- **Pallavi Mohan**, Wooi Boon Goh, Chi-Wing Fu, and Sai-Kit Yeung. 2019. Head-Fingers-Arms: Physically-Coupled and Decoupled Multimodal Interaction Designs in Mobile VR. In *The 17th International Conference on Virtual-Reality Continuum and its Applications in Industry (VRCAI '19)*. Association for Computing Machinery, New York, NY, USA, Article 28, 1–9.
- **P. Mohan**, W. B. Goh, C. Fu and S. Yeung, DualGaze: Addressing the Midas Touch Problem in Gaze Mediated VR Interaction, *2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, Munich, Germany, 2018, pp. 79-84.
DOI: 10.1109/ISMAR-Adjunct.2018.00039.
- Xiaoqi Yan, Chi-Wing Fu, **Pallavi Mohan**, and Wooi Boon Goh. 2016. CardboardSense: Interacting with DIY Cardboard VR Headset by Tapping. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems (DIS '16)*. ACM, New York, NY, USA, 229-233.
-

¹The superscript * indicates joint first authors

Bibliography

- [1] Xiaoqi Yan, Chi Wing Fu, Pallavi Mohan, and Wooi Boon Goh. Cardboardsense: Interacting with DIY cardboard VR headset by tapping. In *DIS 2016 - Proceedings of the 2016 ACM Conference on Designing Interactive Systems: Fuse*, pages 229–233, 2016. ISBN 9781450340311. doi: 10.1145/2901790.2901813. [vii](#), [25](#)
- [2] Pallavi Mohan, Wooi Boon Goh, Chi Wing Fu, and Sai Kit Yeung. Dual-Gaze: Addressing the Midas Touch Problem in Gaze Mediated VR Interaction. In *Adjunct Proceedings - 2018 IEEE International Symposium on Mixed and Augmented Reality, ISMAR-Adjunct 2018*, pages 79–84. IEEE, oct 2019. ISBN 9781538675922. doi: 10.1109/ISMAR-Adjunct.2018.00039. URL <https://ieeexplore.ieee.org/document/8699248/>. [vii](#), [45](#)
- [3] Pallavi Mohan, Wooi Boon Goh, Chi-Wing Fu, and Sai-Kit Yeung. Head-fingers-arms: Physically-coupled and decoupled multimodal interaction designs in mobile vr. In *The 17th International Conference on Virtual-Reality Continuum and Its Applications in Industry, VRCAI '19*, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450370028. doi: 10.1145/3359997.3365697. URL <https://doi.org/10.1145/3359997.3365697>. [viii](#), [90](#)
- [4] Clay Wilson. Avatars, Virtual Reality Technology, and the U.S. Military: Emerging Policy Issues. Technical report, 2008. URL <http://integrator.hanscom.af.mil/>. [1](#)
- [5] Alan B. Craig, William R. Sherman, and Jeffrey D. Will. *Developing virtual reality applications : foundations of effective design*. Morgan Kaufmann, 2009. ISBN 9780123749437. [1](#)

- [6] M McGill, Daniel Boland, Roderick Murray-Smith, and Stephen Brewster. A dose of reality: overcoming usability challenges in vr head-mounted displays. *Proceedings of the 33rd Annual*, (May):2143–2152, 2015. ISSN 1450331459. doi: 10.1145/2702123.2702382(<http://dx.doi.org/10.1145/2702123.2702382>). 3
- [7] Nasser Dardas, Nasser H Dardas, and Mohammad Alhaj. Hand Gesture Interaction with a 3D Virtual Environment. *International Journal of ACM Jordan*, 2(1), 2011. ISSN 2078-7952. URL <https://www.researchgate.net/publication/235918973>. 3
- [8] Wanhong Lin, Lear Du, Carisa Harris-Adamson, Alan Barr, and David Rempel. Design of hand gestures for manipulating objects in virtual reality. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 10271, pages 584–592. Springer, Cham, 2017. doi: 10.1007/978-3-319-58071-5_44. URL http://link.springer.com/10.1007/978-3-319-58071-5_{_}44. 3, 23
- [9] Daniel P. Mapes and J. Michael Moshell. A Two-Handed Interface for Object Manipulation in Virtual Environments. *Presence: Teleoperators and Virtual Environments*, 4(4):403–416, jan 2015. ISSN 1054-7460. doi: 10.1162/pres.1995.4.4.403. URL <http://www.mitpressjournals.org/doi/10.1162/pres.1995.4.4.403>. 3, 23, 89
- [10] Oviatt and Sharon. Multimodal interactive maps: designing for human performance. *Human-Computer Interaction*, 12(1):93–129, 1997. ISSN 0737-0024. doi: 10.1207/S15327051HCI1201&2_4. URL <https://dl-acm-org.ezlibproxy1.ntu.edu.sg/citation.cfm?id=1462973>. 3, 91
- [11] Boris Velichkovsky, Andreas Sprenger, and Pieter Unema. Towards gaze-mediated interaction: Collecting solutions of the “Midas touch problem”. In *Human-Computer Interaction INTERACT '97*, pages 509–516. Springer US, Boston, MA, 1997. doi: 10.1007/978-0-387-35175-9_77. URL http://link.springer.com/10.1007/978-0-387-35175-9_{_}77. 5, 45, 49, 50, 54
- [12] Ivan E. Sutherland. The ultimate display. In *Proceedings of the IFIP Congress*, pages 506–508, 1965. 10

- [13] Asmaa Saeed Alqahtani, Lamy Foad Daghestani, and Lamiaa Fattouh Ibrahim. Environments and System Types of Virtual Reality Technology in STEM: A Survey. Technical Report 6, 2017. URL www.ijacsa.thesai.org. 10
- [14] Tomasz Mazuryk and Michael Gervautz. Virtual Reality - History, Applications, Technology and Future. pages 1–72, 1996. URL <https://www.semanticscholar.org/paper/Virtual-Reality-History-2C-Applications-2C-Technology-Mazuryk/6cebef400df6c387852b6c76570f226a86965104http://www.sciencedirect.com/science/article/pii/B9780124047051000066><http://linkinghub.elsevier.com/retrieve/pii/B9780124047051000078><http://www.cg.tuwien.ac.at/research/publications/1996/mazuryk-1996-VRH/><http://citeseerx.ist.psu.edu/viewdoc/dow>. 10
- [15] Frank Steinicke. *Being really virtual: Immersive natives and the future of virtual reality*. Springer International Publishing, jan 2016. ISBN 9783319430782. doi: 10.1007/978-3-319-43078-2. 10
- [16] Ivan E. Sutherland. A head-mounted three dimensional display. In *Proceedings of the December 9-11, 1968, fall joint computer conference, part I on - AFIPS '68 (Fall, part I)*, page 757, New York, New York, USA, 1968. ACM Press. doi: 10.1145/1476589.1476686. URL <http://portal.acm.org/citation.cfm?doid=1476589.1476686>. 10
- [17] Jamie Sharpe and Richard Self. Computers for everyone. *Meta-science*, 25(2):279–280, 2016. ISSN 0815-0796. doi: 10.1007/s11016-016-0060-2. URL <https://computing.derby.ac.uk/ojs/index.php/c4e/article/view/90>. 10
- [18] Zoran Zivkovic. Optical-Flow-Driven Gadgets for Gaming User Interface. pages 90–100. Springer, Berlin, Heidelberg, 2004. doi: 10.1007/978-3-540-28643-1_12. URL http://link.springer.com/10.1007/978-3-540-28643-1_12. 11
- [19] Bill Buxton and George W. Fitzmaurice. HMDs, caves & chameleon. *ACM SIGGRAPH Computer Graphics*, 32(4):69–74, nov 1998. ISSN 00978930.

- doi: 10.1145/307710.307732. URL <http://portal.acm.org/citation.cfm?doid=307710.307732>. 11, 12
- [20] Maryam Vafadar. Virtual reality: Opportunities and challenges. 2013. 12
- [21] Francisco Rebelo, Paulo Noriega, Emília Duarte, and Marcelo Soares. Using virtual reality to assess user experience. In *Human Factors*, volume 54, pages 964–982, dec 2012. doi: 10.1177/0018720812465006. URL <http://journals.sagepub.com/doi/10.1177/0018720812465006>. 12
- [22] M Slater and M V Sanchez-Vives. Enhancing Our Lives with Immersive Virtual Reality. *Front. Robot. AI*, 3:74, 2016. doi: 10.3389/frobt.2016.00074. URL <https://www.youtube.com/watch?v=3L0N7CKv0BA>. 12
- [23] Mikael Eriksson. Reaching out to grasp in Virtual Reality A qualitative usability evaluation of interaction techniques for selection and manipulation in a VR game. pages 1–57, 2016. URL <http://www.diva-portal.org/smash/get/diva2:946219/FULLTEXT01.pdf>. 12, 20
- [24] Oculus Rift: VR Headset for VR Ready PCs — Oculus. URL <https://www.oculus.com/rift/{#}oui-csl-rift-games=mages-tale>. 12, 13
- [25] Samsung Gear VR with Controller - The Official Samsung Galaxy Site, . URL <http://www.samsung.com/global/galaxy/gear-vr/>. 13
- [26] Samsung VR - Home, . URL <https://samsungvr.com/>. 13
- [27] VIVE™ — Discover Virtual Reality Beyond Imagination. URL <https://www.vive.com/sg/>. 13
- [28] P. Dempsey. The Teardown: HTC Vive virtual reality headset. *Engineering & Technology*, 11(7):80–81, aug 2016. ISSN 1750-9637. doi: 10.1049/et.2016.0731. URL <https://digital-library.theiet.org/content/journals/10.1049/et.2016.0731>. 13
- [29] Daydream. URL <https://vr.google.com/daydream/>. 13, 15
- [30] Google Cardboard – Google VR. URL <https://vr.google.com/cardboard/>. 13

- [31] Soojeong Yoo and Callum Parker. Controller-less Interaction Methods for Google Cardboard. *Proceedings of the 3rd ACM Symposium on Spatial User Interaction - SUI '15*, (AUGUST):127–127, 2015. ISSN 18650929. doi: 10.1145/2788940.2794359. URL <http://dl.acm.org/citation.cfm?doid=2788940.2794359>. 14, 15
- [32] PlayStation VR – Virtual Reality Headset for PS4. URL <https://www.playstation.com/en-us/explore/playstation-vr/>. xix, 16
- [33] Scott Stein. Sony PlayStation VR Release Date, Price and Specs - CNET. dec 2015. URL <http://www.cnet.com/products/sony-playstation-vr/>. 16
- [34] Samsung VR - Home, 2017. URL <https://samsungvr.com/>. xix, 17
- [35] Parth and Rajesh Desai, Pooja Nikhil Desai, Komal Deepak Ajmera, and Khushbu Mehta. A Review Paper on Oculus Rift-A Virtual Reality Headset. *International Journal of Engineering Trends and Technology*, 13(4):175–179, aug 2014. doi: 10.14445/22315381/ijett-v13p237. URL <http://arxiv.org/abs/1408.1173>. 17
- [36] Chun Yu, Yizheng Gu, Zhican Yang, Xin Yi, Hengliang Luo, and Yuanchun Shi. Tap, Dwell or Gesture?: Exploring Head-Based Text Entry Techniques for HMDs. *Chi 2017*, pages 0–0, 2017. doi: 10.1145/3025453.3025964. 18, 21
- [37] Rowel Atienza, Ryan Blonna, Maria Isabel Saldares, Joel Casimiro, and Vivencio Fuentes. Interaction techniques using head gaze for virtual reality. In *Proceedings - 2016 IEEE Region 10 Symposium, TENSYP 2016*, pages 110–114. IEEE, may 2016. ISBN 9781509009312. doi: 10.1109/TENCONSpring.2016.7519387. URL <http://ieeexplore.ieee.org/document/7519387/>. 18
- [38] Changchong Sheng, Libing Jiang, Bo Tang, and Xiaoan Tang. A Novel Menu Interaction Method Using Head-mounted Display for Smartphone-based Virtual Reality. pages 22–25, 2017. 18, 22
- [39] E. Foxlin and M. Harrington. WearTrack: a self-referenced head and hand tracker for wearable computers and portable VR. In *Digest of Papers. Fourth International Symposium on Wearable Computers*, pages 155–162. IEEE

- Comput. Soc, 2002. ISBN 0-7695-0795-6. doi: 10.1109/iswc.2000.888482. URL <http://ieeexplore.ieee.org/document/888482/>. 18
- [40] Vildan Tanriverdi and Robert J. K. Jacob. Interacting with eye movements in virtual environments. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '00*, pages 265–272, New York, New York, USA, 2000. ACM Press. ISBN 1581132166. doi: 10.1145/332040.332443. URL <http://portal.acm.org/citation.cfm?doid=332040.332443>. 18, 48
- [41] Florian Geiselhart, Michael Rietzler, and Enrico Rukzio. EyeVR. *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing Adjunct - UbiComp '16*, pages 277–280, 2016. doi: 10.1145/2968219.2971384. URL <http://dl.acm.org/citation.cfm?doid=2968219.2971384>. 18
- [42] Jason Orlosky, Takumi Toyama, Kiyoshi Kiyokawa, and Daniel Sonntag. ModulAR: Eye-Controlled Vision Augmentations for Head Mounted Displays. *IEEE Transactions on Visualization and Computer Graphics*, 21(11): 1259–1268, nov 2015. ISSN 1077-2626. doi: 10.1109/TVCG.2015.2459852. URL <http://ieeexplore.ieee.org/document/7164337/>. 18, 48
- [43] Francisco Lopez Luro and Veronica Sundstedt. A comparative study of eye tracking and hand controller for aiming tasks in virtual reality. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research and Applications*, ETRA '19, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450367097. doi: 10.1145/3317956.3318153. URL <https://doi.org/10.1145/3317956.3318153>. 19
- [44] FOVE Eye Tracking Virtual Reality Headset. URL <https://www.getfove.com/>. xx, 19, 45, 46, 48, 53
- [45] A Cockburn, P Quinn, C Gutwin, G Ramos, and J Looser. Air pointing: Design and evaluation of spatial target acquisition with and without visual feedback. *Journal of Human Computer Studies*, 69:401–414, 2011. doi: 10.1016/j.jhcs.2011.02.005. URL www.elsevier.com/locate/jhcs. 19, 67
- [46] Daniel Vogel and Ravin Balakrishnan. Distant Freehand Pointing and Clicking on Very Large, High Resolution Displays. Technical

- report, 2005. URL <https://www.dgp.toronto.edu/{~}ravin/papers/uist2005{~}distantpointing.pdf>. 19, 67
- [47] Isaac Cho and Zachary Wartell. Evaluation of a bimanual simultaneous 7DOF interaction technique in virtual environments. In *2015 IEEE Symposium on 3D User Interfaces (3DUI)*, pages 133–136. IEEE, mar 2015. ISBN 978-1-4673-6886-5. doi: 10.1109/3DUI.2015.7131738. URL <http://ieeexplore.ieee.org/document/7131738/>. 19
- [48] Eric Whitmire, Mohit Jain, Divye Jain, Greg Nelson, Ravi Karkar, Shwetak Patel, and Mayank Goel. DigiTouch: Reconfigurable Thumb-to-Finger Input and Text Entry on Head-mounted Displays. *PACM Interact. Mob. Wearable Ubiquitous Technol. Article*, 1(21):1–21, 2017. ISSN 24749567. doi: 10.1145/3130978. 19, 21
- [49] Jens Grubert, Lukas Witzani, Eyal Ofek, Michel Pahud, Matthias Kranz, and Per Ola Kristensson. Text Entry in Immersive Head-Mounted Display-based Virtual Reality using Standard Keyboards. pages 1–8, 2018. URL <http://arxiv.org/abs/1802.00626>. 19
- [50] Jaeyeon Lee, Mike Sinclair, Mar Gonzalez-Franco, Eyal Ofek, and Christian Holz. Torc: A virtual reality controller for in-hand high-dexterity finger interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI ’19, pages 71:1–71:13, New York, NY, USA, 2019. ACM. ISBN 978-1-4503-5970-2. doi: 10.1145/3290605.3300301. URL <http://doi.acm.org/10.1145/3290605.3300301>. 19
- [51] Akira Ishii, Masaya Tsuruta, Ippei Suzuki, Shuta Nakamae, Tatsuya Minagawa, Junichi Suzuki, and Yoichi Ochiai. ReverseCAVE. *ACM SIGGRAPH 2017 Posters on - SIGGRAPH ’17*, pages 1–2, 2017. doi: 10.1145/3102163.3102208. URL <http://dl.acm.org/citation.cfm?doid=3102163.3102208>. xxi, 19, 22, 67, 68, 74
- [52] Jan Gugenheimer, David Dobbstein, Christian Winkler, Gabriel Haas, and Enrico Rukzio. FaceTouch. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA ’16*, pages 3679–3682, New York, New York, USA, 2016. ACM Press. ISBN 9781450340823. doi: 10.1145/2851581.2890242.

- URL <http://dl.acm.org/citation.cfm?doid=2984511.2984576><http://dl.acm.org/citation.cfm?doid=2851581.2890242>. 20, 67
- [53] Jan Gugenheimer, Evgeny Stemasov, Julian Frommel, and Enrico Rukzio. ShareVR: Enabling Co-Located Experiences for Virtual Reality between HMD and Non-HMD Users. *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems - CHI '17*, pages 4021–4033, 2017. doi: 10.1145/3025453.3025683. URL <http://dl.acm.org/citation.cfm?doid=3025453.3025683>. 20
- [54] Jihyun Lee, Byungmoon Kim, Bongwon Suh, and Eunye Koh. Exploring the Front Touch Interface for Virtual Reality Headsets. *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA '16*, pages 2585–2591, 2016. doi: 10.1145/2851581.2892344. URL <http://dl.acm.org/citation.cfm?doid=2851581.2892344><http://arxiv.org/abs/1608.00447>. 20
- [55] Sangho Lee, Gyutae Ha, Jaekwang Cha, Jinhyeok Kim, Hojun Lee, and Shiho Kim. CyberTouch - Touch and Cursor Interface for VR HMD. pages 503–507. 2015. doi: 10.1007/978-3-319-21380-4_85. URL http://link.springer.com/10.1007/978-3-319-21380-4_{_}85. 20
- [56] Marcos Serrano, Barrett M. Ens, and Pourang P. Irani. Exploring the use of hand-to-face input for interacting with head-worn displays. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*, pages 3181–3190, New York, New York, USA, 2014. ACM Press. ISBN 9781450324731. doi: 10.1145/2556288.2556984. URL <http://dl.acm.org/citation.cfm?doid=2556288.2556984>. 20
- [57] Koki Yamashita, Takashi Kikuchi, Katsutoshi Masai, Maki Sugimoto, Bruce H. Thomas, and Yuta Sugiura. CheekInput. pages 1–8, 2017. doi: 10.1145/3139131.3139146. 20
- [58] Jinhyuk Kim, Jaekwang Cha, Hojun Lee, and Shiho Kim. Hand-free natural user interface for VR HMD with IR based facial gesture tracking sensor. In *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology - VRST '17*, pages 1–2, New York, New York, USA, 2017. ACM Press. ISBN 9781450355483. doi: 10.1145/3139131.3143420. URL <http://dl.acm.org/citation.cfm?doid=3139131.3143420>. 20

- [59] Katsuhiro Suzuki, Fumihiko Nakamura, Jiu Otsuka, Katsutoshi Masai, Yuta Itoh, Yuta Sugiura, and Maki Sugimoto. Recognition and mapping of facial expressions to avatar by embedded photo reflective sensors in head mounted display. In *2017 IEEE Virtual Reality (VR)*, pages 177–185. IEEE, 2017. ISBN 978-1-5090-6647-6. doi: 10.1109/VR.2017.7892245. URL <http://ieeexplore.ieee.org/document/7892245/>. 20
- [60] William Saunders and Daniel Vogel. Tap-kick-click: Foot interaction for a standing desk. In *DIS 2016 - Proceedings of the 2016 ACM Conference on Designing Interactive Systems: Fuse*, pages 323–333. Association for Computing Machinery, Inc, jun 2016. ISBN 9781450340311. doi: 10.1145/2901790.2901815. 20
- [61] Daniel Lopes, Filipe Relvas, Soraia Paulo, Yosra Rekik, Laurent Grisoni, and Joaquim Jorge. FEETICHE: FEET Input for Contactless Hand gEsture Interaction. pages 1–10. Association for Computing Machinery (ACM), 2019. ISBN 9781450370028. doi: 10.1145/3359997.3365704. 20
- [62] Te-Yan Wu and Mike Y Chen. EyeExpression: Exploring the Use of Eye Expressions as Hands-free Input for Virtual and Augmented Reality Devices. 2017. doi: 10.1145/3139131.3141206. URL <https://doi.org/10.1145/3139131.3141206>. 20
- [63] Hemant Bhaskar Surale, Aakar Gupta, Mark Hancock, and Daniel Vogel. Tabletinvr: Exploring the design space for using a multi-touch tablet in virtual reality. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI ’19, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450359702. doi: 10.1145/3290605.3300243. URL <https://doi.org/10.1145/3290605.3300243>. 20
- [64] David Navarre, Philippe Palanque, Rémi Bastide, Amélie Schyn, Marco Winckler, Luciana P. Nedel, and Carla M. D. S. Freitas. A Formal Description of Multimodal Interaction Techniques for Immersive Virtual Reality Applications. pages 170–183. Springer, Berlin, Heidelberg, 2005. doi: 10.1007/11555261_17. URL http://link.springer.com/10.1007/11555261_{_}17. 20
- [65] Ivan Poupyrev, Suzanne Weghorst, Mark Billingham, and Tadao Ichikawa. A framework and testbed for studying manipulation techniques for immersive

- vr. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology*, VRST '97, pages 21–28, New York, NY, USA, 1997. ACM. ISBN 0-89791-953-X. doi: 10.1145/261135.261141. URL <http://doi.acm.org/10.1145/261135.261141>. 20
- [66] Jens Grubert, Lukas Witzani, Eyal Ofek, Michel Pahud, Matthias Kranz, and Per Ola Kristensson. Text Entry in Immersive Head-Mounted Display-Based Virtual Reality Using Standard Keyboards. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pages 159–166. IEEE, mar 2018. ISBN 978-1-5386-3365-6. doi: 10.1109/VR.2018.8446059. URL <https://ieeexplore.ieee.org/document/8446059/>. 21
- [67] Jens Grubert, Lukas Witzani, Eyal Ofek, Michel Pahud, Matthias Kranz, and Per Ola Kristensson. Effects of Hand Representations for Typing in Virtual Reality. pages 1–8, 2018. URL <http://arxiv.org/abs/1802.00613>. 21
- [68] Doug A. Bowman, Christopher J. Rhoton, and Marcio S. Pinho. Text Input Techniques for Immersive Virtual Environments: An Empirical Comparison. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 46(26):2154–2158, 2002. ISSN 1541-9312. doi: 10.1177/154193120204602611. URL <http://journals.sagepub.com/doi/10.1177/154193120204602611>. 21
- [69] Jia-Wei Lin, Ping-Hsuan Han, Jiun-Yu Lee, Yang-Sheng Chen, Ting-Wei Chang, Kuan-Wen Chen, and Yi-Ping Hung. Visualizing the keyboard in virtual reality for enhancing immersive experience. In *ACM SIGGRAPH 2017 Posters on - SIGGRAPH '17*, volume 1, pages 1–2, 2017. ISBN 9781450350150. doi: 10.1145/3102163.3102175. URL <https://doi.org/10.1145/3102163.3102175><http://dl.acm.org/citation.cfm?doid=3102163.3102175>. 21
- [70] Daniel Schneider, Alexander Otte, Travis Gesslein, Philipp Gagel, Bastian Kuth, Mohamad Shahm Damla, Oliver Dietz, Eyal Ofek, Michel Pahud, Per Ola Kristensson, Jörg Müller, and Jens Grubert. Reconfiguration: Reconfiguring physical keyboards in virtual reality. *ArXiv*, abs/1907.08153, 2019. 21
- [71] James Q. Walker. Decoder-assisted typing using an hmd and a physical keyboard. 2016. 21

- [72] R. Rosenberg and M. Slater. The chording glove: a glove-based text input device. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, 29(2):186–191, may 1999. ISSN 10946977. doi: 10.1109/5326.760563. URL <http://ieeexplore.ieee.org/document/760563/>. 21
- [73] 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, New York, USA, 2018. ACM Press. ISBN 9781450356206. doi: 10.1145/3173574.3174221. URL <http://dl.acm.org/citation.cfm?doid=3173574.3174221>. 21
- [74] Difeng Yu, Kaixuan Fan, Heng Zhang, Diego Monteiro, Wenge Xu, and Hai-Ning Liang. PizzaText: Text Entry for Virtual Reality Systems Using Dual Thumbsticks. *IEEE Transactions on Visualization and Computer Graphics*, 24(11):2927–2935, nov 2018. ISSN 1077-2626. doi: 10.1109/TVCG.2018.2868581. URL <https://ieeexplore.ieee.org/document/8456570/>. 21
- [75] John Paulin Hansen, Kristian Tørning, Anders Sewerin Johansen, Kenji Itoh, and Hirotaka Aoki. Gaze typing compared with input by head and hand. In *Proceedings of the Eye tracking research & applications symposium on Eye tracking research & applications - ETRA'2004*, pages 131–138, New York, New York, USA, 2004. ACM Press. ISBN 1581138253. doi: 10.1145/968363.968389. URL <http://portal.acm.org/citation.cfm?doid=968363.968389>. 21
- [76] Hendrik Koesling, Martin Zoellner, Lorenz Sichelschmidt, and Helge Ritter. With a Flick of the Eye : Assessing Gaze-Controlled Human-Computer Interaction. *Cosmos*, pages 83–92. 21
- [77] Hellen Apel, Simon Vöge, and Christine Kaufmann. Twenty Years of Eye Typing : Systems and Design Issues Gliederung, 2013. URL http://delivery.acm.org.ezlibproxy1.ntu.edu.sg/10.1145/510000/507076/p15-majaranta.pdf?ip=155.69.24.171&id=507076&acc=ACTIVESERVICE&key=FF6731C4D3E3CFFF.B51AA0289512DB67.4D4702B0C3E38B35.4D4702B0C3E38B35&{_}{_}acm{_}{_}=1529684654{_}43174ad315fd9cb9f591fbf8898a6d8http://portal. 21

- acm.org/citation.cfm?id=507076{%}5Cnhttp://dl.acm.org/citation.cfm?id=507076. 21
- [78] Xinyao Ma, Zhaolin Yao, Yijun Wang, Weihua Pei, and Hongda Chen. Combining Brain-Computer Interface and Eye Tracking for High-Speed Text Entry in Virtual Reality. In *Proceedings of the 2018 Conference on Human Information Interaction & Retrieval - IUI '18*, pages 263–267, New York, New York, USA, 2018. ACM Press. ISBN 9781450349451. doi: 10.1145/3172944.3172988. URL <http://dl.acm.org/citation.cfm?doid=3172944.3172988>. 21, 48
- [79] Doug A. Bowman, Ernst Kruijff, Joseph J. LaViola, and Ivan Poupyrev. *3D User Interfaces: Theory and Practice*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 2004. ISBN 0201758679. 22
- [80] Mark R. Mine. Virtual environment interaction techniques. Technical report, Chapel Hill, NC, USA, 1995. 22
- [81] Alex Olwal and Steven Feiner. The flexible pointer: An interaction technique for selection in augmented and virtual reality. *Uist'03*, 03:81–82, 2003. doi: 10.1.1.214.629. URL <http://www.acm.org/uist/archive/adjunct/2003/pdf/posters/p17-olwal.pdf>. 22
- [82] Shimmila Bhowmick and Keyur Sorathia. Report: State of the Art Seminar Explorations on Body-Gesture based Object Selection on HMD based VR Interfaces for Dense and Occluded Dense Virtual Environments. Technical report, 2018. URL http://embeddedinteractions.com/Files/SOAS{_}Report{_}Shimmila{_}Bhowmick.pdf. 22
- [83] Sven Mayer, Valentin Schwind, Robin Schweigert, and Niels Henze. The Effect of Offset Correction and Cursor on Mid-Air Pointing in Real and Virtual Environments. 2018. doi: 10.1145/3173574.3174227. URL <https://doi.org/10.1145/3173574.3174227>. 22
- [84] Mikko Kytö, Barrett Ens, Thammathip Piumsomboon, Gun A Lee, and Mark Billingham. Pinpointing: Precise Head- and Eye-Based Target Selection for Augmented Reality. *Chi 2018*, page (In press), 2018. doi: <https://doi.org/10.1145/3173574.3173655>. 22, 92

- [85] Yuanyuan Qian and Robert J Teather. The Eyes Don ' t Have It : An Empirical Comparison of Head -Based and Eye-Based Selection in Virtual Reality. *Sui '17*, pages 1–8, 2017. doi: 10.1145/3131277.3132182. [22](#)
- [86] Namgyu Kim, G.J. Kim, Chan-Mo Park, Inseok Lee, and S.H. Lim. Multi-modal menu presentation and selection in immersive virtual environments. In *Proceedings IEEE Virtual Reality 2000 (Cat. No.00CB37048)*, page 281. IEEE Comput. Soc, 2002. ISBN 0-7695-0478-7. doi: 10.1109/vr.2000.840509. URL <http://ieeexplore.ieee.org/document/840509/>. [22](#)
- [87] Matthew M. Davis, Joseph L. Gabbard, Doug A. Bowman, and Dennis Gracanin. Depth-based 3D gesture multi-level radial menu for virtual object manipulation. *Proceedings - IEEE Virtual Reality*, 2016-July:169–170, 2016. doi: 10.1109/VR.2016.7504707. [23](#)
- [88] Andy Wu, Derek Reilly, Anthony Tang, and Ali Mazalek. Tangible Navigation and Object Manipulation in Virtual Environments. *Environment*, pages 37–44, 2011. doi: 10.1145/1935701.1935710. URL <http://portal.acm.org/citation.cfm?id=1935710>. [23](#)
- [89] Lung-Pan Cheng, Thijs Roumen, Hannes Rantzsch, Sven Köhler, Patrick Schmidt, Robert Kovacs, Johannes Jasper, Jonas Kemper, and Patrick Baudisch. TurkDeck: Physical Virtual Reality Based on People. *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology - UIST '15*, pages 417–426, 2015. doi: 10.1145/2807442.2807463. URL <http://dl.acm.org/citation.cfm?id=2807442.2807463>. [23](#)
- [90] Paulo Dias and S Eliseu. Gesture Interactions for Virtual Immersive Environments: Navigation, Selection and Manipulation. 9740:211–221, 2016. doi: 10.1007/978-3-319-39907-2. URL <http://link.springer.com/10.1007/978-3-319-39907-2>. [23](#)
- [91] Vijayakumar Nanjappan, Hai Ning Liang, Feiyu Lu, Konstantinos Papangelis, Yong Yue, and Ka Lok Man. User-elicited dual-hand interactions for manipulating 3D objects in virtual reality environments. *Human-centric Computing and Information Sciences*, 8(1):31, dec 2018. ISSN 21921962. doi: 10.1186/s13673-018-0154-5. URL <https://hcis-journal.springeropen.com/articles/10.1186/s13673-018-0154-5>. [23](#)

- [92] Merwan Achibet. Contributions to the Design of Novel Hand-based Interaction Techniques for Virtual Environments. *Graphics [cs.GR]. INSA de Rennes*, dec 2015. URL <https://tel.archives-ouvertes.fr/tel-01312186/https://tel.archives-ouvertes.fr/tel-01312186/document>. 23
- [93] T. Hongyong and Y. Youling. Finger tracking and gesture recognition with kinect. In *2012 IEEE 12th International Conference on Computer and Information Technology*, pages 214–218, Oct 2012. doi: 10.1109/CIT.2012.62. 25
- [94] J. Rekimoto. GestureWrist and GesturePad: unobtrusive wearable interaction devices. *Proceedings Fifth International Symposium on Wearable Computers*, pages 21–27. ISSN 1530-0811. doi: 10.1109/ISWC.2001.962092. URL <http://ieeexplore.ieee.org/document/962092/>. xix, 28
- [95] Lee Seon-Woo and K Mase. Activity and location recognition using wearable sensors. *Pervasive Computing, IEEE*, 1(3):24–32, 2002. 28
- [96] Andrew Wilson and Steven Shafer. XWand: UI for intelligent spaces. *Proceedings of the conference on Human factors in computing systems - CHI '03*, page 545, 2003. doi: 10.1145/642611.642706. URL <http://doi.acm.org/10.1145/642611.642706>{%}5Cn<http://portal.acm.org/citation.cfm?doid=642611.642706>. 28
- [97] Juha Kela, Panu Korpipää, Jani Mäntyjärvi, Sanna Kallio, Giuseppe Savino, Luca Jozzo, and Sergio Di Marca. Accelerometer-based gesture control for a design environment. *Personal and Ubiquitous Computing*, 10(5):285–299, 2006. ISSN 16174909. doi: 10.1007/s00779-005-0033-8. 28
- [98] Toshiki Iso and Kenichi Yamazaki. Gait analyzer based on a cell phone with a single three-axis accelerometer. *Proceedings of the 8th conference on Human-computer interaction with mobile devices and services - MobileHCI '06*, 12(15):141, 2006. doi: 10.1145/1152215.1152244. URL <http://dl.acm.org/citation.cfm?id=1152215.1152244>. 28
- [99] Abu Saleh Md Mahfujur Rahman, M Anwar Hossain, and Abdulmotaleb El Saddik. Spatial-geometric approach to physical mobile interaction based on

- accelerometer and IR sensory data fusion. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 6(4):1–23, 2010. ISSN 15516857. doi: 10.1145/1865106.1865112. URL <http://portal.acm.org/citation.cfm?doid=1865106.1865112>. 28
- [100] Sven Kratz, Michael Rohs, and Georg Essl. Combining acceleration and gyroscope data for motion gesture recognition using classifiers with dimensionality constraints. *Proceedings of the 2013 international conference on Intelligent user interfaces - IUI '13*, page 173, 2013. doi: 10.1145/2449396.2449419. URL <http://dl.acm.org/citation.cfm?doid=2449396.2449419>. 28
- [101] Xiang 'Anthony' Chen, Tovi Grossman, Daniel J. Wigdor, and George Fitzmaurice. Duet: exploring joint interactions on a smart phone and a smart watch. *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*, pages 159–168, 2014. ISSN 07300301. doi: 10.1145/2556288.2556955. URL <http://dl.acm.org/citation.cfm?doid=2556288.2556955>. 28
- [102] Kurt Partridge, Saurav Chatterjee, Vibha Sazawal, Gaetano Borriello, and Roy Want. TiltType: Accelerometer-Supported Text Entry for Very Small Devices. *Proceedings of the 15th annual ACM symposium on User interface software and technology - UIST '02*, 4(2):201–204, 2002. doi: 10.1145/571985.572013. URL <http://portal.acm.org/citation.cfm?doid=571985.572013>. xix, 29
- [103] E Jones, J Alexander, A Andreou, P Irani, and S Subramanian. GesText: Accelerometer-Based Gestural Text-Entry Systems. In *Proc. CHI 2010* (pp. 2173–2182). pages 2173–2182, 2010. doi: 10.1145/1753326.1753655. URL <http://dx.doi.org/10.1145/1753326.1753655>. 29
- [104] Mayank Goel, Leah Findlater, and Jacob Wobbrock. WalkType: Using Accelerometer Data to Accomodate Situational Impairments in Mobile Touch Screen Text Entry. *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*, pages 2687–2696, 2012. doi: 10.1145/2207676.2208662. URL <http://dl.acm.org/citation.cfm?doid=2207676.2208662>. 29

- [105] Ken Iwasaki, Takashi Miyaki, and Jun Rekimoto. Expressive Typing: A New Way to Sense Typing Pressure and Its Applications. *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems - CHI EA '09*, pages 4369–4374, 2009. doi: 10.1145/1520340.1520668. URL <http://portal.acm.org/citation.cfm?doid=1520340.1520668>. 29
- [106] Dominik Schmidt, Fadi Chehimi, Enrico Rukzio, and Hans Gellersen. Phone-Touch: a technique for direct phone interaction on surfaces. *Proceedings of the 23rd annual ACM symposium on User interface software and technology - UIST '10*, page 13, 2010. doi: 10.1145/1866029.1866034. URL <http://portal.acm.org/citation.cfm?doid=1866029.1866034>. 29
- [107] William Hutama, Peng Song, Chi-Wing Fu, and Wooi Boon Goh. Distinguishing multiple smart-phone interactions on a multi-touch wall display using tilt correlation. *Proceedings of the 2011 annual conference on Human factors in computing systems - CHI '11*, page 3315, 2011. doi: 10.1145/1978942.1979433. URL <http://dl.acm.org/citation.cfm?doid=1978942.1979433>. 30
- [108] Ken Hinckley and Hyunyoung Song. Sensor Synaesthesia : Touch in Motion , and Motion in Touch. *Human Factors*, pages 801–810, 2011. doi: 10.1145/1978942.1979059. URL <http://portal.acm.org/citation.cfm?id=1979059>. 30
- [109] Seongkook Heo and Geehyuk Lee. ForceTap: Extending the Input Vocabulary of Mobile Touch Screens by adding Tap Gestures. *MobileHCI*, pages 113–122, 2011. doi: 10.1145/2037373.2037393. URL <http://dl.acm.org/citation.cfm?doid=2037373.2037393>. 30
- [110] Mayank Goel, Jacob Wobbrock, and Shwetak Patel. GripSense: Using Built-in Sensors to Detect Hand Posture and Pressure on Commodity Mobile Phones. *Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST '12*, pages 545–554, 2012. doi: 10.1145/2380116.2380184. URL <http://dl.acm.org/citation.cfm?doid=2380116.2380184>. 30
- [111] M Serrano, Lecolinet, and Guiard. Bezel-Tap gestures: quick activation of commands from sleep mode on tablets. *Proceedings of*

- CHI 2013*, pages 3027–3036, 2013. doi: 10.1145/2470654.2481421. URL <http://dl.acm.org/citation.cfm?id=2481421>
<http://Users/grahamwilson/Documents/Papers/2013/Serrano/Serrano2013ProceedingsofCHI2013.pdf>
<http://c80d98e4-9a96-4487-8d06-8e1acc780d86/Paper/p10863>. 30
- [112] Off-the-shelf Smartphone, Cheng Zhang, Aman Parnami, Caleb Southern, Edison Thomaz, Gabriel Reyes, Rosa I Arriaga, and Gregory D Abowd. BackTap : Robust Four-Point Tapping on the Back of an. *Uist*, pages 111–112, 2013. doi: 10.1145/2508468.2514735. 30
- [113] Ian Oakley, DoYoung Lee, MD. Rasel Islam, and Augusto Esteves. Beats: Tapping Gestures for Smart Watches. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*, pages 1237–1246, 2015. doi: 10.1145/2702123.2702226. URL <http://dl.acm.org/citation.cfm?doid=2702123.2702226>. 30
- [114] Chris Harrison, Julia Schwarz, and Scott E. Hudson. TapSense. *Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST '11*, page 627, 2011. doi: 10.1145/2047196.2047279. URL <http://dl.acm.org/citation.cfm?doid=2047196.2047279>. 30
- [115] C.H.A.N.G.C. C. and L.I.N.C. J. LIBSVM: A Library for support vector machines, 2011. URL <https://www.csie.ntu.edu.tw/~cjlin/libsvm/>. 34
- [116] Rob Jacob and Sophie Stellmach. What you look at is what you get. *Interactions*, 23(5):62–65, 2016. ISSN 10725520. doi: 10.1145/2978577. URL <http://dl.acm.org/citation.cfm?doid=2991131.2978577>. 45
- [117] Oculus Patented Eye Tracking Device Days After Acquiring The Eye Tribe. URL <https://uploadvr.com/oculus-patented-new-eye-tracking-device-days-acquiring-eye-tribe/>. 48
- [118] Bonkon Koo, Hwan-Gon Lee, Yunjun Nam, and Seungjin Choi. Immersive BCI with SSVEP in VR head-mounted display. In *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology*

- Society (EMBC)*, pages 1103–1106. IEEE, aug 2015. ISBN 978-1-4244-9271-8. doi: 10.1109/EMBC.2015.7318558. URL <http://ieeexplore.ieee.org/document/7318558/>. 48
- [119] Thies Pfeiffer, Marc E Latoschik, Ipke Wachsmuth, and J Herder. Evaluation of binocular eye trackers and algorithms for 3d gaze interaction in virtual reality environments. *Journal of Virtual Reality and Broadcasting*, 5(16):16, 2008. ISSN 1860-2037. doi: urn:nbn:de:0009-6-16605. URL <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.140.4757{&}rep=rep1{&}type=pdf>. 48
- [120] Robert J. K. Jacob. What you look at is what you get: eye movement-based interaction techniques. *Proceedings of the SIGCHI conference on Human factors in computing systems Empowering people - CHI '90*, pages 11–18, 1990. ISSN 07366906. doi: 10.1145/97243.97246. URL <http://portal.acm.org/citation.cfm?doid=97243.97246>. 48
- [121] Anke Huckauf and Mario H. Urbina. Object selection in gaze controlled systems. *ACM Transactions on Applied Perception*, 8(2):1–14, 2011. ISSN 15443558. doi: 10.1145/1870076.1870081. URL <http://portal.acm.org/citation.cfm?doid=1870076.1870081>. 48
- [122] Howell Istance, Richard Bates, Aulikki Hyrskykari, and Stephen Vickers. Snap clutch, a moded approach to solving the Midas touch problem. In *Proceedings of the 2008 symposium on Eye tracking research & applications - ETRA '08*, page 221, New York, New York, USA, 2008. ACM Press. ISBN 9781595939821. doi: 10.1145/1344471.1344523. URL <http://portal.acm.org/citation.cfm?doid=1344471.1344523>. 48
- [123] R. Jacob and S. Stellmach. Interaction technologies: What you look at is what you get: Gaze-based user interfaces. *Interactions*, 23(5):62–65, 2016. ISSN 15583449. doi: 10.1145/2978577. 48
- [124] Pawel Kasprowski, Katarzyna Harezlak, and Michal Niezabitowski. Eye movement tracking as a new promising modality for human computer interaction. *Proceedings of the 2016 17th International Carpathian Control Conference, ICC 2016*, pages 314–318, 2016. doi: 10.1109/CarpathianCC.2016.7501115. 49

- [125] Augusto Esteves, Eduardo Velloso, Andreas Bulling, and Hans Gellersen. Orbits. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology - UIST '15*, pages 457–466, 2015. ISBN 9781450337793. doi: 10.1145/2807442.2807499. URL https://perceptual.mpi-inf.mpg.de/files/2015/09/Esteves_{_}UIST15.pdf<http://dl.acm.org/citation.cfm?doid=2807442.2807499>. 49
- [126] Roman Bednarik, Hana Vrzakova, and Michal Hradis. What do you want to do next. In *Proceedings of the Symposium on Eye Tracking Research and Applications - ETRA '12*, page 83, 2012. ISBN 9781450312219. doi: 10.1145/2168556.2168569. URL http://delivery.acm.org.ezlibproxy1.ntu.edu.sg/10.1145/2170000/2168569/p83-bednarik.pdf?ip=155.69.24.171{id=2168569{&}acc=ACTIVESERVICE{&}key=FF6731C4D3E3CFFF.B51AA0289512DB67.4D4702B0C3E38B35.4D4702B0C3E38B35{&}{_}{_}acm{_}{_}=1529679669{_-}226d925e66e771f2fa2102ba248be. 49
- [127] Manu Kumar, Andreas Paepcke, and Terry Winograd. EyePoint. *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '07*, page 421, 2007. doi: 10.1145/1240624.1240692. URL <http://portal.acm.org/citation.cfm?doid=1240624.1240692>. 49
- [128] Thammathip Piumsomboon, Gun Lee, Robert W. Lindeman, and Mark Billingham. Exploring natural eye-gaze-based interaction for immersive virtual reality. In *2017 IEEE Symposium on 3D User Interfaces, 3DUI 2017 - Proceedings*, pages 36–39, 2017. ISBN 9781509067169. doi: 10.1109/3DUI.2017.7893315. 49
- [129] Mohamed Khamis, Carl Oechsner, Florian Alt, and Andreas Bulling. VRpursuits: Interaction in virtual reality using smooth pursuit eye movements. In *Proceedings of the 2018 International Conference on Advanced Visual Interfaces - AVI '18*, pages 1–8, New York, New York, USA, 2018. ACM Press. ISBN 9781450356169. doi: 10.1145/3206505.3206522. URL <http://dl.acm.org/citation.cfm?doid=3206505.3206522><https://doi.org/10.1145/3206505.3206522>. 49
- [130] Abdul Moiz Penkar, Christof Lutteroth, and Gerald Weber. Designing for the eye: design parameters for dwell in gaze interaction. *Proceedings of the 24th*

- Australian Computer-Human Interaction Conference*, pages 479–488, 2012. doi: 10.1145/2414536.2414609. 54
- [131] Haruhisa Kato and Hiromasa Yanagihara. Pacman ui: Vision-based finger detection for positioning and clicking manipulations. In *Proceedings of the 15th International Conference on Human-computer Interaction with Mobile Devices and Services*, MobileHCI '13, pages 464–467, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2273-7. doi: 10.1145/2493190.2494652. URL <http://doi.acm.org/10.1145/2493190.2494652>. 67
- [132] OnePlus 5 Specs - OnePlus (United States). URL <https://www.oneplus.com/5/specs>. 77, 110
- [133] Silvertec VR Glasses. URL <http://www.silvertec.com/in-vr002>. 77, 110
- [134] KOLMOGOROV and A. Sulla determinazione empirica di una lgge di distribuzione. *Inst. Ital. Attuari, Giorn.*, 4:83–91, 1933. URL <https://ci.nii.ac.jp/naid/10010480527/>. 84, 101, 111
- [135] Zhaorui Chen, Jinzhu Li, Yifan Hua, Rui Shen, and Anup Basu. Multimodal interaction in augmented reality. In *2017 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2017*, volume 2017-Janua, pages 206–209, 2017. ISBN 9781538616451. doi: 10.1109/SMC.2017.8122603. 91
- [136] Arjan J.F. Kok and Robert van Liere. A multimodal virtual reality interface for 3D interaction with VTK. *Knowledge and Information Systems*, 13(2): 197–219, oct 2007. ISSN 02191377. doi: 10.1007/s10115-007-0066-6. URL <http://link.springer.com/10.1007/s10115-007-0066-6>. 91
- [137] Wan-Lun Tsai, You-Lun Hsu, Chi-Po Lin, Chen-Yu Zhu, Yu-Cheng Chen, and Min-Chun Hu. Immersive virtual reality with multimodal interaction and streaming technology. In *Proceedings of the 18th ACM International Conference on Multimodal Interaction - ICMI 2016*, pages 416–416, New York, New York, USA, 2016. ACM Press. ISBN 9781450345569. doi: 10.1145/2993148.2998526. URL <http://dl.acm.org/citation.cfm?doid=2993148.2998526>. 91
- [138] Eric Foxlin and Michael Harrington. WearTrack. Technical report. URL <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.135.230{&}rep=rep1{&}type=pdf>. 92

-
- [139] Huidong Bai, Alaeddin Nassani, Barrett Ens, and Mark Billinghurst. Asymmetric Bimanual Interaction for Mobile Virtual Reality. In *ICAT-EGVE 2017 - International Conference on Artificial Reality and Telexistence and Eurographics Symposium on Virtual Environments*, pages 83–86, 2017. ISBN 978-3-03868-038-3. doi: 10.2312/egve.20171343. URL <https://diglib.eg.org/handle/10.2312/egve20171343>. 92
- [140] Didier Casalta, Yves Guiard, and Michel Beaudouin Lafon. Evaluating two-handed input techniques. In *CHI '99 extended abstracts on Human factors in computing systems - CHI '99*, page 236, New York, New York, USA, 1999. ACM Press. ISBN 1581131585. doi: 10.1145/632716.632862. URL <http://portal.acm.org/citation.cfm?doid=632716.632862>. 92
- [141] Anupam Agrawal, Rohit Raj, and Shubha Porwal. Vision-based multimodal human-computer interaction using hand and head gestures. In *2013 IEEE CONFERENCE ON INFORMATION AND COMMUNICATION TECHNOLOGIES*, pages 1288–1292. IEEE, apr 2013. ISBN 978-1-4673-5758-6. doi: 10.1109/CICT.2013.6558300. URL <http://ieeexplore.ieee.org/document/6558300/>. 116
- [142] Tilt Brush by Google. URL <https://www.tiltbrush.com/>. 121
- [143] Oad Percy. Iris localization using Daugman ' s algorithm. *Blekinge Institute of Technology*, pages 1–48. 130