Dynamic Motion Coupling of Body Movement for Input Control

Christopher Clarke

M.Sc Computer Science, Lancaster University, U.K.

B.Eng (Hons) Computer Systems Engineering, Lancaster University, U.K.

School of Computing and Communications

Lancaster University, UK

Thesis submitted for the degree of Doctor of Philosophy

December 2019

Dynamic Motion Coupling of Body Movement for Input Control

Christopher Clarke
M.Sc Computer Science, Lancaster University, U.K.
B.Eng (Hons) Computer Systems Engineering, Lancaster University, U.K.

Submitted for the degree of Doctor of Philosophy December 2019

Abstract

Touchless gestures are used for input when touch is unsuitable or unavailable, such as when interacting with displays that are remote, large, public, or when touch is prohibited for hygienic reasons. Traditionally user input is spatially or semantically mapped to system output, however, in the context of touchless gestures these interaction principles suffer from several disadvantages including memorability, fatigue, and ill-defined mappings. This thesis investigates motion correlation as the third interaction principle for touchless gestures, which maps user input to system output based on spatiotemporal matching of reproducible motion. We demonstrate the versatility of motion correlation by using movement as the primary sensing principle, relaxing the restrictions on how a user provides input. Using TraceMatch, a novel computer vision-based system, we show how users can provide effective input through investigation of input performance with different parts of the body, and how users can switch modes of input spontaneously in realistic application scenarios. Secondly, spontaneous spatial coupling shows how motion correlation can bootstrap spatial input, allowing any body movement, or movement of tangible objects, to be appropriated for ad hoc touchless pointing on a per interaction basis. We operationalise the concept in MatchPoint, and demonstrate the unique capabilities through an exploration of the design space with application examples. Finally, we explore how users synchronise with moving targets in the context of motion correlation, revealing how simple harmonic motion leads to better synchronisation. Using the insights gained we explore the robustness of algorithms used for motion correlation, showing how it is possible to successfully detect a user's intent to interact whilst suppressing accidental activations from common spatial and semantic gestures. Finally, we look across our work to distil guidelines for interface design, and further considerations of how motion correlation can be used, both in general and for touchless gestures.

Declaration

This thesis is a presentation of my original research work. No part of this thesis has been submitted for another degree or qualification. The work was done under the guidance of Hans Gellersen at Lancaster University.

Author: **Christopher Clarke**

Acknowledgements

Firstly, I would like to express my sincere gratitude to my advisor Prof. Hans Gellersen for the continuous support of my Ph.D study and related research, for his patience, motivation, and immense knowledge. His guidance helped me throughout the research and writing of this thesis. I could not have imagined having a better advisor and mentor for my Ph.D study.

I am indebted to my fellow lab mates (both old and new) for the stimulating discussions, for the sleepless nights we were working together before deadlines, and for all the fun we have had in the last four years. I'd like to thank all those who organised and attended the Summer School on Computational Interaction in the summer of 2017. This was a fantastic experience to meet you all and engage in intellectually stimulating discussions.

My sincere thanks also goes to Dr. Don Kimber, Dr. Patrick Chiu, and Dr. Yanxia Zhang, and everyone at FXPAL, who provided me an opportunity to join their team as an intern, and welcomed me with open arms. The experience was one I will truly cherish for the rest of my life.

Most importantly, I would like to thank my family: Maria who has been my rock throughout this journey and provided me with endless support and encouragement; my parents for always believing in me, and without whom I would have not had the courage to embark on this journey; and my sister and brother-in-law who have always been there to support me and provide a caring ear to listen. I have gained so many friends along the way, but unfortunately I have also lost two of the closest people to me. This thesis is dedicated to my grandparents, John and Jean Robinson, who throughout my life have been guiding lights, with their advice, support, and caring. I owe more to them than I could ever express.

Publications

This work has been published in peer-reviewed publications at conferences. Below are the references of these publications, and the associated chapters to which they relate.

- **Chapter 3:** Christopher Clarke, Alessio Bellino, Augusto Esteves, Eduardo Velloso, and Hans Gellersen. 2016. TraceMatch: a computer vision technique for user input by tracing of animated controls. In Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '16). ACM, New York, NY, USA, 298-303. DOI: https://doi.org/10.1145/2971648.2971714
- **Chapter 3:** Christopher Clarke, Alessio Bellino, Augusto Esteves, and Hans Gellersen. 2017. Remote Control by Body Movement in Synchrony with Orbiting Widgets: an Evaluation of TraceMatch. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. 1, 3, Article 45 (September 2017), 22 pages. DOI: https://doi.org/10.1145/3130910
- **Chapter 4:** Christopher Clarke and Hans Gellersen. 2017. MatchPoint: Spontaneous Spatial Coupling of Body Movement for Touchless Pointing. In Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17). ACM, New York, NY, USA, 179-192. DOI: https://doi.org/10.1145/3126594.3126626
- Eduardo Velloso, Marcus Carter, Joshua Newn, Augusto Esteves, Christopher Clarke, and Hans Gellersen. 2017. Motion Correlation: Selecting Objects by Matching Their Movement. ACM Trans. Comput.-Hum. Interact. 24, 3, Article 22 (April 2017), 35 pages. DOI:https://doi.org/10.1145/3064937

Contents

List of Figures

List of Tables

1

Introduction

One of the most common forms of human expression are gestures - intentional movements of the limbs or body used to communicate a message. They are a natural form of expression and human communication which we use in everyday life. For the purpose of interaction, touchless gestures are body movements which are used to provide input without direct manipulation of an input device, and may be sensed remotely or using body-worn sensors. Touchless gestures can come in many shapes and forms - from a simple pose of the hand, to distinct movements of the head. Information from a gesture can be encoded in the movement or pose of a body part (e.g. a closed fist in the case of a hand). Poses may occur statically at a single point in time, or dynamically as it varies over time at either a fixed spatial position or over a motion path. In the field of human computer interaction, the use of touchless gestures to express input has been studied for decades [138].

By utilising motor skills developed from an early age, and our innate awareness of body movement, gestures have the potential to create interfaces with increased learnability and general ease-of-use [115]. They are compelling for interaction for a number of reasons when touch is not available. Their "come as you are" nature does not require users to seek out external devices, allowing for spontaneous in-situ interaction that is always available and can be invoked from multiple positions. Multiple users can have simultaneous control for relaxed, intuitive interaction in a lean-back manner. They can be used for remote interaction at a distance with out-of-reach displays, or with medium to large displays where both viewing and interacting with the display is not possible at close range. All of this can be achieved whilst maintaining sterility and can be executed in an eyes-free manner due to proprioception, our own awareness of how our body is positioned and moves.

Traditionally, when mapping user input to system output there are two dominant paradigms for touchless gestures: semantic and spatial [73]. Semantically mapped gestures involve the recognition of a pre-defined set of gestures which are mapped to specific commands in an interface. In contrast, spatially mapped interactions are conventionally based on a rigid configuration of a display coupled with a pointing device. This is often operationalised in a graphical user interface (GUI) such as the familiar desktop paradigm of point-and-select. Sometimes, these paradigms are combined in an interface, such as when semantically mapped gestures are used to initiate the pointing or manipulation mechanism, or to confirm selection. However, touchless gestures suffer from a number of core usability problems identified by the human computer interaction

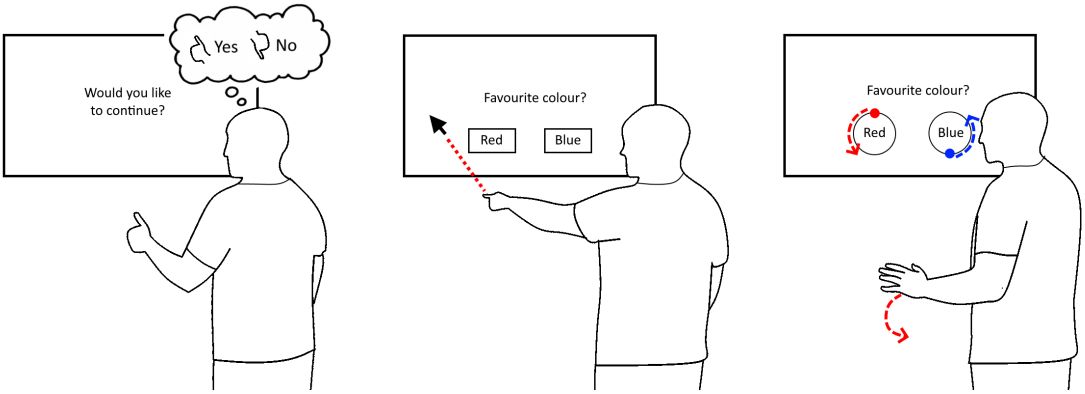

Figure 1.1: Three main metaphors for touchless gestures. Left: Library-based uses a pre-defined set of gestures mapped to input commands, often requiring users to remember which gesture corresponds to which command. Centre: Cursor-based metaphor uses pointing and the familiar cursor paradigm for selection, but can cause fatigue known as "gorilla arm". Right: Motion correlation uses corresponding motions to determine a user's intent to interact, with selection occurring when the user synchronises their motion with an on-screen widget.

community [19] – from context-dependent issues such as embarrassment [4] or ill-informed gesture mappings [188], to more general usability issues such as fatigue [101] or the midas touch (a.k.a. "live mic") problem [286].

1.1 Motion Correlation

In this thesis, we investigate a third interaction paradigm for touchless gestures which draws upon our innate human ability to mimic and synchronise with external stimuli – *motion correlation*. In contrast to spatial and semantic input, motion correlation relies on spatiotemporal matching of objects available for interaction and the user's motion, see Figure 1.1. The core principle of motion correlation is that objects available for interaction are revealed through distinct motions, and a user's intent to interact is inferred through movement which corresponds to the desired object. Velloso et al. define the principle of motion correlation by three properties [268]:

1. Motions are displayed to the user which represent actions;
2. A user signals their intent to interact by following the motion which corresponds to the desired action;
3. The system compares its output (the displayed motions) with its input (the user's movement) to determine selection.

Mimicking motion on the interface was first suggested as a strategy for selection by Williamson and Murray-Smith in their pioneering work on *"Pointing without a Pointer"* [289]. Their work was grounded in the notion of *continuous interaction*, in which the exchange of information between a user and a computing system is continuous at a relatively high resolution, in contrast to the exchange of discrete messages [60]. Using this principle, the interfaces they created consisted of agents which moved in unique ways, and the system determined selection of an agent by looking for user responses which correlated with the movement of the agents. Fekete et al. built upon this by developing *Motion-pointing*, which adopted the technique for selection in a traditional graphical user interface by using a mouse to match elliptical trajectories [73]. These works provide the foundation of motion correlation as a viable alternative to spatial and semantic input techniques.

Later work on smooth pursuit eye movements demonstrated how motion correlation is suitable for application in a wide variety of contexts, from smart watches to public displays [273, 69]. Motion correlation in the context of eye tracking offers unique advantages because spatial and semantic mapping of gaze as input is much more problematic than other modalities due to the physiological nature of eye movements, and the inherent sensing issues of eye trackers. These works also demonstrated how motion correlation can be used in holistic interface designs, and have since inspired the use of other input modalities, such as touchless gestures [40].

1.2 Research Questions

The overarching goal of this thesis is to position motion correlation as the third fundamental interaction paradigm for touchless gestures. To achieve this, we aim to advance the understanding of the design space and provide a clear foundation to build upon by understanding the underlying mechanisms of how we match against motion. With this in mind, we ask four fundamental questions of motion correlation in the context of touchless gestures:

RQ1: *Can movement be used as the primary sensing principle for interaction?* The concept of motion correlation has been used for different input modalities, from a mouse [73], to gaze [273], to hand movements [40]. However, the underlying concept of motion correlation requires movement, which can be generated from many sources. In particular, we investigate:

- **RQ1.1:** *Is it feasible to harness movement as the primary sensing principle using motion correlation to allow users to provide input more flexibly?* With advances in computer vision, detection of motion using optical flow is readily available. We therefore investigate the practicality of how generic motion can be used as input.
- **RQ1.2:** *How effective are users at providing input using different body parts, or when holding objects, to generate the required motion?* Users can express their intent to interact with different body movements, or whilst using objects. Understanding how adept users are at providing input under different conditions can help inform interface design.
- **RQ1.3:** *How do users provide input spontaneously when given the opportunity to interact with different body parts and objects?* The flexibility of how to provide input is only useful if users take advantage of it. This question looks to address how users interact when given the flexibility to provide input using any means necessary.

RQ2: *How can motion correlation be used to bootstrap spatial pointing?* Previous work has focussed on motion correlation for discrete selection of objects, however many interactions in the real-world require more precise input control. We explore how motion correlation can be extended to better support a wider range of input for interaction with touchless gestures, with a particular focus on spatial gestures. Subsequently, we investigate:

- **RQ2.1:** *Can the control-display gain for spatial interaction be derived from motion correlation?* The matching of motions has application beyond selection, such as for pairing [168], and calibration [199] of input devices. For spatial interactions, mapping user input to output is non-trivial, and we explore whether motion correlation presents a unique opportunity to do this on a per-interaction basis.
- **RQ2.2:** *What unique capabilities arise from combining motion correlation with spatial coupling?* Motion correlation has distinct advantages compared with spatial input, and vice versa, which we discuss in chapter 2. By combining these two interaction principles, we investigate if we can leverage both of their advantages for unique interaction techniques.

- **RQ2.3:** *What are the usability implications of dynamically setting the control display gain?* Using motion correlation for dynamically assigning control-display gains has potential ramifications on the user's ability to consistently provide input accurately. We therefore investigate how precisely and accurately users can provide input when motion correlation is combined with spatial interaction.

RQ3: *How do users reproduce trajectories in the context of motion correlation?* Our ability to synchronise with external stimuli is dependent on a number of physiological subsystems working in coordination. There is little known about how users actually synchronise with target motion in the context of motion correlation as a selection technique. To establish a deeper understanding, we address the following sub-questions:

- **RQ3.1:** *How can we extract the ground truth of how well users follow a target, independent of any particular detection mechanism intended for interaction?* In the context of motion correlation, the underlying assumption of how well we can synchronise has been based on intuition, or has been inferred through detection rates of sensing techniques. To assess how well people follow motion, one must look at user ability independent of detection algorithms.
- **RQ3.2:** *How much do users lead or lag a target when synchronising with different shapes, and across different input methods?* Unlike other input modalities, a user's ability to synchronise with touchless gestures is based on sensorimotor synchronisation. A deeper understanding of how users follow targets can be used to inform algorithm and interface design.
- **RQ3.3:** *Can simple harmonic movement better help users synchronise with target movement?* Designing motions that users are more easily able to follow is currently underexplored. We investigate whether using simple harmonic motion for target movement helps users synchronise with external stimuli.

RQ4: *How robustly can we detect users matching motions?* The success of an input technique can be decided on its ability to accurately detect user intentions, and reject accidental activations. This is dependent on both the user's ability to provide accurate input to the system, and the system's ability to accurately detect said input. We look towards optimising the detection process by answering two sub-questions:

- **RQ4.1:** *Which algorithms are appropriate for detecting motion correlation?* Motion correlation can be viewed as a trajectory matching problem, where the trajectory of the user is matched against the trajectory of the target's motion. There is well-established work beyond the HCI literature that looks at matching and indexing spatiotemporal trajectories (e.g. [282]), and which can be applied for detection in the context of motion correlation.
- **RQ4.2:** *What are the best parameterisations for achieving optimal performance across different types of input?* Algorithms have multiple parameterisations that can influence the detection rate across different target conditions, and with different methods of input. The context in which motion correlation is used will also result in different considerations for whether to optimise selection time, or robustness to false activations as a priority.

1.3 Methodology

In order to answer the proposed research questions we iteratively design, develop and evaluate systems and algorithms. Where appropriate, we use real-world examples to explore the design space, and ground our work with empirical data gained from data collections.

We begin with **RQ1**, investigating how motion can be used as the primary sensing input. We use a build methodology to develop and build TraceMatch – a system which uses computer vision techniques to accept generic motion as input. Based on a data collection of participants matching motion, we empirically test how robust the system is at detecting and matching user movement, and how robust it is to rejecting accidental selections. We extract preliminary user performance metrics to gain initial insights, which help us to formulate and refine our approach in the next phase – the user evaluation.

To evaluate how well users match motion in an interactive setting, we conduct a user study with TraceMatch using a mixed-methods approach. As touchless gestures have many applications, we ground our user study on the perennial problem of remote control to assess the system in context. We collect quantitative performance data of users using the system to gain insights into user performance, whilst collecting qualitative preference data with different types of input using an abstract task. We then investigate the system in a pseudo real-world setting designed to allow laid-back, spontaneous interactions in a real-world context. These were used to investigate participant's spontaneous choices for interaction, and to explore whether the technique could be extended to multi-level input.

Identification of research gaps in the previous study informed the methodology for the investigation into how expressive motion correlation can be (**RQ2**). We introduce *spontaneous spatial coupling* – an input technique which combines motion correlation with cursor-based interaction to support continuous interaction. This allows users to create and bind to pointers on-demand using any part of their body, with motion correlation as the underpinning interaction principle. We first discuss the underlying principles of the technique, and design considerations. In order to study this technique, we operationalise it by developing MatchPoint – a system which builds upon TraceMatch. We use MatchPoint to explore the design space surrounding the unique opportunities afforded by spontaneous spatial coupling. We ground our exploration with real-world application examples. We then evaluate how adept users are with the interaction technique by performing a lab-based user study on multi-directional pointing performance using the ISO 9241-9 standard.

Finally, we conduct a data collection to gain deeper insights into how users match against different motions using different types of input (**RQ3**). Analysis is performed using offline post-processing techniques to extract the ground truth and provide insight into how quickly users start to gesture, how well they can maintain synchrony with the moving targets, and how large their movements are. We use this analysis to inform algorithm design into the detection of matching gestures (**RQ4**). We draw on literature in sensorimotor synchronisation to inform our study design and analysis, and test algorithms from both the motion correlation and time-series comparison literature.

1.4 Contributions

This thesis advances the understanding of motion correlation by demonstrating what the coupling of motion can support in the context of interaction using clear principles. In particular, we make the following contributions:

- **Demonstration of motion as a sensing principle which allows users to provide input flexibly (RQ1):** We provide TraceMatch as a systems contribution to harness the universality of motion using computer vision techniques with a general purpose webcam (RQ1.1). TraceMatch is an input method which abstracts from body part segmentation, and relaxes the restrictions about how a user must provide input, and in what position they must be to interact with the system. We also validate TraceMatch, showing that users are

effective at selecting input by synchronising with displayed motion using different types of movement. We provide deeper insights into how different ways of performing matching motion (with head, hand, or while holding objects) affect performance (RQ1.2), and into preferences and spontaneous choices of different body movements for input (RQ1.3). Finally, we explore ways in which the technique can be extended from binary selection to multi-level input displayed within one orbiting widget.

- **Demonstration that motion correlation is more than just a selection principle (RQ2):** We introduce spontaneous spatial coupling – a formalisation of how motion correlation can bootstrap cursor-based interaction for more expressive interaction (RQ2). Rather than relying on rigid one-to-one mapping of user input to pointer, we leverage the properties of motion to go beyond just discrete selection. In the context of touchless gestures we show how motion correlation bootstraps spatial input by selecting the functionality of the pointing instance, whilst implicitly determining the control-display gain and appropriate mapping for the spatial input phase (RQ2.1). We contribute MatchPoint, a system which operationalises this concept using computer vision techniques. Using MatchPoint we explore the design space that emerges from looking beyond motion correlation for just discrete selection, and demonstrate unique opportunities using practical examples, including multiple, simultaneous pointing instances and creation of ad-hoc tangible interfaces (RQ2.2). Finally, we provide insights into how users can use the technique with different methods of input using a multi-directional pointing performance using the ISO 9241-9 standard (RQ2.3).

- **Deeper insights into how users match moving targets (RQ3):** We provide an in-depth look at how users match against moving targets in the context of motion correlation using body movements. We develop techniques for extracting the ground truth of user's synchronicity against circular and line-based targets (RQ3.1). Using empirical data gained from a data collection with participants, we investigate how closely they can synchronise with targets across different conditions (RQ3.2), and look into which type of movement they deem more favourable to inform interface design. Informed by prior work on hand movements for reciprocal pointing tasks [94], we design movement based on simple harmonic motion and demonstrate how this helps users to be more closely synchronised (RQ3.3).

- **Demonstration of the robustness of motion correlation (RQ4):** By viewing motion correlation as a trajectory matching problem, we provide an in-depth evaluation of similarity-based algorithms which shows that motion correlation is a robust technique that can have very high detection rates ($\approx$100%) with zero detections as a result of accidental user movement (RQ4.1). We provide parameters and thresholds for motion correlation practitioners (RQ4.2).

- **Guidelines for the creation of motion correlation interfaces with touchless gestures:** From the insights gained across the studies, we distil guidelines to inform interface design, covering which shapes and speeds to use in which contexts, how many simultaneous targets could be presented, and where best to position them on the interface.

1.5 Thesis Structure

The thesis is structured as follows:

Chapter 2 begins by discussing touchless gestures as "natural" interaction methods, and the different ways in which users can provide input. We then classify touchless gestures into two

distinct categories: semantic and spatial, and discuss the advantages and disadvantages of both. We use this to position motion correlation as a third interaction principle for touchless gestures and discuss the underlying theories from the sensorimotor synchronisation literature that reveal how adept we are at following motions. We then explore existing works that use motion correlation as an input principle. Techniques for sensing touchless gestures, and traditional gesture recognition pipelines are discussed to inform and contrast our design choices of creating a system that abstracts from body part segmentation. Finally, we discuss the application areas in which touchless gestures have been explored, to motivate and provide greater context as to the importance of touchless gestures as an input technique.

Chapter 3 introduces TraceMatch. We discuss the design and implementation of the system, and the data collection process which was used to inform parameter choice. We then investigate TraceMatch in more depth with a comprehensive user study. We validate the real-time performance by showing that users are effective at selecting input by synchronising with displayed motion using different types of movement, and provide insight into how different ways of performing matching motion (with head, hand, or while holding objects) affect performance. We explore preferences and spontaneous choices of different body movements for input, and ways in which the technique can be extended from binary selection to multi-level input displayed within one orbiting widget.

Chapter 4 introduces the concept of *spontaneous spatial coupling* and discusses the properties which define it. We operationalise this with MatchPoint - a computer vision-based system that builds upon TraceMatch to demonstrate the concept of spontaneous spatial coupling. We then explore the design space using real-world examples, and finally evaluate how adept users are at pointing using different body parts, or when holding an object, with dynamically adjusted control-display gains.

Chapter 5 takes a closer look at the underlying mechanisms of how users synchronise with motion using body movements. Using empirical data, we explore how adept users are at matching against on-screen targets, and investigate how harmonic motion can be used to help users synchronise with targets. We then perform an extensive parameter search on a number of algorithms to determine optimum parameters for different motion trajectories, and to assess the robustness of motion correlation against accidental activation.

Chapter 6 discusses motion correlation as the third input paradigm based on our findings. We then introduce design guidelines for touchless motion correlation interfaces that we refine across the work conducted throughout this thesis. We also discuss future considerations for motion correlation interfaces based on our insights, and limitations of our work.

Finally, **Chapter 7** concludes the thesis.

2

Related Work

In this thesis we explore the use of motion correlation as a novel selection principle for touchless input using body movements. Touchless input forms part of a new computing era - that of the "natural" user interface (NUI). We begin by looking at the NUI, what it is defined by, and position touchless gestures within this emerging paradigm. We then take a deeper look at touchless input using body movements, ranging from hand to whole body input, through the lens of the two dominant conceptual models used. We discuss previous work on motion correlation, from its roots in perceptual control theory, to the processes involved in how we follow motion, and how it has been used in the HCI literature for selection and pairing. Following this, we look at sensing techniques for touchless input, and discuss how these are dominated by detection and/or segmentation of specific body parts and often require complex recognition algorithms. Finally, we look at application areas which touchless gestures have found application.

2.1 Touchless Gestures as "Natural" Input

In 1973 Xerox PARC introduced the Xerox Alto, an experimental workstation with the first operating system to use a graphical user interface (GUI), and which demonstrated the windows, icons, menus and point (WIMP) metaphor [262]. Inspired by Ivan Sutherland's SketchPad [259] and Douglas Englebart's On-Line System (NLS) [68], this represented a huge paradigm shift from the keyboard-based command line interface (CLI) that came before, and ushered in the era of personal computing. The GUI was created to simplify how we interact with a computer and make computing more accessible. In contrast to the CLI metaphor of remembering and recalling specific commands, the GUI relies on recognition of desired commands – fundamentally making all available actions visible and highly discoverable to the user.

The GUI was designed, and optimised, for a specific configuration of input and output devices – a mouse and keyboard combined with a desktop monitor. Today's input/output (I/O) sensing landscape is much more rich and diverse. Due to decades of research, and technological advances, there are now an abundance of devices capable of sensing user input in a variety of different forms. Speech [153], eye gaze [244], multi-touch [106], and body movements [5] can all be utilised for interaction, either in isolation or as complementary input modalities in a multimodal fashion [50, 263, 145]. As computing has become a ubiquitous entity that is embedded in our environment, it has necessitated that output and feedback extend beyond just a desktop display. Outputs may come in many different forms that encompass our many senses beyond

just vision, from auditory [85, 33] to haptic [143], and even olfactory [129]. Even the basic premise of visual output is no longer limited to the desktop or television, as displays come in a variety of form factors, from small smart watches to large public displays. The GUI can not be shoehorned as the de facto standard for this vast landscape of I/O capabilities. The interface must be tailored to the I/O – enter the "natural" user interface (NUI).

The goal of interface design is to minimise cognitive and physical load on the user, enable seamless and effortless interaction, and in effect make the interface invisible. The term "natural", as used in the context of natural user interfaces, refers to the way in which a user experiences a system [286]. These experiences are at different levels of conscious control during interaction depending on a user's familiarity and experience with a device or system. Fitts and Posner developed a prominent model about how we acquire psychomotor skills which consists of three stages: cognitive, associative, and autonomous [75]. In the cognitive stage users must problem solve, as they have no routines or rules available to solve the problem at hand. This requires cognitive effort, and in the context of interaction the user's attention is not on the task that the interface enables, but rather the interface itself. In the associative stage the user will focus on the *how* of the interaction as opposed to the *what* - many tasks require years or decades to master. In the autonomous stage, automated routines are carried out which require minimum cognitive load. NUIs do not offer a silver bullet for interaction design resulting in skilled users from the offset, but the goal is to instead exploit our innate ability to transfer skills and knowledge from our interactions in the real-world to accelerate the transition from problem-solving to skilled interaction.

Our use of gestures using body movements for human-to-human communication have been extensively studied. We spontaneously gesture in as much as up to 80 to 90% of the words we speak [170], and research suggests children (under the age of 2) use gestures independently of spoken language to compensate for their inability to talk, such as when we point at objects that we have do not know the name of [198, 113]. As we, and our verbal communication skills, mature we produce new types of gestures that accompany our speech - iconic and beat gestures [166]. Iconic gestures are used to depict some aspect of an entity (an action, person, or object), whereas beat gestures are hand and arm movements which emphasize or mark the structure of our spoken communications [170]. In the absence of spoken language, our primary communication channel is through the use of sign language, distinctive articulations of the hands that convey semantic meaning and that have evolved from iconic gestures [43]. Gesturing is universal, and the ability to sense body movement has resulted in a plethora of HCI research on how we can use them for input.

2.1.1 Touchless Gesture Input Principles

One of the earliest work on using gestures for interaction, Bolt's seminal "Put-that-there", leveraged the deictic nature of gestures and demonstrated how pointing could be used to contextualise speech for interaction [29]. Speech was used to identify the action ("put"), and pointing with the hand was used to identify the object of interest ("that"), and the spatial location to which it should be moved to ("there"). The combination of speech and gestures has been studied extensively based on its integral part of our use of language [171]. Gestures can also be used in a unimodal fashion, without the need for speech or other modalities. Numerous categorisations have emerged to describe the highly variable physical characteristics of hand gestures [207, 128, 5], inspired by previous work on conversational behaviour [130, 170]. Karam and schraefel categorise hand gestures into five distinct gesture styles as used from an HCI perspective [128]:

- **Deictic/Pointing:** Used to indicate the spatial position or identity of an entity.

- **Manipulative:** A tight relationship exists between the gesturing body part and entity being manipulated (e.g. dragging a volume slider).
- **Semaphoric:** Communicative gestures that utilise a dictionary of static or dynamic gestures which are mapped to specific functions.
- **Gesticulative:** Used in combination with conversational speech, they rely on the analysis of body motion in the context of the user's speech. Aigner et al. further differentiate gesticulative gestures into iconic and pantomimic, but do so on the basis that they can be articulated in a uni-modal fashion, i.e. without the need for speech [5]:
  - **Iconic:** Used to depict information about an entity, such as its size, shape, or motion path.
  - **Pantomimic:** A sequence of multiple gestures whereby an actor imitates the use of an object.
- **Language:** Sign language gestures are grammatically and lexically complete and are used for communication-based interaction, akin to detecting a sequence of words to form a sentence.

With the exception of language-based gestures, these gesture styles are also applicable to other parts of the body. In this thesis, we are not concerned with language-based gestures nor those that require additional modalities. Language-based gestures are built upon complete lexicons and research in this area follows a similar trend to that of conversational agents in that a user communicates with a system. We instead focus on selecting one of a fixed number of options.

The remaining gesture styles operate on the basis of either semantically or spatially mapping the user's input to the output space [73]. Analogous to the CLI metaphor, semaphoric gestures are semantically mapped to a set of corresponding actions, although the gestures themselves may occur spatially in motor space. Manipulative and pointing gestures involve a spatial mapping, and usually a logical abstraction of the user's attention in the form of a cursor. A property shared by the WIMP metaphor which relies heavily on spatial mapping of input. Multiple styles may be employed concurrently when these gesture styles are operationalised in a system, such as when a semantic gesture is used as an activation gesture for a cursor-based interface (e.g. Kinect's wave-to-start [1]). Carter et al. discuss these fundamental selection principles in the context of the dominant selection technique employed in a system [40]: cursor-based for spatially dominated selection, and library-based for semantically dominated. In the following subsections we discuss these principles and their applicability to touchless bodily interaction in more depth.

2.2 Semantic gestures

Systems that rely on semantic gestures for selection require a pre-defined set of gestures that are mapped to specific commands in the interface. Semantic gestures often involve dynamic movements of the body part performing the gesture, ranging from simple side-to-side motion, such as waving with the hand or shaking the head, to more complex motion trajectories such as the tracing of alphanumeric symbols. Static poses of the hand can also be used as semantic input, for example joining the thumb and forefinger to form the "okay" symbol. Semantic gestures have the potential for highly expressive input, by specifying not only the command but its parameters as well [15]. There is no requirement for direct spatial mapping between input and output spaces because gestures are mapped semantically. This means semantic gestures can

[1] Kinect Gestures | Wave to Kinect: `https://support.xbox.com/en-GB/xbox-360/accessories/body-controller` Accessed 23/11/19

be invoked from multiple locations, independent of orientation, and can be used as input to a wide range of devices, from smart-home applications with limited-display devices [84], to large wall-sized public displays [3]. Due to proprioception, there is no requirement for visual feedback and they do not require screen real-estate, thus reducing visual clutter. Semantic gestures can be used in an eyes-free manner with feedback from either auditory [288] or mid-air haptic channels [151]. This is especially useful in the automotive industry, where reducing driver distraction is a primary concern [61, 218]. This also makes them suitable as an accessible means of input for people who have vision impairment [127]. However, semantic gestures have widely discussed usability issues [189], including questions of how gestures are revealed, discovered and learned [15, 279, 3].

2.2.1 Midas Touch

A semantic gesture consists of three distinct stages: (1) *registration* signals the user's intent to interact and the beginning of the gesture, (2) *continuation* contains the information conveyed by the gesture, and (3) *termination* signals the end of the gesture. In touch-based gestures the registration and termination stages may be as trivial as touching the screen and releasing the finger. For mid-air gestures this is not as apparent and there may not be an explicit delimiter that marks the start and end of a gesture – also referred to as the "gesture spotting" problem [297]. The lack of clear delineation leads to the Midas touch problem, named after King Midas in Greek mythology, for whom everything he touched turned into gold. In the context of interaction this refers to false activations whereby the system incorrectly interprets a user's actions as intent to interact with the system [133]. A mid-air gesture recogniser may constantly be looking for input from the user, and any inadvertent movement could be misclassified as an intended gesture towards the system.

Several techniques have been proposed in the literature to reduce the Midas touch effect. Alternative modalities can be used in a complementary fashion, as either a selection mechanism such as in Bolt's "Put-that-there" in which speech was used as the trigger to identify objects or spatial locations [29]. Other modalities can also be used to infer a user's intent to interact based on user engagement [236]. *Activation gestures* are those which one would not expect the user to perform accidentally and are thus reserved to signify the start of an interaction [133, 103, 279]. Upon activation, the sensing device looks for user input until a continuation or a *closure gesture* has been detected [89], or until the user leaves the sensor area. Activation gestures must be carefully designed, as they too are subject to the Midas touch problem [191]. *Active zones* are another approach in which specific input zones are defined with respect to the sensing device. Only when the gesturing body-part is within the pre-defined bounds of the active zone are gestures registered [15, 137]. Conveying the bounds of the active zone to the user may be troublesome, and such an approach removes the flexibility of mid-air gesture systems to provide input from multiple locations. All of the aforementioned solutions may help alleviate the Midas Touch problem, but they do not remove it and there is also the need to reveal these initial registration gestures to users [279].

2.2.2 Gesture Mapping

The ultimate goal of gesture set design is to maximise the ease-of-use and memorability of the mappings between user input and system output from a user perspective, whilst also maximising recognition accuracy from a technological perspective. Our own conceptual mappings of gestures are dependent on a number of factors. Semantic gestures in particular depend on habit, background, and culture [163]. A benign gesture used commonly in one culture may have significantly different meaning in another. The previous example of the "okay" symbol has

significantly different meanings if used in the United Kingdom (okay), Japan (money), France (zero), or Brazil (an obscene gesture). The advantages often touted regarding gestures - their naturalness, intuitiveness, ease-of-learning - are heavily dependent on the gesture mapping between input action and system command. Wobbrock et al. investigated what "natural" interaction looked like in terms of multi-touch input gestures [294]. Their work revealed that despite agreement for physical manipulation tasks, such as moving an object or changing its position, there does not exist a universally agreed upon set of gestures. This has also been found to be the case for mid-air hand gestures [266]. For interaction there are many abstract tasks for which there may be no innate mapping that draws consensus. There are three methods that have been proposed for the design of gestures: pre-designed, user-elicited and user-defined gesture sets.

Pre-designed gestures sets are those created by a designer, or set of designers, according to a design process. Knowledge of the underlying recogniser technology and specific application requirements can be used to create mappings that are easy-to-remember, perform, and guessable [293]. For example, two gestures that are conceptually very different to a user may be very similar to a recogniser. User elicited design involves participatory design [294]. A subset of users generate gestures which are compiled at design-time into an agreed upon gesture set [251]. Designers are still an integral part of this process, as they must ensure that identical gestures are not assigned to the same input action. Morris et al. found that gesture sets created by larger groups of people were preferred to those created by individual designers (in this case the authors of the paper) [178]. Interestingly, they also noted that the HCI researchers' gesture sets contained more complex gestures, from both a physical and conceptual perspective. User-defined gesture sets are created by the users without oversight from a designer. Nacenta et al. demonstrated how user-defined gestures can offer greater customisability and ease-of-use for participants compared with pre-designed gesture sets, resulting in greater memorability [183]. However, this comes at a cost to collaborative awareness and transferability, because user-defined gestures are application and user-specific. In all three methodologies, memorability is still an issue for users, which leads us to the next issue with semantic gestures.

2.2.3 Recall vs Recognition

Once a library of gestures has been established for use, users must discover and learn which gesture corresponds to which system action or command, and at a later date recall these same mappings. The majority of interfaces that utilise semantic gestures rely on recall of the gesture set, violating Nielsen's heuristic of recognition instead of recall [187]. It was this same premise of "see and point versus learn and remember" that helped propel the GUI to prominence by catering to novice users [241]. Kurtenbach et al. describe three similar principles to support learning of gestures: revelation, guidance, and rehearsal [139]. It is the system's responsibility to reveal its commands and how to invoke them. The problem of recall becomes more apparent for transient interactions with public displays, where users may not even be aware of the input modality that is required to trigger an interaction, i.e. they may not be aware the system accepts gesture as input.

Feedback and feedforward techniques can be used to facilitate learning of complex gesture sets for mid-air interactions. Feedforward techniques reveal the available commands to the user and how to perform them prior to the user executing the gesture. Crib sheets are a feedforward technique that can be displayed on-demand to show the available gestures and what actions they correspond to [139]. Kurtenbach et al. demonstrated how these can be combined with contextual animations to help users learn new gestures. However, they require the user to divide their attention between the gesture being performed and the shown example. Feedback can be provided to users either in the continuation phase [14, 80] or after a gesture has been completed [32] and provide information to the user regarding the current recognition state of the system. Feedback

techniques not only help facilitate the learning of gesture sets, but have the added advantage of providing users with visibility of the system status. We refer readers to [55] for an exhaustive description of gesture guiding systems.

Techniques that combine both feedforward and feedback have also been around for decades in the form of pie menus [37] and marking menus [139]. In mouse and touch-based interfaces, marking menus are revealed by waiting during gesture registration (inferred through mouse click or touch down) until the system displays the available gestures [139]. Bau and Mackay extended marking menus to more complex gesture sets with the concept of dynamic guides, which continuously update feedforward and feedback during the gesture continuation phase [14]. ShadowGuides and Aperge extended this technique to the more complex multi-touch and chord gesture problems [80, 88]. Delamare et al. extended the dynamic guides technique to mid-air gestures in 3D using a mouse for gesture registration [56]. The aforementioned techniques are dependent on the wait (dwell) during the gesture registration phase, and thus are not always applicable to mid-air gestures due to the difficulty in establishing gesture registration as discussed earlier.

In the context of gesture recall, too much guidance when learning a gesture can hinder later recall. Anderson and Bischof compared four types of guide for 2D gestures and measured the associated learning of gesture sets using a transfer and retention paradigm, common to the motor learning literature [7]. They tested a crib sheet, a static guide, a dynamic guide based on OctoPocus [14], and a novel adaptive guide. Retention tests to assess consolidation of motor skills were performed without any guide feedback, in addition to transfer tests performed with the non-dominant hand to see how well the learnt gestures were generalisable. Both sets of tests were conducted at 15 minutes and 24 hours after the initial training phase. They found that guides with higher guidance given during the training phase showed high performance benefits whilst the user used the guide, but at the expense of lower learning and retention effects. This can be explained by the *guidance hypothesis*, which states that a user may become overly reliant on the guidance given during the training phase which in turn hinders learning of the underlying process [233].

There has been limited work in revealing and learning mid-air gestures. Sodhi et al. projected visualizations on to the hands of users as guidance in *LightGuide* [247]. They found that participants were more accurate compared with video guidance using a 3D model of the hand. However, LightGuide requires users to divide visual attention between the gesturing hand and display, and the practicality of projecting visualisations on users is questionable. In *StrikeAPose*, Walter et al. used a teapot gesture as an activation gesture and investigated revelation techniques for public displays [279]. Three techniques were used: spatial division akin to showing a crib sheet, temporal division whereby content was disrupted to show the registration gesture, and integration in which the registration gesture was integrated into the content being shown on the public display. They found spatial division outperformed the others, however this focussed on the revelation of an initial activation gesture and not a whole gesture set. In contrast, *Gestu-Wan* explored the discoverability of a whole gesture set for public displays [225]. Gestures were manually subdivided into intermediate postures and structured hierarchically in a tree. Based on a user's given state in a gesture (i.e. posture) the system displays available commands and the respective posture required to transition. Gupta et al. investigated how haptics could be incorporated into the learning process for finger tap gestures and command shortcuts [98]. Combined with visual stimuli, they demonstrated that haptic learning is comparable with visual learning after thirty minutes which shows promise for learning gesture sets for systems with limited display capabilities.

2.3 Spatial gestures

Spatial gestures map the user's input to the device's output in the spatial domain, and include pointing and manipulative gestures. Pointing is a fundamental interaction principle that draws on human spatial abilities and skills. The principle is at the core of graphical user interfaces on our desktops, tablets and smartphones, but naturally extends to touchless interaction with displays that are remote, shared, large, public, ambient or used in settings that prohibit touch for hygienic reasons [29, 67, 27, 276, 265, 191]. The use of pointing in interfaces is often mediated using a logical abstraction of the user's attention, most commonly in the form of a cursor (a.k.a. a pointer). A static configuration of icons and or menus represents actions available to a user, which are selected using a point-and-select metaphor. This metaphor is at the heart of the WIMP paradigm that underpins the GUI. However, many other aspects of the GUI are optimised for the pairing of mouse and keyboard, and do not extend into the realm of touchless interaction. When we point with our hands for example, we do so with coarse accuracy and low precision due to inherent human limitations, such as hand jitter [182]. The mouse on the other hand has been refined as the ultimate pointing device with decades of research. Spatial gestures also include manipulative gestures which control an entity by applying a tight coupling between the actual movement of the gesturing body-part and the entity being manipulated [207]. These can be achieved using one hand to grip and manipulate the object with the thumb and forefinger [190, 237], or they may build upon their touch-based counterparts by using bimanual input [105]. Researchers have also posited unique metaphors for 3D manipulation, such as virtual objects that have been skewered with a bimanual handle bar [248].

Related work generally assumes use of the hands for pointing (e.g., [276, 242, 12, 122, 40]), and the non-dominant hand has been shown to have comparative performance relative to the dominant hand for pointing tasks [121]. Work in other areas has shown that humans are equally natural at pointing with other parts of their body (literally, from head [159, 22, 132] to toe [270]). Although pointing with the hands often outperforms pointing with other body parts, the head has also been shown to be an accurate and efficient way of providing input by pointing [24, 148]. There are compelling advantages of using alternate body parts for pointing and interaction, such as accessibility devices [117] or when the hands are otherwise busy with other tasks. Fitzmaurice et al. demonstrated how physical tokens can be tightly coupled with virtual objects for manipulation or expressing actions [77], based on prior work employing real-world props as handles for spatial controls [104]. There has also been prior work that investigates how physical objects can be coupled with virtual entities in an ad-hoc manner [92, 274, 28].

2.3.1 Gesture Mapping

Spatial gestures suffer from similar usability issues to semantic gestures, including the Midas touch problem (see Section 2.2.1), and how to map the user's input to the system's output. Conventional user interfaces support pointing by tightly coupling a pointing device with a display surface, or by integrating pointing and display. Mapping the user's movement to the display is a challenge for touchless pointing, as the user's input space is not straightforward to discover. The user's movements can be mapped to the display in an absolute manner, whereby movement of the user's input results in a fixed translation of the on-screen cursor [276]. Ray-casting methods can also be used to map input to output spaces. An invisible ray is cast from the input device or body part and the point at which it intersects the display dictates the position of the cursor. Ray-casting is much more complex because it requires the sensor to have absolute knowledge about the user's position and the display in 3D space. There are also different methods for casting the ray itself.

For hand-based interaction, the ray can be cast from one or multiple fingers [276, 193, 162].

Bimanual ray-casting can also be used to leverage inter-hand coordination which can mimic multi-touch gestures [12], or be used for selecting objects in a 3D space based on their intersection [296]. Two points are required for directional ray-casting, one of which is commonly located at the end of the index fingertip, however researchers have explored alternate origins for the ray including the eye, head, centre of the body, or forearm [120, 12, 11, 167]. For head-based pointing, the ray is usually cast from the centre of the head, however it could also be cast from the position of the eyes, and due to limitation in eye tracking technology the ray cast from the head is sometimes used as a proxy for the gaze position [24, 202].

In general, we point in coarse directions and an alternative to absolute positioning techniques are relative pointing, most commonly found in the mouse and desktop scenario. Movement of the on-screen cursor is dictated by the velocity or acceleration of user input [276, 280]. However, relative pointing requires a clutch movement in order to disengage the pointer and reposition. This can be complicated in touchless bodily interactions, and alternative modalities, active zones, or semantic gestures can be used to engage the clutching mechanism in a similar fashion to activation gestures. Relative pointing can be advantageous because fast movements are translated to coarse-grained displacements on the screen, whereas slow and methodical movements translate to the fine-grained movements required to accurately select smaller targets. Both absolute and relative pointing techniques can be combined to leverage the advantages of both. Vogel and Balakrishnan suggest a hybrid approach, which allows users to switch modes between either absolute or relative pointing depending on whether coarse- or fine-grained selection is required [276]. Prolonged usage and ill-conceived mapping combined with hand-based pointing can exacerbate known fatigue issues [97, 101].

2.3.2 Gorilla Arm

One of the unfortunate side effects of mid-air pointing with the hand is muscle fatigue – commonly referred to as the "gorilla arm" effect [97, 101]. Interfaces that rely upon pointing and manipulation tasks are likely to require prolonged periods of interaction in which the arm is in a raised position. Intense muscle tension over prolonged periods from a result of maintaining a static pose, or the effort to move the hand through a trajectory, can harden arteries and restrict blood-flow. Hincapie-Ramos et al. propose the *consumed endurance* metric to quantify the fatigue experienced during mid-air interactions [101]. They suggest that relative movements should be used to enable users to perform gestures in areas of least effort. Microsoft released a set of human interface guidelines for the Kinect which detailed numerous ways to reduce the gorilla arm effect [175]. These include enabling users to freely switch between hands, avoiding long interactions that require the user to keep their hands raised, and avoidance of fine-grained targeting in the vertical axis. Jude *et al.* showed that accuracy can be maintained when pointing on small displays if the user is able define their input space during a calibration phase [122]. Other work suggests allowing interaction from a rested position, e.g. a table, to limit fatigue [97]. Many of the constraints that sensing devices place on users in order to detect gestures or to reduce the Midas Touch problem can exacerbate gorilla arm fatigue issues, e.g. specific postures or hand positions relative to the body.

2.4 Motion Correlation

So far we have discussed spatial and semantic gestures which map user input to system output in the spatial domain or through semantic meaning. Building upon our natural ability to mimic external motion, *motion correlation* represents a third input principle for selection, defined by three properties [268]:

1. Motions displayed on an interface represent actions that are selectable by the user;
2. A user mimics the desired action's movement in order to signify their intent to interact;
3. Selection is determined by the coincidence between the user and the object's movements

The principle behind such techniques has been referred to by several names including rhythmic path mimicry [40], periodical motion coincidence [73], and feedback-based active selection [287], but is essentially the same across them: each selectable target on the interface presents a distinct movement and the user selects the desired target by matching (e.g. *PathSync* [40]) or counteracting (e.g. *Eggheads* [287]) the corresponding movement.

Motion is the change in position of an object over time. When determining if two signals match each other we can consider their spatial properties (e.g. position, trajectory), temporal properties (e.g. frequency, velocity), or a combination of both – their spatiotemporal properties. In this thesis, we focus on spatiotemporal motion correlation, which utilises information in both the spatial and temporal domains for inferring user intent. More specifically we are concerned with the spatiotemporal coincidence between both input and output signals. A match is detected based on the principle that spatial and temporal properties of both input and output are aligned according to some similarity measure. We use the term correlation, not in the statistical sense, but in the relationship between input and output. It is important that output motion is temporally synchronised with user motion, in contrast to spatial gestures in which movement at the output is in response to the user.

2.4.1 Temporal Coincidence and Spatiotemporal Properties of Gestures

When considering only the temporal properties we discard any information about the spatial aspects of the motion and can consider discrete inputs such as pushing a button. Previous work has looked at the temporal coincidence between input and output signals without considering the spatial aspect. *Switch scanning* displays items for selection in a grid, with an indicator that constantly changes which item is highlighted [246]. In order to select an item, the user must time their input action (e.g. pressing a button) to coincide with their preferred item being highlighted. Switch scanning is a standard accessibility technique for when accurate and aiming is problematic, e.g. due to a physical disability. This same technique was adopted in *Rhythmic Menus*, in which items are periodically highlighted in a drop-down menu upon depression of the mouse button, and the highlighted item is selected upon release of the mouse button [164]. *Resonant Bits* explores temporal correlation in terms of resonance and how a system's continuous feedback can guide the user's rhythmic input [21]. Selection occurs by rocking the phone in time to displayed pendulums. When the phone is rocked at the same frequency as one of the displayed pendulums, the pendulum's amplitude is increased until selection. All of these techniques rely on synchronising with the interface in the temporal domain. Of note are other techniques which use rhythmic input but do not rely on temporal coincidence between input and output, e.g. [87].

Malacria et al. recognised the potential for using the spatiotemporal properties of a gesture for interaction with their *Cyclostar* techniques, two novel techniques which leverage spatiotemporal properties of the gesture [158]. *Cyclopan* involves oscillatory motion of a gesture in which the orientation, amplitude, frequency of the oscillatory motion are mapped to the pan direction, distance, and gain respectively. In *CycloZoom+*, they utilise the spatiotemporal properties of a circular gesture. The direction, frequency, radius, and centre of the circular gesture is mapped to the zooming direction, speed, accuracy, and focus of the zooming interaction. Unlike the Cyclostar techniques in which individual spatiotemporal properties are mapped to specific actions, motion correlation techniques rely on the coincidence of all spatiotemporal properties of both input and output signal, enabling for greater flexibility in mappings at the cost of reduced resolution.

2.4.2 Motion Perception and Motor Behaviour

We have an innate ability to perceive and synchronise with externally generated rhythm and motion, which has been studied extensively in the psychological and human movement science literature. Let us consider the process of mimicking the movement of a visually moving target with our hand. In the first instance we lock on to the target with our eyes by performing a *saccadic* eye movement – a rapid eye movement from one target to another. This saccade precedes the pointing movement of the hand [205], and gaze is maintained on the target until the pointing is completed [184]. Once locked onto the moving target, our eyes perform a specific type of eye movement known as *smooth pursuits* – a type of eye movement that is very difficult to replicate without external stimuli to follow [136]. Importantly, selective attention is required to maintain our gaze on the target in the presence of conflicting stimuli [13].

The coordination of one's movements with an external rhythmic event is known as sensorimotor synchronisation [213]. This is a phenomenon that is most apparent when we listen or dance to music and synchronise our body movements with the rhythm. Although we are more proficient at synchronising discrete body movements (e.g. foot tapping) with audio cues (e.g. a metronome) , we can also synchronise to visual stimuli that exhibits rhythm, such as a flashing light [215]. In the sensorimotor literature, the most commonly studied type of synchronisation is discrete body movement, such as finger tapping, however others have studied continuous motion such as drawing continuous circles in time to a metronome [299]. Different underlying processes have been implicated in the control of rhythmic movements depending on whether a discrete or continuous movement is used. Discrete movements use significant points in time as a reference with which to synchronise, known as *event-based* timing. In contrast, continuous movements such as drawing circles have no salient visual or kinaesthetic event which can be used as a reference point in which to synchronise against. For synchronisation using continuous movements it is believed that we use an *emergent timing* process whereby movement starts out of synchrony, but due to proprioception the error is reduced by changing the non-temporal properties of the movement, e.g. stiffness of the arm which changes the size of the drawn circle which inherently changes the timing [299].

A flashing light demonstrates only temporal variation of the visual stimulus. When synchronising with visual stimuli, it has been reported that we are better at coordinating when the target is presented in a continuous rather than discrete manner, and also when spatial information is provided in addition to temporal [36, 108]. When synchronising with visual targets that exhibit spatiotemporal properties (i.e. varying in both time and space domains), we are much more adept at following targets in-phase (i.e. body part and target move in the same direction) than we are at anti-phase tracking (i.e. body part and target move in opposite directions) [222, 42]. This is to such an extent, that when asked to track a target anti-phase with increasing frequency, participants will abruptly shift to in-phase tracking at a certain frequency [197, 292]. Interesting insights that can be gained also include the phenomena of *anchoring* during rhythmic movements. Originally formulated by Beek when investigating ball juggling, *anchor points* are thought to be critical points during cyclical movements that are used to time and stabilise movement cycles [17]. Both gaze and musculoskeletal factors are thought to influence the locations and properties of anchor points. When synchronising our body movements with low frequency in-phase oscillations we follow the target with smooth pursuit eye movements, however at higher frequencies users will fixate on an end point – a form of visuo-motor anchoring [222]. The endpoint at which the user gazes will exhibit reduced spatial variability in the movements. Musculoskeletal factors, such as wrist pose, contribute to the anchoring phenomenon by affecting not only spatial variability, but also the timing of the synchronisation [221].

Our ability to mimic and synchronise with external motion is grounded in our innate sensorimo-

tor capabilities, with evidence of mirror neurons and motor resonance behaviours, where sensory and motor neuron regions fire simultaneously during imitation of movements [219, 141]. Our movements themselves are naturally cyclic and harmonic, and repeated cyclic movements are mechanically efficient [94]. It is therefore compelling to use the principles of rhythmic motion and synchronicity as a means of providing input. We now look to how this innate capability has been used as input in user interface design using a wide range of input devices.

2.4.3 Perceptual Control Theory

The first researchers to utilise the principle of motion correlation for interaction were Williamson and Murray-Smith. In pointing without a pointer, they showed that selection of one object among many could be achieved without using a logical abstraction of the user's attention, i.e. the cursor/pointer [289]. Their motivation for selection without a pointer was driven by interaction where an input device may not lend itself for pointing (e.g. accelerometers) or where an output cannot provide the necessary visual feedback. In the traditional point-and-select method, the user explicitly defines their attention with the position of the cursor, and confirms their intent to select an object with a selection mechanism, e.g. mouse click. Instead, Williamson and Murray-Smith developed a technique which estimates user intention based on perceptual control theory [204]. This was achieved by introducing "disturbances" to variables under the user's control, and based on observations of the user's corrections in response to the disturbances it is possible to infer their intention. This was operationalised in applications in which smooth pseudo-random movement was injected into objects for selection. To select an object, the user stabilises the movement of the desired object though movement of the input device, which in turn applies the same control action to all other objects simultaneously. Although not entirely like the rest of the work presented here, this demonstrated the fundamental principle behind motion correlation as a selection technique – the system looks for correlation between the user's input and the system's output, in this case the disturbances. The technique demonstrates how a user can select one object from many without the need for a cursor, and thus without needing to split their attention between a cursor and a target.

2.4.4 Selection in Graphical User Interfaces

Fekete et al. built upon the principle of using spatio-temporal coincidence of input and output signals for selection in *motion-pointing* [73]. In contrast to Williamson and Murray-Smith's approach of selection through stabilisation of pseudo-random disturbances, Fekete et al. introduced the concept of imitation of a rhythmic movement as a selection principle. They operationalise this selection principle using cyclic motions on elliptical trajectories and contextualise it for selection by a mouse in a graphical user interface. In the same vein as "Pointing without a Pointer" motion-pointing provides more visibility of the available commands at the cost of increased visual load, whilst relying on the user's proprioceptive feedback for synchronisation and error correction instead of visual feedback from a cursor. Gestures are expressive and can be used to specify both a command and its parameters (e.g. the object to which the command should be executed on). Motion-pointing demonstrated how different properties of the same gesture can be used to differentiate different commands, by varying the phase, speed, or eccentricity, of a moving target along an ellipse. They discovered distinct phases during the matching process. The initial phase (1 - 1.5 seconds) involves the user observing and initiating the synchronisation, this was followed by the coupling phase (1-2 seconds) in which the user is in synchrony with the displayed motion, and finally a third phase in which synchronisation was lost and in some cases recovered, but not always. As motion matching was not 100% accurate they include an additional discrimination phase using a dynamic pie menu based on the four closest matching

motions. This highlights a fundamental challenge of motion-matching – designing motions that are distinct enough for detection.

2.4.5 Smooth Pursuit Eye Movements

Eye tracking is a compelling input modality because our gaze is naturally drawn to objects of interest that we may wish to interact with. Previous work explored using gaze as spatial or semantic input in a similar way to traditional input techniques. In the spatial domain, gaze has been investigated as a replacement for the mouse cursor [114, 78], and gaze gestures have been proposed for semantic mapping [64]. However, detecting and utilising gaze for interaction poses a number of significant challenges. For spatial mapping, a user's gaze must be calibrated to the display coordinates which often requires an explicit calibration stage. Accurately detecting eye movements is a non-trivial task, confounded by external factors such as lighting or even make-up. Confirming a selected object is often performed using dwell, however this requires users to fixate their gaze on a target longer than necessary [211]. There is also the same Midas touch issue that pervades input techniques based on body input, with limited means of disengaging gaze as input.

Vidal et al. leveraged the smooth pursuit eye movement for interaction in their seminal *Pursuits* technique [273]. By correlating the eye movements with on-screen targets for selection users are able to spontaneously use eye trackers for interaction. Unlike previous work which requires users to actively synchronise with the displayed motions using mouse movements, Pursuits leverages our innate ability to lock-on and follow moving targets with our eyes. Vidal et al. go beyond variations of elliptical trajectories to demonstrate how motions can be integrated into more natural trajectories based on naïve physics, e.g. following the motion of fish swimming or flies flying. These can not only be used for explicit interaction such as purposefully following an object for selection, but also for implicit interaction such as revealing which object a user is looking at. Pursuits presents a major advantage over previous gaze techniques which require accurate mapping between input and output space.

The scale-independence of the Pursuits approach was highlighted in Esteves et al.'s *Orbits* work which used the Pursuits selection technique and applied it to smart-watches. Like Fekete et al.'s oscillatory elliptical movements, Orbits consist of targets that orbit in a circular trajectory that act as explicit controls for selection. Esteves et al. investigated how robust the technique was against false positives and how user's coped with selecting one target amongst many in dense arrangements. In a first study using a remote eye tracker and laptop display they found that an angular frequency of $120^o/s$ combined with the largest trajectory size (2.36^o of visual angle) produced the highest recognition rate compared with angular frequencies of $60^o/s$ and $240^o/s$. They also found a significant effect of size on recognition rate, however their smallest orbit (0.55^o of visual angle) is approaching the manufacturer's reported gaze estimation error of 0.4^o. In a follow-up study using a wearable eye tracker and smart watch, they found that the same angular frequency of $120^o/s$ achieved the highest recognition rates compared with $180^o/s$ and $240^o/s$. They also investigated how many circular targets could be displayed simultaneously, and discovered that users were able to accurately select a target from up to eight targets. Beyond eight targets recognition rate dropped off due to inadvertent selections of incorrect targets.

These works have inspired a number of follow-up papers in the eye tracking community. AmbiGaze demonstrated that the technique could be used with motion generated by mechanical means in a distributed manner [271], text pursuits showed the use in public displays [131], and Dhuliawala et al. demonstrated that electrooculography-based (EOG) glasses can be used with the Pursuits technique, in spite of the fact EOG is unsuitable for gaze pointing [59].

2.4.6 Touchlesss Gestures

Motion correlation with body movements presents unique challenges compared with smooth-pursuit eye movements. We must consider how well users can synchronise their body movements with external stimuli, including any errors introduced as a result of poor sensorimotor synchronisation. To date, two pieces of work have investigated motion correlation with touchless gestures using hand movements. Carter et al. developed *PathSync* independently of, but at the same time as Freeman et al. developed *Do That, There*. Both interaction techniques are based on the premise of motion correlation using mid-air hand gestures, however they are aimed at different and unique problems.

Do That, There is primarily aimed at *addressing* in-air gesture systems using a multimodal feedback approach. Freeman et al. discuss the concept of using motion correlation for users to direct input at specific input devices. The "Do That" element of the interaction technique uses light animations to convey the required motion and tempo of the rhythm in which users must synchronise with. They investigated the effect of including audio and haptic feedback of when the user successfully matched their movements with the visual motion, and also the effect of different hand movements, including:- side-to-side, up-and-down, forwards-and-backwards, and both clockwise and anti-clockwise circular movements. The "There" element provided feedback to users that they were in the gesture sensor's sweet spot and could provide gestural input to the system. For the "Do That" element they investigated the five different gesture types using four different time intervals (the time to complete one cycle): 500, 700, 900, and 1100 ms. The paper concludes with four distinct guidelines. Pertinent to this thesis, were the suggestions to use line-based movements (i.e. vertical/horizontal) when possible, as opposed to 2D shapes, and that gesture intervals should be at least 700 ms for line-based movements, and 900 ms for circular gestures. In addition, they suggest that interactive light should be used to show where a user should gesture (i.e. display the motion visually), and that uses should be provided feedback about their movements.

It is important to note how time interval is defined for these gestures. In the case of the circle, it is defined as being the time required to complete one full oscillation. In the case of line-based movements, it is the time taken until stopping the movement (i.e. the interval for the hand to return to its original position is twice this). Freeman et al. found that the higher time intervals (i.e. slower gestures) resulted in higher success rates for all movement types, with side-to-side and up-and-down achieving the highest success rates. At the 1100ms interval side-to-side and up-and-down achieved 100% success rates, closely followed by the clockwise (98%) and anti-clockwise (97%) circular movements. The circular movements were also found to be slower for matching than other movements, with side-to-side found to be significantly quicker than all other movements. The intervals selected for this experiment are much lower than previous works [273, 69]. In "Do That, There" interval times were selected based on previously reported rate limits of sensorimotor synchronisation by Repp [214] and van der Wel et al. [264]. However, it is worth noting that [214] is concerned with finger tapping to an auditory or visual event, and [264] involves the movement of a dowel from side-to-side in time with a metronome.

At the same time, Carter et al. implemented *PathSync* based on the motion correlation concept for mid-air hand gestures enabling multi-user touchless and cursor-less interaction with public screens [40]. PathSync investigated the use of motion correlation as a discrete selection technique, demonstrating the discoverability, intuitiveness and multi-user capacity of motion-matching for hand-based gestures. They examined how adept users are at matching different shapes using closed-loop shapes that vary in two dimensions (i.e. x and y axes), and found that the square and diamond shapes has higher success rates than the circle and rounded square shapes. The authors suggest that this could be due to the corners of the square acting as salient points for synchronisation, but no in-depth investigation was conducted. In a second follow-up study, forty participants evaluated PathSync against the *Press-to-select (PtS)* method. The PtS

method is commonly used for commercial gestural systems (e.g. the Xbox One Kinect gesture interface) and involves mapping the user's hand to an on-screen cursor (1:1). Selection is achieved by extending the hand towards the screen in a "pressing" motion. The study investigated selection of one object amongst many, and found no significant differences between error rate or time-to-select. Participants' subjective opinions were also evenly split between the techniques regarding ease-of-use, perceived speed, and frustration. In a third study, a simple quiz game was deployed on two public displays over a 4-week period. Over this period 1065 sessions were recorded of users interacting with the system, demonstrating its discoverability, intuitiveness and robustness.

Since the start of this thesis, several other approaches have been developed, some of which inspired by the work featured here. Touchless gestures are not limited to large movements of the arms, and a more discreet form of input are small movements of the hand known as micro-gestures. In *Rhythmic Micro-gestures* Freeman et al. investigated how well users could perform small micro-movements using the concept of motion correlation [82]. Using four micro-gestures that could be robustly sensed by a Leap motion they found that rhythmic micro-gestures could be performed with simple non-visual feedback by using audio cues, demonstrating that the concept of motion correlation can be applied to eyes-free interaction mobile scenarios. SynchroWatch demonstrates temporal-based motion correlation due to sensing constraints using a thumb-worn ring equipped with magnet to synchronise with on-screen targets by tapping their thumb against the index finger [216]. The principle of spatiotemporal motion correlation could be adapted using spatiotemporal finger-based ring gestures, e.g. [301]. Whereas, SynchroWatch used a magnet located in a ring to sense finger input from a smartwatch, *WaveTrace* used the in-built inertial measurement unit (IMU) for detection of larger macro-movements of the arm [272]. By comparing the pitch, yaw, and roll of the user's wrist movements to on-screen circular targets, WaveTrace demonstrated that the relative positioning of the IMU is sufficient for the matching process. The same concept of using the IMU was also applied to a head-mounted displays in order to detect head movements in an augmented reality application [70].

2.4.7 Beyond Selection

So far, we have discussed the use of motion correlation as a tool for selection, yet correlation of two motions can be used for either implicit interaction, such as calibration, or explicit interaction beyond selection using the parametric properties of motion. Previous work has demonstrated that matching movements in the spatiotemporal domain yields additional information that can be leveraged for implicit calibration of a user to an output device. For gaze interaction, one of the unique properties of the Pursuits approach is that matching of input and output movements is dependent upon relative movement. This removes one of the biggest barriers to interaction with gaze – the requirement for calibration. In *Pursuit Calibration*, Pfeuffer et al. demonstrated that not only can the Pursuits technique be used without calibration, but it can also be used for the calibration process itself [199]. When a match is detected one assumes that the user was looking at the motion generated, therefore any matches detected can be used to map the user's gaze to the display's coordinate space. Once calibration has taken place, users can interact using gaze pointing. Three applications demonstrated that calibration could be performed either explicitly, combined with interaction, or performed implicitly without the user's knowledge.

So far we have discussed synchronisation between the spatiotemporal properties of the output display and user's input. Previous works have also investigated how the synchronisation of motion between multiple input devices, or between a user and an input device. By utilising the parametric properties of movement, correlating two motions can be used for authentication, device pairing, and extraction of user-device relationships.

Pairing and Authentication

Many pairing techniques are based on one device presenting a secret that the user has to input on the other. Patel et al. presented a variant where the user's phone prompts a terminal to display a gesture, which the user has to reproduce with their phone in hand to authenticate it for pairing [196]. *Smart-Its Friends* and *Shake Well Before Use* both utilise the motion correspondence between two devices that are shaken together for device-pairing [107, 168]. The latter extends the principle by utilising the properties of the motion whilst being shaken for generating authenticated secret cryptographic keys. They demonstrate that the activity of shaking is vigorous enough to not cause false positives, whilst also variable enough to cause different entropy patterns required for generating secure cryptographic keys. Hinckley demonstrated that tablets can be paired by synchronising their accelerometer data over a wireless network to detect when they are bumped together [102]. When they are bumped together, the devices exhibit similar sensor patterns, but with spikes in the opposite direction. Once paired, tablets can then be tiled together to create a temporary larger display. More recently, *Tap-to-Pair* demonstrated that tapping on a device's wireless antenna drops the signal strength enough to be used as a pairing mechanism [302]. A device to be paired advertises a distinct blinking pattern, and a secondary device becomes paired upon detection of a user mimicking this blinking pattern by tapping on the antenna, demonstrating the capability of using one-dimensional data for the matching process.

User Identification

Identifying whom is interacting with a display can be problematic in multi-user interactive environments. Researchers have shown that identification of a user or specific body part can be inferred by correlating a user's input action from the system's perspective (e.g. touch event) with a user's input action recorded from a different perspective (e.g. an internal sensor or wearable device). Schmidt et al. demonstrated this concept for cross-device interaction between mobile phones and interactive surfaces, correlating the phone's on-board IMU with a touch event on a surface to facilitate data transfer, user-identified input and authentication on shared surfaces [230, 232]. Other work has used the fusion of internal device sensors and visual features to identify which devices belong to which users. *ShakeID* used the correlation between a phone's accelerometer sensor data and the user's skeletal model acquired from a depth camera in order to identify which phone belonged to which user [223]. *CrossMotion* demonstrated a more generic sensing approach by tracking movement on a per-pixel basis using dense optical flow and correlating the movement seen in the camera with the device's IMU [291]. In both cases, this demonstrated the additional contextual information that can be gathered by correlating user movements with system inputs. Webb et al. go beyond identifying users to identifying which body parts are used to interact with an interactive surface based on Guiard-abiding bimanual touch [283]. By using wearables, such as a wrist-band or a ring, the system identifies which hand is interacting with the display, providing the contextual information to create interactions that leverage the complementary nature of hands when performing bimanual interaction [93]. In all of these cases, the correlation between multiple perspectives of the user's inputs provides rich information to augment the interaction and provide additional capabilities.

2.5 Touchless Sensing Techniques

Whether detecting intricate hand movements or whole body poses, sensing human body movements is a non-trivial task. The sensing techniques used are largely dependent on the type of

gesture to be sensed and consist mainly of two stages: detection, and tracking (spatial) or recognition (semantic). In the first stage, the body part(s) are detected and segmented to isolate the gesture-relevant data. Then, for pointing gestures the desired body part must be tracked over multiple frames to generate a time series of the body part in question which can be spatially mapped to the on-screen control. If the detection stage of the algorithm is robust and quick enough, then it may be used on a frame-by-frame basis, however it is quite common to use a more sophisticated detection algorithm followed by a lightweight tracking algorithm that utilises a priori information. On the other hand, detecting semantic gestures involves extracting information about which gesture the user is performing based on the characteristics of the detected body part(s) (e.g. pose, trajectory), which must be subsequently classified based on the pre-defined mappings between gestures and commands.

In order to sense the users body movements in the detection phase there are two main approaches: *contact-based* and *remote sensing*. Contact-based sensing requires a physical interface between the user and sensing device, and include technologies which use sensors attached to the body, such as gloves, body-suits, or a smart watch. Contact-based approaches are often considered more accurate but can be uncomfortable for prolonged periods and may necessitate installation and coupling with external devices which can reduce spontaneity of interactions. In contrast, remote sensing techniques detect and analyse body movement with no physical intermediary between the user and input device. Camera-based technologies using visible or infrared light are amongst the most prolific of these technologies and use computer vision algorithms to detect, segment and recognise user movements. Remote sensing approaches allow for more spontaneous interactions, but can suffer from the occlusion problem, whereby the gesturing body-part may be hidden from view.

2.5.1 Contact-based Techniques

Some of the initial pioneering work on touchless interaction using the hands was realised using instrumented gloves. As early as 1977, Sandin, DeFanti and Sayre created a wired data glove that utilised photo cells to detect and measure finger flexion [256]. In 1987, Zimmerman et al. developed the DataGlove, which at the time far exceeded camera-based technology and provided real-time monitoring of the hand with six degrees of freedom [306]. Commercialisation of the DataGlove led to numerous research institutions acquiring the technology and pushing the field of gesture recognition forward. Baudel and Beaudouin-Lafon's seminal Charade system used the DataGlove to combine spatial and semantic gestures, achieving recognition rates of up to 98% for trained users (70-80% for untrained users) [15]. Whereas contact-based gloves offer the most complete representation of the human hand, they are expensive and not suited for casual interactions. They require complex calibration and setup procedures and are limited to sensing the hand movements of the user wearing the gloves.

Saponas et al. demonstrated that electromyography (EMG) can be used to classify different gestures by sensing muscular activity in the forearm [228]. Similarly, Dementyev and Paradiso used force sensitive resistors to detect subtle tendon movements in the wrist, also highlighting that additional information from an accelerometer improved classification results [57]. Inertial measurement units consisting of accelerometers, gyroscopes and magnetometers can be found in a number of commercially available devices. Researchers have investigated how these in-built sensors can be used for gesture detection in devices including the Wii remote [6, 149, 229] and smartwatch [203, 23, 285, 258]. These sensors are also used to detect head movements in head-mounted displays for both augmented and virtual reality environments, and eyewear devices such as Google Glass [176, 300, 47, 71, 116].

2.5.2 Remote Sensing Techniques

Camera-based technologies sense user movement at a distance using computer vision techniques. A single device has the capability to sense multiple users and is not just limited to sensing a specific body part. Krueger's VideoPlace was one of the first works to use real-time image processing to create a silhouette of the user, demonstrating the potential of camera-based approaches for gestural interaction [138]. There are two broad categories of approaches in the computer vision field for detecting the user for gesture recognition: model-based and appearance-based [86].

Model-Based

In the model-based approach the aim is to recover the pose of the user, or of a specific body part such as the hand, based on knowledge of it's structure. This effectively bridges the gap between computer vision approaches and glove-based approaches by recovering the full kinematic structure of the user's hand or whole body. The introduction of depth sensors has advanced the development of 3D-model based approaches by leveraging the additional depth data. Both the Microsoft Kinect and the Leap Motion recover 3D-models of the user's body and hands respectively, and have resulted in popular commercial offerings as consumer devices. Developments in the 3D-model based approaches can also be attributed to developments in the field of machine learning and artificial intelligence. Shotton et al.'s machine learning based approach was one of the first to accurately extract skeletal representation of users using the Kinect, trained using 900,000 images [243].

Despite the advantages of the extra depth dimension, it is relatively uncommon to find depth sensors in the wild. In contrast, 2D cameras are inexpensive and abundant in commercial devices such as mobile phones and laptops. Machine learning has evolved, with deep learning paving the way for recovery of skeletal information from web cameras [284, 245, 39, 38]. However, machine learning depends on high quality training data for the models, even more so in the case of deep learning where very large training sets are required. The size of the training data is also important for the generalisability of the models, so that they can detect or recognise a wide range of objects under varying conditions. Another downside at the current state-of-the-art are that these techniques are computationally expensive and require high-end graphics processing units to achieve real-time performance.

Appearance-Based

Appearance-based approaches represent users in terms of colour, shape, or the motion of their movements without an explicit underlying model. Gestures are inferred through the analysis of these features as they appear in images without an intermediate modelling stage of the user. Colour-based approaches look for colours in the image that match the user's skin colour, and assume that these correspond to the areas of interest that will be used for gesture recognition [261]. However, skin colour varies across human races and external factors such as illumination changes or shadows are problematic. Some vision-based techniques use fiducial markers to help in the detection, such as QR codes or reflective IR beacons [192]. An alternative is to look for the shape of the body part to be matched in the scene [20]. In the case of the hands this avoids issues with illumination and skin colour, however there may be issues detecting the correct contour with respect to background objects and issues with occlusion and view points.

Original work using motion assumed only the user's hand occurred in the scene of the camera [83]. For gestures involving spatial motion, such as pointing or tracing of alphanumeric symbols, it is common to segment the body part and then track it in subsequent frames. If the detection algorithm can be run on a per-frame basis then it can be used for all frames, but if this is

not possible then a tracking algorithm is used to track the body part from frame to frame [210]. Tracking algorithms can be split into four main categories: template-based such as discriminative correlation filters [154], optimal estimation approaches such as the Kalman filter [126], Monte Carlo based methods such as the particle filter [112], or mode-seeking algorithms such as CamShift [298]. These approaches are highly dependent on the quality of the initial segmentation of the body part. Alternatively, an approach that has been adopted for action recognition is to utilise the motion trajectories of users, in the form of spatiotemporal features, to classify different types of user action [142, 155, 281]. These approaches do not look for the user per se, but instead classify actions based on the motions observed in the scene. Motion-based approaches can overcome the issue of when the hand is holding an object. Whereas other techniques rely on the hand being detected, and thus may struggle when an object is being held or if the object occludes the hand completely from the camera.

Beyond the Visible Spectrum

Remote sensing techniques are dominated by vision-based techniques, however there are other sensing techniques that use other parts of the electromagnetic spectrum. Most commonly the visual and infrared wavelengths are used, however there are technologies that use wavelengths in the gigahertz spectrum, such as WiSee which uses the Wi-Fi signal to detect gestures [206], or extremely high frequency wavelengths such as those used by Google's Soli sensor [147]. Wi-Fi signals are not very accurate, and Google's Soli sensor is a radar based approach that relies on close proximity to the device.

2.5.3 Gesture Recognition

For semantic gestures, a gesture recogniser is used to interpret the semantic meaning of the body part's location, pose, or posture, irrespective of the underlying technology used for detection. Recognising gestures is non-trivial due to differences in user characteristics and behaviour which is not fixed, is often context-dependent, evolves over time, and differs between users. Recognition algorithms vary depending on the type of gesture to be detected (static vs dynamic) and on the underlying detection techniques. For static gestures, input to a recogniser could include the pose, orientation, relative position, and any other additional context, whereas dynamic gestures would also include temporal properties such as trajectory and velocity.

For static gestures, it is common to use general classifiers or template-based matching [210]. Clustering techniques are often used to classify gestures based on their input features. Common techniques include k-means [79], k-nearest neighbour [51], mean-shift [46], or support vector machines [53]. The selection of which algorithm may depend on the knowledge of how many gestures (i.e. clusters) are to be detected, and the amount of training data available. Recognising dynamic semantic gestures also requires the algorithm to take into account the temporal aspect of the gesture. Hidden Markov Models (HMM) are one of the most popular techniques because they contain an implicit solution to the gesture spotting problem (see Section 2.2.1). HMMs are finite-state automata where each state can have more than one transition arc. In the context of gesture recognition each HMM represents a gesture and the HMM with the highest forward probably determines the most likely gesture present [210]. Artificial neural networks were one of the first techniques to be used for gesture recognition [181], and the more recent deep learning-based neural networks have proved highly successful in image processing tasks including gesture recognition [195]. A single network can be trained to recognise many gestures and achieve high accuracy given sufficient training data, however, selection of the network model can be non-trivial and problems of over-fitting to the training data whereby the network does not accurately classify gestures outside the training set.

Due to the unique properties of motion correlation, gesture recognition algorithms can be much simpler than their semantic based counterparts, because only the motion path need be considered. Williamson and Murray-Smith first used a probabilistic method based on the variance of both the object position, and of input control combined with the object position [289]. Fekete et al. instead viewed the matching function as a similarity-based problem between two trajectories, and utilised a normalised Euclidean distance over a window of two seconds [73]. Work on motion correlation with smooth pursuits eye movements followed suit, and the most commonly used algorithm was the Pearson product-moment correlation. This has the advantage of being scale invariant as the trajectories to be matched do not need to be in the same coordinate space [273, 69]. Pursuits used the Pearson product-moment correlation to calculate the correlation between the x and y-axis independently [273]. If the lower of these correlation coefficients is greater than a defined threshold then the trajectories are considered to be a match.

In PathSync, Carter et al. investigated the use of different shapes for matching against and discovered a fundamental issue with the Pearson correlation coefficient used in earlier work [40]. When the coordinates of the user's movement or the target remain constant (e.g. when following a square), the standard deviation of the movement equals zero and the resultant coefficient is thus invalid. To compensate for this, they propose to rotate the data using principal component analysis (PCA) such that variation is maximised along both axes. They also propose two additional modifications for detection, the first is a bi-level threshold in which the required threshold is higher for the initial activation, but to deactivate the threshold must drop below a more relaxed threshold. The second modification was to only "select" an option if the target had been activated for more than one second (i.e. coefficient must remain above threshold for one second). They also only perform the matching process if the hands are raised (i.e. in the *active zone* of the sensor).

2.6 Application Areas

Touchless input expands interaction to where direct touch is not possible (out-of-reach), practical (hands occupied), desirable (hygiene), or safe (hazardous environments). In this thesis we specifically consider TV control as a context for touchless interaction. TV control, and more generally smart home interactions, provides a challenging context for interaction design where users tend to act spontaneously and upon impulse. Research has highlighted how users desire instant control "right now" with minimal action, and "right here" without having to go out of their ways [134]. A study estimated that in 2010 most British households had 4 or more remote controls in their living room [150], increasing the complexity of otherwise trivial tasks such as changing TV channels or dimming the lights [134]. Freeman and Weissman were first to explore the idea of controlling a television using hand gestures over 20 years ago [83], observing fatigue issues now often referred to as "gorilla arm" [97, 101]. Due to increased connectivity and technological advances, smart TVs now offer viewers more services than ever, ranging from electronic programming guides to in-built web browsing. More recent research on the topic has largely focussed on library-based gestural techniques [45, 111, 118] and highlighted issues with learning and remembering of gestures and gesture-to-function mappings [266]. More generally there is a wide range of works on universal remote control devices, including switchable [307], personalisable [99], spatially-aware [18, 290, 231] and smartphone-based remote controls [220, 58].

Touchless gestures have also been adopted for entertainment purposes. The Playstation EyeToy was the first digital camera device created specifically for touchless gesture recognition, however this was eclipsed by the Nintendo Wii which revolutionised gaming with its innovative motion controls, allowing users to control an avatar in a game using only their body movements. Microsoft's response to the Wii, the Kinect, became the fastest selling gaming peripheral

ever [212]. The Kinect leveraged depth sensing technology to remove the need for physical controllers, however it failed to make a big impact in the gaming world. Beyond console gaming the Kinect found niche applications in a number of application spaces, including virtual reality (VR). The Kinect and other body tracking devices allow users to fully immerse and interact in their virtual world using gestures or through body movements [41]. These techniques naturally extend to the augmented reality and smart glass domain, allowing users to interact with virtual objects or issue commands using free-hand gestures [144, 177, 201, 109]. In addition to hand and body gestures, the requirement for users to wear a head-mounted display in mixed reality environments has led to research into head gestures as input [8, 303, 70, 140].

Touchless gestures are suited to a wide range of applications beyond the home. Large displays raise specific challenges for interaction, because even if a user is close enough to directly manipulate the interface, there will be portions of the displays that are beyond arm's reach or out of view [26]. In addition, some interactive tasks are more suited to be performed at a distance, such as arranging photos on a large canvas or navigating a large document [276]. Touchless input enables multiple users to provide input seamlessly without having to acquire remote controls, suitable when multiple users collaborate. Another application domain with multiple users are public displays which are increasingly becoming interactive, and face similar challenges to large displays. They may be out-of-reach of users due to either logistical or security reasons, and it may be desirable for interaction to avoid direct touch for hygienic reasons [279]. Interaction with public displays may be transient, thus the primary concern of input modality lies with immediate usability because it has been shown users will often give up if they do not immediately success in interacting [160].

Building on hygienic issues, touchless input has also been investigated for use in domains that require strict adherence to hygiene protocols, such as surgery operating rooms (see Mewes et al. for a literature review on touchless interaction in interventional radiology and surgery [174]). Operating rooms present challenging conditions from a human-computer interaction perspective due to limited space, tight time constraints, and the need to maintain asepsis. Mice and keyboards are a breeding ground for bacteria and are a common method of spreading infections in intensive care units [234]. Sterile covers can be used to facilitate use of direct interaction devices (e.g. mouse, touchscreen), however these can not be used in-situ at the surgical site and it is common to delegate interaction tasks to other members of the surgical team [172, 119]. Since Graetzel et al.'s early work on mimicking standard mouse functions using hand gestures [91], more bespoke gesture-based systems have been proposed in the surgical domain using both hand and head as input, e.g. [277, 253, 260, 65, 191].

Touchless gestures also have implications for improving safety in the automotive industry. Reading and interacting with infotainment screens and satellite navigation systems contributes to driver distraction, which is one of the biggest causes of accidents on the road [257, 1]. Touchless gestures have the potential to improve safety by reducing visual demand and cognitive workload by enabling eyes-free input [123, 165, 239, 100]. Beyond the research domain, a basic vocabulary of touchless gestures can be found in BMW's luxurious 7 series range of cars [173].

2.7 Conclusion

In this section we have discussed touchless bodily interaction as natural input to computing systems for a number of applications. We have discussed existing gestural techniques and their limitations relating to the Midas touch, gesture mapping, discoverability, and memorability. The use of motion correlation addresses these issues: rather than the user learning discrete gestures they are interactively guided to synchronise with a displayed motion. Previous work has shown key insights about motion correlation which make it a compelling candidate for touchless ges-

tures and demonstrates how it alleviates some of the limitations of existing touchless gesture models:

- *Generic interaction principle.* Motion correlation is not tied to any particular input method or application. At the core of motion correlation is motion – a universal property which can be generated and expressed using any part of the body, from small eye movements [273, 69, 271], to larger hand movements [287, 73, 40, 81], previous work has shown they are all adept at providing matching motion to a target stimuli. In Chapter 3 we demonstrate how motion can be used as the primary sensing input to enable flexible input.
- *Enabling interaction with input devices not suited for pointing.* Motion correlation requires the sensing of motion and can be used for devices which may not be suitable for pointing [289], or in cases where a user is not in a comfortable position to point. The TraceMatch system introduced in Chapter 3 demonstrates how any type of movement can be used for posture-agnostic input, irrespective of whether or not the user is in a suitable position for pointing.
- *Does not require a tight coupling between input and output devices.* Only relative positions of target and user movement are required to determine if a match exists [69]. Determining a match does not rely on absolute mappings or prior calibration between input and output devices [273]. In Chapter 4, we show how the lack of tight coupling can be leveraged for defining spatial input on a per-interaction basis.
- *High discoverability and self-revealing.* All possible actions that are available to the user are revealed through the motions displayed on screen, resulting in highly discoverable interfaces, and removing the need to learn or recall pre-defined gestures [40].
- *Re-use of the same gesture shape.* Motion correlation relies on both spatial and temporal properties of the gesture, therefore the same shape can be re-used by varying the properties of the target's motion, including the speed, phase, and direction of the motion [73]. As user's are guided through the process, these properties become a distinct property of the motion space meaning a simplified gesture vocabulary can be used for an interface. In Chapter 5 we investigate four simple shapes for use with motion correlation. Such a limited gesture set is enabled by the ability to re-use the same shape.
- *High density targets.* For motion correlation, a minimum amount of space is required for the user to perceive the motion, however within this multiple target icons can be co-located as long as they have unique trajectories [69]. In contrast, cursor-based interaction requires space on a per-target basis for disambiguation due to the inaccuracies in pointing.

Previous work on motion correlation with smooth pursuit eye movements has provided insights into the number of simultaneous targets that can be selected by users. In Chapter 3, we investigate how efficient different body parts are at selecting one target amongst many shown simultaneously. Previous work with hand gestures for motion correlation showed a wide range of shapes can be used to synchronise with, and offer insights into timing of gestures, which we further explore in Chapter 5. We note the absence of any work providing interactive control beyond discrete selection with motion correlation techniques, which we investigate in Chapter 4.

3

TraceMatch: Providing Input with Any Motion

We begin by addressing the first research question - can movement be used as the primary sensing principle for interaction? At the heart of motion correlation is motion - a universal property that can be expressed by any body part. Previous motion correlation systems have relied on dedicated hardware for tracking of hand movements [40], eye gaze [273, 69], and movement produced with mouse [73], or trackpad [289]. However, the use of dedicated input devices constrains deployment of the technique and limits the ways in which users can mimic a displayed motion. In this Chapter, we present Tracematch, a vision-based system which uses motion as input. The system requires only a general-purpose camera and does not assume any particular distance or posture of the user. By leveraging the spatiotemporal properties of motion correlation, we search for the motion in a scene under the basic assumption that any movement that correlates with displayed motion must have been a result of the user. The principal motivation for TraceMatch is to enable users to select a displayed control with minimal effort (a small circular movement) and maximum flexibility (freedom to perform the movement in ways that are convenient in any given situation).

The principle behind the technique is shown in Figure 3.1: a control is presented to the user as a circular widget with an orbiting target, and the user can trigger input by performing any movement in synchrony with the displayed motion. What distinguishes *TraceMatch* as a remote input technique is the use of uniform movement that is highly appropriable. The central premise of the technique is abstraction from the different ways in which users might want to produce input. It relies on a single form of rhythmic motion, but users can perform such movement with

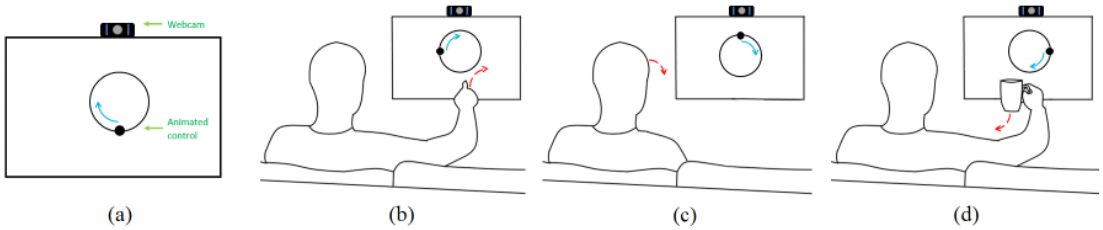

Figure 3.1: TraceMatch provides a uniform means of remote control that users can appropriate flexibly: (a) Controls are displayed as orbiting widgets; (b) Users simply mimic the motion of a control to trigger input; (c) Users can use any part of their body for input, for example their head if their hands are occupied; (d) Users can gesture input with a hand without having to put down any object they might be holding.

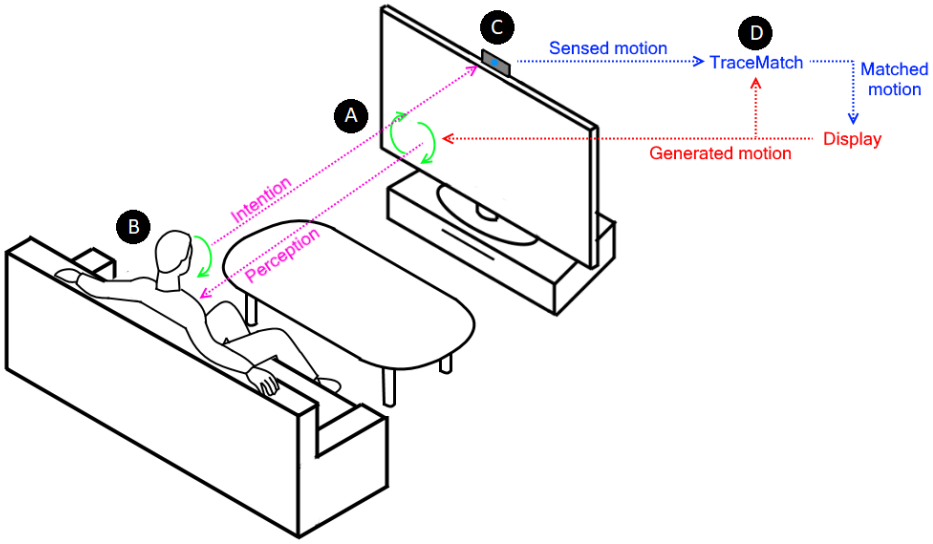

Figure 3.2: TraceMatch is a generic sensing technique for input by tracing: (A) A device displays a control as moving target; (B) The user selects the control by following the displayed motion with any part of their body; (C) A webcam serves as generic input device; (D) TraceMatch analyses the scene video for matching motion and triggers input accordingly.

different parts of their body without needing to pick up any device or put down any objects they might already be holding. This raises the question of how effective users are in producing input with different parts of their body and under different conditions, and what their preferences and spontaneous choices are. The other defining property of TraceMatch is that it uses circular motion.

Esteves et al. introduced *Orbits* as a new type of widget that displays input options as small targets orbiting the widget on a circular path [69]. Although originally designed for input by gaze, Orbits are equally compelling for gestural interaction. It is a motion we would not expect to be produced accidentally (thus avoiding the Midas touch effect) and that provides uniformity across the different ways of performing movement. Their circular design gives them a consistent button-like appearance for display, while different input options can be encoded in the direction, phase and speed of the orbiting targets. Prior work has also provided inspiring designs of controls displaying circular motion [69, 271]. However, circular motion limits variation of input, prompting questions of how reliably users are able to select one among multiple orbiting targets that vary in direction, phase and speed of movement, and how many targets can be presented at the same time without degrading input performance.

TraceMatch is a generic technique that lends itself to deployment in wide-ranging contexts, for input to any type of device that can display controls in animated form. However, our work is specifically motivated to provide users with 'lazy' input options that require minimal effort for mundane tasks, such as controlling a Smart TV or ambient lighting while lounging on a couch (e.g. selecting one from N, where $N < 10$). In such a situation the user might lean on one hand and hold a cup in their other, but they should nonetheless be able to provide input – for example by tracing the displayed motion with their head, or with their hand without having to put the cup down. Hence, we do not assume any specific posture of the user, or preference for producing gestures, but capture the entire video scene and analyse it for occurrence of *any* motion that matches a displayed control.

In this chapter, we address RQ1.1 (*Is it feasible to harness movement as the primary sensing principle using motion correlation to allow users to provide input more flexibly?*) by presenting the system implementation of TraceMatch, and an evaluation of how different parameters affect the system's sensitivity for detecting input for different target sizes and speeds while avoiding

unintended activation. The evaluation is based on a data collection with five users, where they followed the motion of orbiting targets in a variety of ways (with their head, dominant hand, non-dominant hand, and while holding a pen or a cup), demonstrating the flexibility afforded by the technique. We post-process the data collected to ensure that user behaviour was independent of any particular detection method, however this motivates questions of how users respond to the system in an interactive setting. To address RQ1.2 (*How effective are users at providing input using different body parts, or when holding objects, to generate the required motion?*) and RQ1.3 (*How do users provide input spontaneously when given the opportunity to interact with different body parts and objects?*), we present three studies that shed light on these questions and provide insight into user performance and preference, and support of 'multiple choice' tasks, in the context of remote control of a television (see Fig. 3.2). The first study is a controlled experiment in which we assess users' performance for selection of one among multiple presented Orbits with different body movements, and while holding objects. The study shows that users are effective with the technique when using different body parts, or whilst holding objects, and gives insight into effects of movement condition, speed of displayed motion, and number of simultaneously presented targets on input performance. The second study engages users with two interactive TV application prototypes with Orbits embedded for control, to gain insight into spontaneous choice of movements in realistic application contexts. The third study explored Orbit variants that integrate multiple targets in one widget for more expressive input, and probe into the use of direction, colour and speed to convey and provide multiple input options in a single Orbit.

In sum, the contributions of this chapter are:

- TraceMatch – a system which uses computer vision techniques to accept input with any technique;
- An initial optimisation of system parameters for TraceMatch, based on maximising detection of matching motions whilst minimising accidental motion;
- A validation of TraceMatch, showing that users are effective at selecting input by synchronising with displayed motion using different types of movement;
- Insight into how different ways of performing matching motion, with head, hand, or while holding objects, affect performance, and into preferences and spontaneous choices of different body movements for input;
- An exploration of ways in which the technique can be extended from binary selection to multi-level input displayed within one orbiting widget;

3.1 System Design

TraceMatch analyses a scene in two main stages (see Figures 3.3 and 3.4). The first stage involves extracting motion from the scene using image processing techniques. It is 'generous' and considers any motion in the scene as potential input. We do not segment the user, hands or other body parts but track movement of any feature. For example, the user can hold an object while they perform a hand gesture, or perform the gesture with an object. The second stage matches the observed motion against the movement of any control displayed, using a combination of path correlation and model-fitting. As an interactive system, the primary requirement of algorithms for the TraceMatch system was the capability of real-time performance. Secondarily, we minimise computational expense where possible, and aim for a CPU-only based implementation so that the system can be run on the widest range of devices (e.g. laptops, tablets). Where possible, we choose algorithms that are publicly available from the OpenCV computer vision library [1],

[1] OpenCV: https://opencv.org/

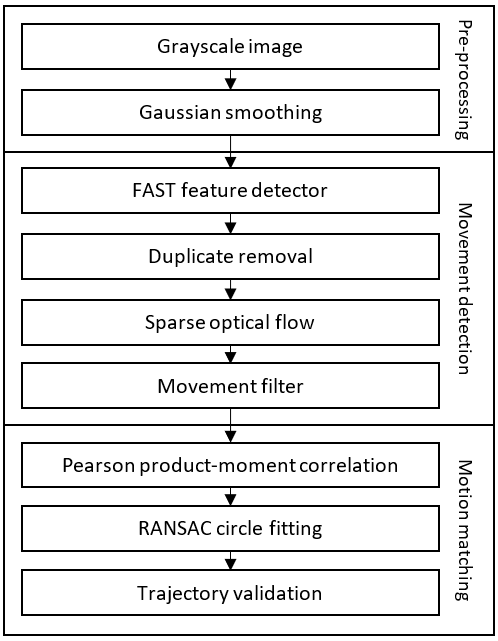

Figure 3.3: Processing pipeline of the TraceMatch system.

however we acknowledge that performance increases may be gained from using state-of-the-art algorithms from the computer vision research field.

3.1.1 Image Processing

We first convert the images captured by the camera to gray scale for feature and optical flow processing and smooth them by applying a 5×5 Gaussian kernel to reduce image noise. After preprocessing, we use the FAST corner feature detector to find points of interest in the scene [224]. A feature is a small point of interest which exhibits variation in two dimensions, such as a corner of a real-world object or a small patch of texture. For the proposed system the users may not always be in the same position therefore it is important that the feature detector is capable of detecting real-world spatial features from different camera perspectives, e.g. if we detect the arm of a user in one position we would expect to detect it in another. FAST is a fast, repeatable corner feature detector that exhibits little loss of efficiency and was designed for use in real-time video applications. FAST uses machine learning to classify whether an image patch is a corner or not before applying non-maximal suppression to further remove non-corners. FAST is not immune to high levels of noise and is also dependent on a threshold, which can be set to balance number of features detected in a scene versus processing time. There must be a balance struck between the number of features and the processing time, as the goal is for the system to run in real-time whist extracting sufficient features to match any potential motions from the user. We use a threshold of 20 for the difference between intensity of the central pixel and pixels of a circle around the centre.

In addition to FAST, we also considered other feature detectors from the OpenCV library, including SIFT, SURF, ORB, Good Features to Track, Kaze, and Brisk. Despite their accuracy, SIFT and SURF are patented algorithms which were too computationally expensive for our purpose. Initial testing demonstrated FAST to be optimal in terms of the number of features generated, their accuracy and the computational cost to do so. This was not formally tested, and it may be possible that other detectors work better under different conditions (e.g. in specific environments or with specific textures).

For each feature point we must track its position over a window of frames, W, and keep a

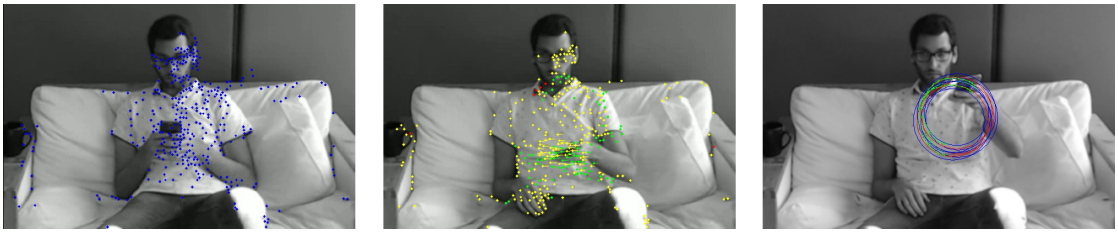

Figure 3.4: The stages of TraceMatch for matching the motion of an Orbit using a mobile phone. Left: Features (blue) are detected using the FAST feature detector. Centre: Moving features (green) are compared with the motion of an Orbit using the Pearson correlation. Right: The first feature to be matched is shown with its trajectory (green) and a fitted circle (red) found using RANSAC with inlier thresholds (blue).

history of its previous positions. In a visual scene the optical flow is the apparent motion of objects caused by an object's, i.e. the user, movement relative to the observer, i.e. the camera. Reapplying the FAST feature detector may detect the same features once they have moved, however it is unable to relate the newly detected features to the previously detected. There are two main types of optical flow algorithms - sparse optical flow, e.g. Lucas-Kanade [152] and dense optical flow, e.g. Farneback [72]. Dense optical flow techniques calculate the optical flow for each pixel, whereas sparse optical flow only calculates the optical flow for a subset of the pixels, e.g. detected features. For TraceMatch we use sparse optical flow because it is faster and more computationally efficient than the more accurate dense optical flow algorithm. We use the iterative pyramidal implementation [31] of the Lucas-Kanade optical flow method [152] to track features across image frames using a maximum of 3 pyramid levels and optical flow window size of 51×51. The Lucas-Kanade tracker is used to find a feature in all subsequent frames until it is not found, either because the feature is no longer in the scene or due to an error. We also continue to apply the FAST feature detector to every frame, in case an object we want to track enters the scene. Duplicate features may be present due to multiple detections of the FAST feature detector, therefore we discard new features if they fall within a $20 \times 20px$ area of another existing feature.

After optical flow has been performed there may be hundreds of features points for a scene, however many of these points may relate to stationary objects. In order to reduce downstream processing we record the average displacement of each feature point. Only those features that have a minimum average displacement of $0.5px$ and have been tracked for at least W frames are passed to the motion matching algorithms to detect if the motion matches with that of the rotary widget(s).

3.1.2 Motion Matching

The first step of the motion matching process is to assess the similarity between the candidate features and each orbiting target using the Pearson product-moment correlation coefficient (PCC). The PCC is calculated using the trajectories of the feature and rotary widget over a window of size W for the x axis, $corr_x$, and y axis, $corr_y$, separately. The Pearson product-moment correlation coefficient returns a value between -1 and 1. A positive value indicates a positive correlation, a negative value indicates a negative correlation, and the magnitude indicates the strength of the correlation. The Pearson correlation coefficient, r_{xy}, is defined as:

$$r_p(A,B) = \frac{\sum_{i=1}^{n}(a_i - \bar{a})(b_i - \bar{b})}{\sqrt{\sum_{i=1}^{n}(a_i - \bar{a})^2}\sqrt{\sum_{i=1}^{n}(b_i - \bar{b})^2}} \tag{3.1}$$

Where a and b are one-dimensional time-series of length n with paired values, $\bar{a}$ is the mean value of a, and $\bar{b}$ is the mean value of b. If the minimum of the horizontal and vertical, coefficients are above a minimum threshold, th_{corr}, then the feature is classed as a potential match and is passed to the next stage of the system.

Related work relied solely on PCC for motion matching [40, 69, 273], however the PCC is calculated independently for each axis which means that the circular motion of a control can be matched by elliptical and, in the extreme, up-and-down or side-to-side movement. If $corr_x$ and $corr_y$ are greater than a minimum threshold, th_{corr}, the feature's trajectory is fitted to a circle using a simple version of Random Sampling Consensus (RANSAC) to further refine the matching [74]. The centre of a circle can be found using only three points which are chosen randomly from the feature's trajectory (i.e. from the W points tracked within a window). Given three points, A, B, and C, the centre of the circle can be found by finding the intersection of the two lines perpendicular to, and passing through the midpoints of, the lines AB and BC. The centre of the circle, c, can then be found as follows:

$$c_x = \frac{m_{AB}m_{BC}(A_y - C_y) + m_{BC}(A_x + B_x) - m_{AB}(B_x + C_x)}{2(m_{BC} - m_{AB})} \tag{3.2}$$

$$c_y = -\frac{1}{m_{AB}}\left(c_x - \frac{A_x + B_x}{2}\right) + \frac{A_y + B_y}{2} \tag{3.3}$$

where m_{AB} is the gradient of AB and m_{BC} is the gradient of BC. The radius of the circle, r, is then calculated by taking the average Euclidean distance of points A, B and C to the centre of the circle. The Euclidean distance to the centre of the circle is calculated for each point in the window to assess if the feature's trajectory is circular. A data point is classed as an inlier if:

$$(1 - th_{in})r < d_i < (1 + th_{in})r \tag{3.4}$$

where d_i is the Euclidean distance from the data point to the centre of the circle, and th_{in} is the inlier threshold which should satisfy $0 < th_{in} < 1$. If at least 98% of the points on the feature's trajectory are classed as inliers, the trajectory is classed as circular. For smaller window sizes the 98% threshold will not affect the result as effectively 100% will be required, however for larger window sizes we allow for some anomalous data points (upto 2%). If there are insufficient inliers another three points are randomly selected and another circle is fitted. This continues until sufficient inliers are found or 20 iterations have elapsed.

Once we have ascertained that the data points from the image feature's window form a circle we need to assess if the arc length of the image feature's trajectory matches that of the rotary widget, i.e. if the rotary widget has completed half a circle then the image feature should have also. When calculating the arc length of the rotary widget from a practical perspective it is important to consider the frames per second (fps) of the camera. The arc length is a function of the speed of the rotary widget, the size of the window and the fps of the system. If the rotary widget has a frequency of f Hz, the camera is capturing images at N fps with a window of W then then arc length of the rotary widget, a_{RW}, is defined as:

$$a_{RW} = \frac{W}{fN} \tag{3.5}$$

We compare the arc length of the image feature's trajectory, a_F, with the arc length of the rotary widget's trajectory, a_{RC}, where a_F is defined by:

$$a_F = \sum_{i=1}^{W-1} abs(\theta_i - \theta_{i+1}).r \tag{3.6}$$

where θ_i is the angle between the ith data point and the centre of the circle. The motions of the feature and the rotary widget are matched if a_F falls in the range $a_{RC} \pm 0.1$.

3.2 Parameter Optimisation

The first study we conducted was used to optimise the system parameters, and provide initial insights into the effectiveness of TraceMatch as an input technique. Our objective was to optimise the system's sensitivity and its robustness to accidental motion matching. Our method was to collect data from users following an orbiting target under different conditions (true positive dataset), as well as a data set representing viewing and browsing activity without intent to trigger any target (false positive dataset). The recordings are then post-processed with different parameter combinations to measure how many times the algorithm determined a successful match for both true and false positive datasets.

3.2.1 Participants

Five participants (3M/2F) aged between 23 and 32 years (mean=26.6) were selected to take part in the study. Four participants were right-handed, one was was left-handed, and none of the users had previous experience with the system. We chose five participants to limit the amount of post-processing required. A larger participant number may have provided more generalisable parameters, however our aim was to acquire indicative parameters of the system for later analyses with a larger number of participants.

3.2.2 Apparatus

The study setup was designed to represent a living room scenario, with a 55" Smart TV and a couch placed 2.23m from the TV (based on a TV size to viewing distance calculator). An unmodified, off-the-shelf Logitech C920 web camera was mounted on the top of the TV. The Logitech C920 is capable of capturing 1080p (1920×1080) at 30fps, however we took a 640 x 480 region of interest in the centre of the image to control that only movement related to the simulated application setting was captured. As the data collection took place in a busy lab, a white screen was placed behind the couch to occlude the experiment.

3.2.3 Procedure

The data collection consists of two main parts: one for assessing how well the system can match a user's motion to the target's, and the other to assess how well the system rejects accidental motion. We then post-process the recordings and vary the parameters of the detection algorithm. We select the window size, W, according to the desired arc length. The system parameters were varied as follows:

- **PCC threshold ($\mathbf{th_{corr}}$)** The threshold of the minimum value for the horizontal PCC, $corr_x$, and vertical PCC, $corr_y$. The lower the value the more features are passed to the circle fitting stage of the system. [0.86, 0.89, 0.92, 0.95, 0.98]
- **Inlier threshold ($\mathbf{th_{in}}$)** The threshold which determines whether a data point is classed as an inlier or an outlier. The lower the value the closer the motion must be to a circle for it to be counted as a match. [0.05, 0.10, 0.15, 0.20]
- **Arc length (a)** The arc length of the rotary widget that must be matched, i.e. 0.5 indicates the user must follow the rotary widget for half a rotation. [0.5, 0.75, 1.0]

Target Following

In order to assess the sensitivity of the system, users were tasked with following an orbiting widget displayed in the centre of the TV screen with five different methods of input: head, dominant hand, non-dominant hand, with a pen in hand, and with a cup half filled with water in the hand. The pen and cup were chosen as common objects that a user might hold in an everyday situation. We use two blocks of testing to form a balanced Latin Square in order to minimize carrying over effects among conditions.

In order to evaluate the system we varied the direction, size and orbital period of the orbiting widgets. Based on prior work [81] we use a minimum orbital period of 1 s, and we also look at slower moving targets as used in [40]. In principle, the size of the displayed target need not influence the size of movement the user performs. Previous work has illustrated that the position of the target on the screen may influence the position performs their movement [40]. We vary the size of the target to validate it does not have an effect on the system's ability to detect a match. As a reference, the Kinect for Windows human interface guide recommends a minimum size of 220 pixels for push-to-select buttons (i.e. buttons selected using cursor-based interaction) [175]. In total there were 24 different orbiting widgets presented to the user for each movement condition:

- **Orbital period (t)** The time taken to complete one cycle of the orbiting widget. [1, 2, 4 s]
- **Radius (r)** The radius of the motion of the orbiting target. [25, 50, 100, 450px]
- **Direction (D)** The direction of the orbiting target. [clockwise, anti-clockwise]

Participants were presented with all variations of orbiting widgets in a random order. They were instructed to use whichever motion felt natural for a given movement condition, i.e. the way in which they held the cup or pen, and that it was not necessary to mimic the size of the rotary widget. For each variation the user is presented with a single widget at the centre of the screen, showing the orbit as a circle and the target as a 'dot' moving around the circle. Before the widget appeared, a 3 second countdown is shown to allow the user to rest. The widget is then presented for 7 seconds whilst the user attempts to mimic its motion with the selected movement condition. This is then repeated for all widget variations and movement conditions. A true positive is recorded when the participant successfully mimics the motion (orbital period, direction, and phase) of the orbiting target, and a false negative otherwise. These were then used to measure the sensitivity of the system.

TV Watching/Internet Browsing

In between the widget acquisition blocks participants were tasked with watching TV or browsing the internet for ten minutes, to record cases where the widget should not be activated. Participants were free to choose either activity, and in addition also casually engaged in conversation with the researcher to elicit further physical movement. This resulted in fifty minutes of recordings in which the participants were not explicitly trying to trace a control. This data set was used to assess the system's robustness to false positives (FP): when the participant inadvertently produces a movement that would match an orbiting target. When the recordings were processed, 16 orbiting targets (8 clockwise, 8 anti-clockwise) were simulated with their phases spaced equally by $\frac{\pi}{4}$ radians to detect any accidental matching.

3.2.4 Results

Two parameters sets were chosen for each orbital period based on their sensitivity and false positives (see Table 3.1). *Strict* parameters correspond to those that exhibited the highest sensitivity when aggregating all sizes, users, and movement conditions of the respective orbital period whilst having zero false positives. *Relaxed* parameters are those with the highest sensitivity when aggregating all sizes, users, and movement conditions of the relevant orbital period that produced less than 10 false positives over the 50 minute recording of background activity.

Figure 3.5 shows sensitivity results for different target sizes and frequencies. We observed the highest sensitivity for rotary widgets with an orbital period of 4 s, and low sensitivity for 'fast' targets with an orbital period of 1 s. Widgets with an orbital period of 2 s show a greater standard deviation across users, and for this orbital period we also observed that relaxation of parameters resulted in a more significant increase in sensitivity. Note we found a striking difference in optimal parameters for strict versus relaxed conditions specifically for the 2 s orbital period case. We also observed that both orbital periods of 2 and 4 s yield the same strict parameters aside from arc length. This may be due to the extra length required to suppress false positive activations if we view it from a shape matching perspective (i.e. more complex shapes will be harder to match).

Size effects were not as pronounced as differences in orbital periods. For widgets exhibiting slower movement (4 s) sensitivity increased with size. This effect did not show as clearly for smaller orbital periods. As the size increases it may be easier to discriminate the position of the target, but at a given orbital period it also implies a higher velocity of the target. The former can explain the performance increase with size for 'slow' targets (4 s), and the latter the performance decrease we observed for 'fast' targets (1 s).

Table 3.2 gives insight into performance observed for different users and movement conditions. We observed that for any user there was at least one condition for which we observed high sensitivity (0.88 or better with strict parameters, 0.94 or better with relaxed parameters). Note the differences observed, for instance between users 001 (performing well with all conditions except head movement) and 005 (highest performance with head movement). For every condition we also observed at least one user achieving high sensitivity (0.94 or better with strict parameters, and 1.00 with relaxed parameters). Interestingly, we observed the highest sensitivity across all users for movement with the cup in hand, and the worst with the pen in hand – this surprised us as we thought of a cup as a distractor, and a pen as natural for tracing. However, a cup provides more distinctive features for tracking than a pen.

Orbital period (s)	*Type*	th_{corr}	th_{in}	a	W	FP
1	Strict	0.86	0.10	0.75	23	0
	Relax	0.86	0.15	0.75	23	4
2	Strict	0.95	0.20	0.75	45	0
	Relax	0.86	0.05	0.5	30	6
4	Strict	0.95	0.20	0.5	60	0
	Relax	0.92	0.15	0.5	60	6

Table 3.1: Parameter sets used for testing. Strict sets required no false positives (FP), relaxed sets had to have less than 10 FP.

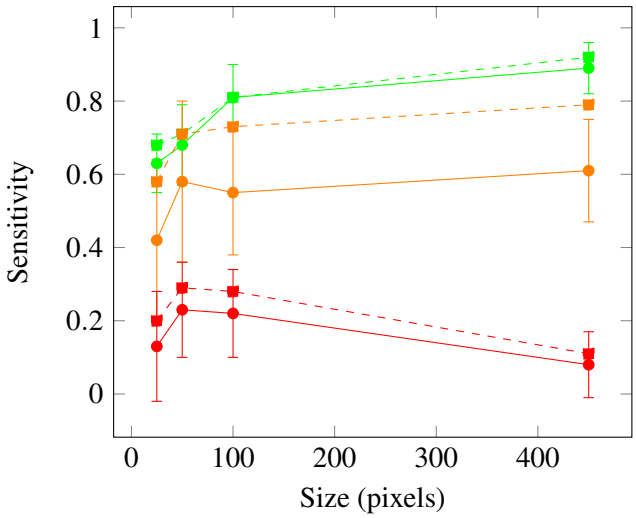

Figure 3.5: Sensitivity plotted against size for orbital periods of 4 s (green), 2 s (orange), and 1 s (red) for aggregated movement conditions and users. Strict parameter sets are shown with solid lines, relaxed parameters sets are shown with dashed lines. Standard deviation across users for strict parameter sets are shown with error bars.

		Movement Condition				
User	*Params*	*Head*	*DH*	*NH*	*Pen*	*Cup*
001	Strict	0.19	1.00	0.94	1.00	1.00
	Relax	0.19	1.00	1.00	1.00	1.00
002	Strict	0.56	0.94	0.81	0.63	0.94
	Relax	0.56	0.94	0.88	0.69	1.00
003	Strict	1.00	0.81	0.81	0.56	0.88
	Relax	1.00	0.81	0.75	0.63	0.94
004	Strict	0.88	1.00	0.44	0.44	1.00
	Relax	0.81	1.00	0.44	0.44	1.00
005	Strict	0.88	0.50	0.69	0.50	0.69
	Relax	0.94	0.56	0.69	0.50	0.75
All	Strict	0.70	0.83	0.73	0.63	0.89
	Relax	0.70	0.87	0.75	0.65	0.94

Table 3.2: Sensitivity for different movement conditions when following a rotary widget with an orbital period of 4 s for aggregated sizes.

3.2.5 Discussion

Our study of TraceMatch demonstrates that the method is effective in matching a user's motion with different sizes of an orbiting target using an unmodified webcam. We observed high sensitivity for different movement conditions, highlighting the system's ability to abstract the motion matching technique from any particular body part. This gives the users flexibility to provide input how they choose and enables seamless interaction during other activities where the user, for instance, may be holding an object.

The results indicate that movement conditions can affect tracking performance but also show individual differences prompting further investigation of user preference and ability in motion following with different parts of their body. Our evaluation was focussed on the performance of the vision-based sensing system and provides insight on parameter choices for the design of interactive applications. We observed a low success rate for targets with the fastest speed. This could be due to either user limitations in following the moving target, or technical limitations of detecting such quick user motions. In our study, we limited tracing to a single visible target for data collection, but for applications selection from among a number of targets is important, prompting further work on how the technique scales. As our study was designed to facilitate parameter exploration, users did not have any feedback on how well their movement matched a target. We would expect that feedback will positively affect input performance.

3.3 Task Success with Different Body Movements

To study how effective users are in producing input with different parts of their body under different conditions (RQ1.2), we undertook a separate user study in which participants used the TraceMatch system for and interactive tasks. Users performed a series of trials which required them to follow the motion of a randomly selected target Orbit from multiple presented Orbits. During the study, we measured the task success rate whilst varying the motion of the displayed Orbits with respect to their SIZE, i.e. the radius of the Orbit (25 and 50px), ORBITAL SPEED (2 and 4 seconds per cycle), DIRECTION (clockwise and anti-clockwise), and the NUMBER OF ORBITS (2, 4, 6, 8, and 4 plus 4) displayed simultaneously.

We maximised the phase difference between Orbits by $360^\circ/n$, where n is the number of Orbits displayed simultaneously in the same direction. The "4 plus 4" variable consisted of four Orbits rotating clockwise, and four anti-clockwise, displayed simultaneously with a 90° phase difference. This was included to investigate if the number of Orbits displayed simultaneously affected the participants' performance. We expected this combination to have similar results to when four Orbits of one direction were displayed.

The TYPE OF MOVEMENT (head, dominant hand, non-dominant hand, mobile phone-in-hand and cup-in-hand) participants used to match the motion of the Orbits was also varied. We used the cup and mobile phone as everyday objects that users would likely use for multi-tasking whilst interacting with the system in a real-life setting, e.g. drinking or sending a message. In the previous data collection we used a pen, however we observed participants performing much smaller movements (e.g. tracing with the tip) which could lead to detection issues. We therefore swapped the pen with a mobile phone, another everyday object people would likely have in hand. The cup was half filled with water to simulate the participants holding a drink, and the experimenter's Samsung Galaxy S5 was used in the event a participant did not have a mobile phone. In total, this resulted in $2 \times 2 \times 2 \times 5 \times 5 = 200$ trials per participant, and $20 \times 200 = 4000$ trials for the study.

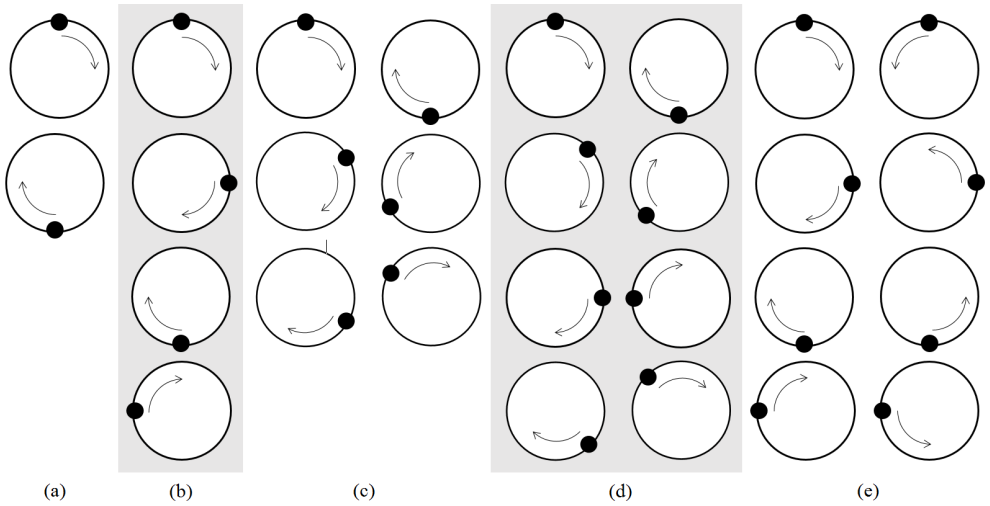

Figure 3.6: Configurations for the number of Orbits: (a) two, (b) four, (c) six, (d) eight, and (e) four plus four. Orbits shown rotating clockwise, with the exception of (e) where Orbits rotate in both directions.

3.3.1 Participants and Apparatus

Twenty participants (10M/10F) aged between 21 and 54 years (mean=29.4, sd=8.21) were selected to take part in the studies. Eighteen participants were right-handed and two were left-handed. None of the users had previous experience with, or knowledge of, the TraceMatch system. All three studies were undertaken sequentially by participants in one sitting, and took approximately one hour to complete per participant. Participants were compensated with £10.

The studies took place in a lab designed to represent a living room scenario. A 55" Smart TV (1920x1080) was used as the display, with a couch placed 2.23m from the TV (based on a TV size to viewing distance calculator). An unmodified, off-the-shelf web camera was mounted on top of the TV. The camera captured a 640×480 region of interest in the centre of a 1920×1080 image to control that only movement related to the simulated application setting was captured.

3.3.2 Procedure

Upon arrival, participants signed a consent form and completed a demographics questionnaire. They were then presented with a basic overview of TraceMatch, which did not include any technical detail. Participants were instructed to find a comfortable position anywhere on the couch. Participants were not given instructions on how to perform the type of movements, but were told that the size of their movement did not have to correlate with the size of the Orbits. Following the introduction, participants took part in a practice session which involved all the variables, excluding number of Orbits, used for the trials. Participants spent, on average, less than four minutes during the practice session.

For each trial, a number of Orbits were presented simultaneously to the participants (see Fig. 3.6), with a target Orbit highlighted in blue. If the participant successfully matched the motion of the target Orbit it turned green and the next trial was presented. If the participant activated an Orbit other than the target, the Orbit they activated flashed red and the trial was unsuccessful. The task was not completed until the participant successfully matched the motion of an Orbit, or ten seconds elapsed. A three second countdown preceded each trial.

A balanced Latin Square design was used to counter balance the different types of movement and minimize carry over effects. A 5×5 Latin square and its mirror image were used, resulting in multiples of 10 participants required for counterbalancing the type of movement. Participants

completed trials in ten blocks, one block for each combination of speed and type of movement. Ten participants, counterbalanced for type of movement, performed the trials with slow Orbits (4 seconds per cycle) followed by fast Orbits (2 seconds per cycle), the other ten, also counterbalanced for type of movement, performed the opposite. For a given speed, participants used all types of movement before changing speeds.

For each block, participants were presented with each number of Orbits in order (i.e. 2, 4, 6, 8, then 4 plus 4), for large Orbits (50px) first and then with small Orbits (25px). All Orbits presented to participants rotated in the same direction when displayed on the screen, with the exception of the 4 plus 4 variable. Participants were shown both clockwise and anti-clockwise directions for each size, the order of which was determined randomly. The 4 plus 4 configuration was displayed twice per size, the first with the target Orbit rotating in one direction, the next with it rotating in the other direction. The ordering of the rotation of direction for the target Orbit was randomised.

After each block, participants completed a questionnaire consisting of six 5-point Likert items:

- I felt comfortable following the targets
- I felt confident following the targets
- I found it easy to synchronise with the position of the targets
- I found it easy to follow the movement of the targets
- It was not physically demanding
- It was not mentally demanding

After all of the trials were completed, participants were verbally asked about their preferred type of movement and speed.

3.3.3 Results

We used a four-way repeated measures ANOVA, Greenhouse-Geiser-corrected in the cases where Mauchly's test indicated a violation of sphericity and with Bonferroni-corrected post-hoc tests where applicable, to test for the effects of the type of movement, speed, size, and number of Orbits, averaged over direction (clockwise and anti-clockwise), on the task success rate. The task success rate is the number of times the participants correctly selected the target Orbit, divided by the total number of trials. A trial was deemed unsuccessful if the participant did not activate an Orbit within 10 seconds, or if an Orbit other than the target was activated. Figure 3.7 shows the task success rate for each type of movement across all variables after averaging for size and direction.

We found significant main effects for speed ($F_{1,19} = 7.72, p = .012$), and number of Orbits ($F_{4,76} = 103.01, p < .001$). There were no significant main effects for size ($F_{1,19} = 2.935, p = .103$), or type of movement ($F_{2.57,48.79} = 2.92, p = .051$). In general, participants performed significantly better with the slow speed (85%) compared with the fast speed (76%). As we expected, participants performed significantly worse when selecting the target from 8 Orbits (57%) compared with all others, at $p < .001$. We also observed a significant difference when participants selected a target from 6 Orbits (76%) compared with all others at $p < .001$. There were no further significant differences when selecting a target from 2 (92%), 4 (88%) or 4 plus 4 (88%) Orbits.

We observed significant two-way interactions for type of movement x speed ($F_{4,76} = 8.77, p < .001$), type of movement x number of Orbits ($F_{7.50,142.51} = 2.56, p = .014$), and speed x number

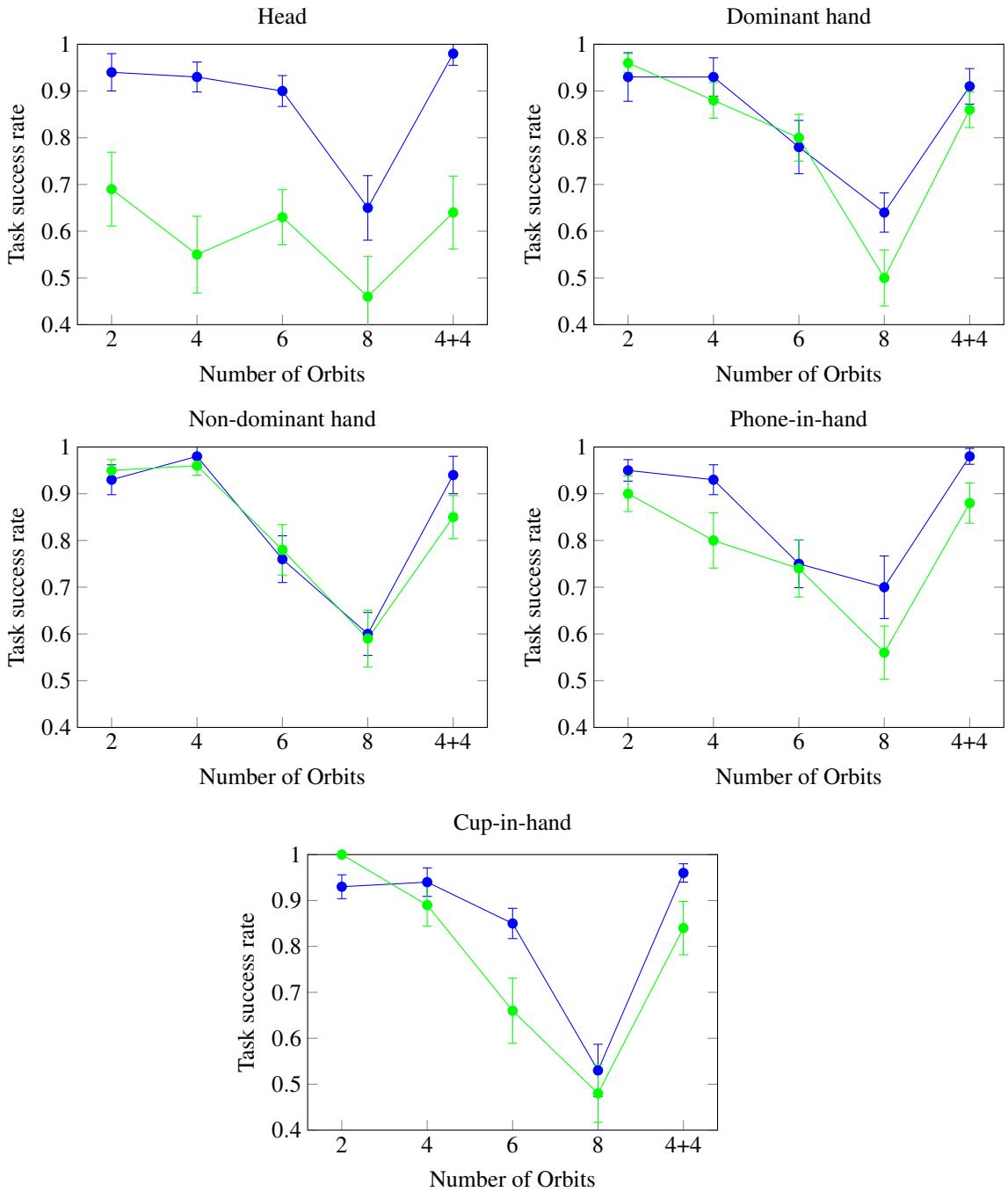

Figure 3.7: Task success rate for each type of movement when following slow (blue), and fast (green) Orbits, averaged for size and direction, plotted against each level of the number of Orbits variable. Standard error is shown with error bars.

of Orbits ($F_{2.74,52.12} = 3.04, p = .041$). There were no other significant interactions. We further investigate the simple main effects for the types of movement, proceeded by simple main effects for both speeds and all number of Orbits.

Type of Movement

We found no significant simple main effect for slow movements, when averaging over size and number of Orbits, between the head (88%), dominant hand (84%), non-dominant hand (84%), phone-in-hand (86%), and cup-in-hand (84%), ($F_{2.26,42.97} = 0.85, p = .45$). However, for fast movements there was a significant simple main effect for types of movement ($F_{2.26,42.97} = 6.52, p = .002$). Participants performed significantly worse with the fast head movement (59%) compared with the fast dominant hand (80%), and fast non-dominant hand (83%) movements, at $p = .04$ and $p = .021$ respectively. We found no significant differences between the fast head movement and the fast phone-in-hand (78%) or fast cup-in-hand movements (77%).

Head

For the head movement, we found significant simple main effects for speed ($F_{1,19} = 15.31, p = .001$), and number of Orbits ($F_{4,76} = 14.97, p < .001$). Participants performed significantly better with the slow head movement (88%) compared with the fast (59%). Participants were significantly worse when using the head movement (averaged over speed) to select a target from 8 Orbits (56%) compared with 2 (81%), 4 (74%), 6 (76%), or 4 plus 4 Orbits (81%), at the $p < .005$ level.

Dominant Hand

For the dominant hand movement, we found a significant simple main effect for the number of Orbits ($F_{2.89,58.84} = 27.40, p < .001$), but no simple main effect for speed. The task success rate was significantly lower when participants selected a target from 6 Orbits (79%) compared with 2 Orbits (94%), at $p = .013$, but was significantly higher compared with selecting a target from 8 Orbits (57%), at $p = .006$. We also observed that the task success rate was significantly lower when selecting a target from 8 Orbits compared to all others, including 4 (90%) and 4 plus 4 Orbits (88%), at $p < .01$.

Non-dominant Hand

For the non-dominant hand movement, we found a significant simple main effect for the number of Orbits ($F_{2.55,48.44} = 25.52, p < .001$), but no significant simple main effect for speed. Selecting a target from 6 Orbits (77%) resulted in a significantly lower task success rate compared with 2 Orbits (94%), and 4 Orbits (97%), at $p = .011$ and $p = .001$ respectively. The task success rate for selecting a target from 8 Orbits (59%) was significantly lower than 2, 4, and 4 plus 4 (89%) Orbits at $p < .001$. There was no significant difference between 6 and 8 Orbits.

Mobile Phone-in-hand

For the mobile phone-in-hand movement, we found significant simple main effects for both speed ($F_{1,19} = 5.84, p = .026$), and number of Orbits ($F_{4,76} = 18.11, p < .001$). Selecting a slow moving target resulted in a significantly higher task success rate (86%) compared with selecting a fast moving target (78%). Selecting a target from 6 Orbits (74%) resulted in a significantly

lower task success rate compared with 2 (93%), 4 (86%), and 4 plus 4 Orbits (93%), at $p = .005$, $p = .044$, and $p = .002$ respectively. We also found a significantly lower task success rate when participants selected a target from 8 Orbits (63%) compared with 2, 4, and 4 plus 4 Orbits at $p < 0.005$. There was no significant difference between selecting a target 6 or 8 Orbits.

Cup-in-hand

For the cup-in-hand movement, we found a significant simple main effect for the number of Orbits ($F_{4,76} = 42.91, p < .001$), but no significant simple main effect for speed. Selecting a target from 8 Orbits (50%) resulted in a significantly lower task success rate compared with all others for $p < .001$. In addition, the task success rate as a result of selecting a target from 6 Orbits (76%) was significantly lower compared to 2 Orbits (96%) at $p = .003$, and both 4 Orbits (91%) and 4 plus 4 Orbits (90%) at $p = .13$.

Speed x Number of Orbits

For the slow speed, there was a significant simple main effect for number of Orbits when averaging types of movement and size ($F_{2.31,43.96} = 69.33, p < .001$). The task success rate was significantly lower for 8 Orbits (62%) compared with all others at p $< .001$, and for 6 Orbits (80%) and all others at $p < .001$. There were no other significant differences between selecting a target from 2 (93%), 4 (94%), and 4 plus 4 (95%) Orbits.

There was also a significant simple main effect for number of Orbits for the fast speed ($F_{4,76} = 55.27, p < .001$). Post-hoc tests revealed a significant difference between 8 Orbits (52%) and all others at $p < .001$. The task success rate was also significant lower when selecting a target from 6 Orbits (72%) compared with selecting a target from 2 (90%), 4 (82%), or 4 plus 4 Orbits (81%), at $p = .001$, $p = .002$, and $p = .019$ respectively.

When taking all sizes and types of movement into account, there was a significant simple main effect for speed when selecting a target from 4 Orbits, ($F_{1,19} = 11.95, p = .003$). Slow moving targets (94%) resulted in a significantly higher task success rate compared with fast moving targets (82%). Slower moving targets (95%) also resulted in a significantly higher task success rate than the faster moving targets (81%) when selecting a target from 4 plus 4 Orbits, ($F_{1,19} = 18.48, p < .001$). There were no significant simple main effects for speed for 2, 6, or 8 Orbits.

3.3.4 Activation Time

Figure 3.8 shows the activation times for successful trials. Average activation times across all users are reported in brackets. The minimum time for activation of the slow and fast Orbits was 2 and 1 seconds respectively. For fast movements, it takes the head movements (4.1s) longer to acquire than the dominant hand (3.3s), non-dominant hand (3.2s), phone (3.5s) and the cup (3.2s). As figure 3.8 illustrates, there were participants who achieved activation times with the head matching those of the other movement conditions. For slow movements, the head (4.1s) was once again slower than the dominant hand (3.6s), non-dominant hand (3.6s), phone (3.5s) and cup (3.5s). The spread of activation times for the slow head movement is less than the faster Orbit, but still larger than those of the other movement conditions for the slow speed. Based on the activation time, we also calculate the average information transfer rate using the generalised form of Shannon's formula [96], see Table 3.3.

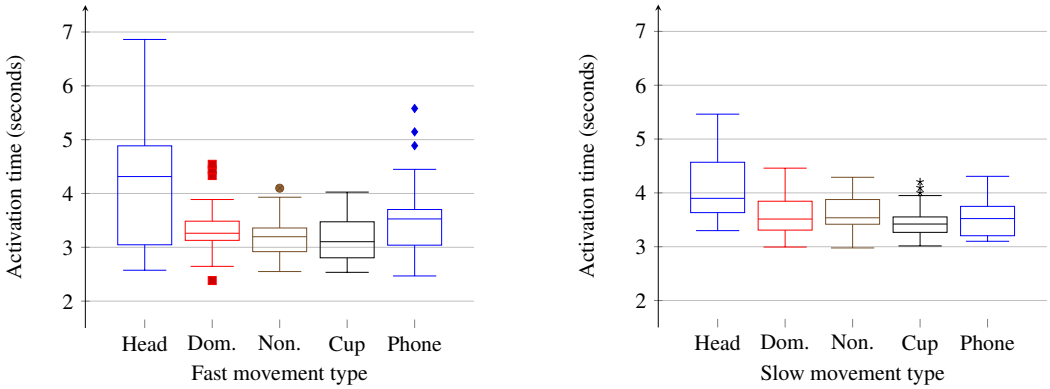

Figure 3.8: Box plots showing average activation time of participants for fast Orbits (left) and slow Orbits (right).

Table 3.3: Average information transfer rate (bits/s) based on Shannon's generalised formula (standard deviation in brackets).

		Number of Orbits				
Input Type	Speed	2	4	6	8	4+4
Head	2	0.47 (0.08)	0.5 (0.09)	0.74 (0.19)	0.82 (0.2)	0.56 (0.15)
	4	0.47 (0.0)	0.7 (0.06)	0.85 (0.11)	0.86 (0.16)	0.75 (0.07)
Dom.	2	0.57 (0.03)	0.74 (0.12)	0.96 (0.18)	0.99 (0.26)	0.77 (0.15)
	4	0.54 (0.04)	0.74 (0.09)	0.81 (0.15)	0.96 (0.14)	0.72 (0.11)
Non.	2	0.61 (0.04)	0.83 (0.16)	0.95 (0.24)	0.98 (0.26)	0.77 (0.19)
	4	0.53 (0.01)	0.77 (0.08)	0.83 (0.17)	0.93 (0.21)	0.78 (0.12)
Phone	2	0.5 (0.06)	0.71 (0.12)	0.83 (0.11)	1.06 (0.28)	0.72 (0.09)
	4	0.56 (0.05)	0.73 (0.09)	0.85 (0.18)	1.05 (0.2)	0.79 (0.11)
Cup	2	0.61 (0.07)	0.77 (0.07)	0.96 (0.22)	1.08 (0.33)	0.74 (0.09)
	4	0.53 (0.02)	0.76 (0.08)	0.91 (0.17)	0.93 (0.16)	0.79 (0.11)

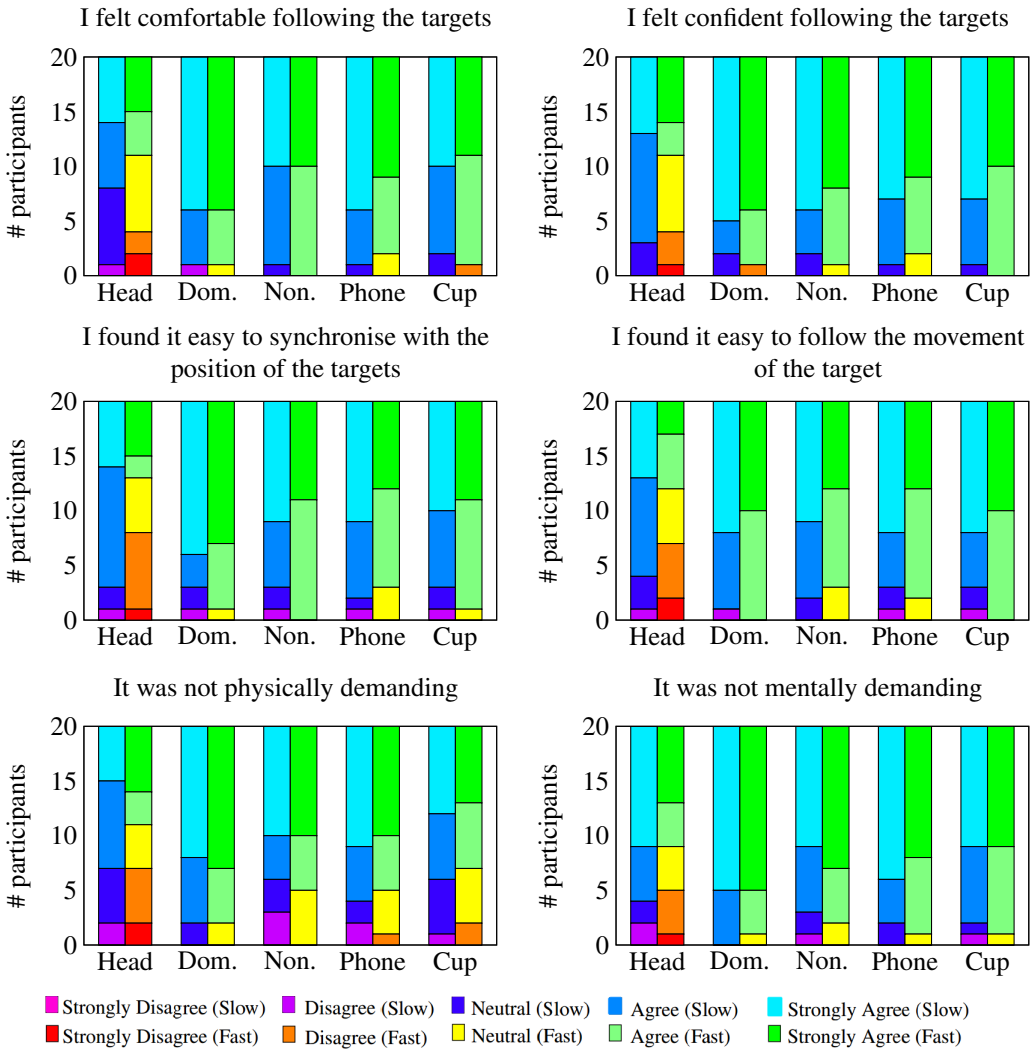

Figure 3.9: Stacked bar charts showing responses to the Likert items for different types of movement and speed.

3.3.5 User Preferences

The most preferred type of movement was the dominant hand (12), followed by the head (3) and phone-in-hand (3), and finally the cup-in-hand (2). No participant selected the non-dominant hand as their favourite type of movement to use. Ten participants preferred the faster targets, and ten participants preferred the slower. Six participants preferred the faster targets with the dominant hand movement, whereas the other six preferred the slower targets. All of the participants who selected the head movement preferred the slower targets, and all of the participants who selected the cup-in-hand preferred the faster targets. One participant preferred the slow targets with the phone-in-hand movement, with the remaining two preferring the faster targets.

3.3.6 Likert Item Responses

We performed a Friedman test on each Likert item to investigate participants' responses for the types of movement and speeds, see Fig. 3.9. Pairwise comparisons were performed with a Bonferroni correction for multiple comparisons.

Participant responses to the comfort Likert item were significantly different across the movement speed combinations, $\chi^2(9) = 41.15, p < .0005$. Responses for the fast head movement (Mdn =

3) were significantly lower than both slow phone-in-hand (Mdn = 5) and fast dominant hand movements (Mdn = 5), at $p = .041$ and $p = .028$ respectively.

There was a significant difference in responses when participants were asked how easy it was to synchronise with the target, $\chi^2(9) = 28.92, p = .001$. Participants felt it was significantly harder to synchronise with the target using the fast head movement (Mdn = 3), compared with both slow dominant hand movement (Mdn = 5) and fast dominant hand movements (Mdn = 5), at $p = .021$ and $p = .008$ respectively.

When participants were asked how easy it was to follow the target, responses were significantly different based on speed and movement condition, $\chi^2(9) = 34.18, p < .0005$. Participants felt it was significantly harder to follow the target with the fast head movement compared with both slow (Mdn. = 5) and fast (Mdn. 4.5) cup-in-hand movements, at $p = .023$ and $p = .014$, and compared with both slow (Mdn = 5) and fast (Mdn = 4.5) dominant hand movements, at $p = .012$ and $p = .014$ respectively. The fast head movement (Mdn = 3) also received significantly worse responses compared with the slow non-dominant hand (Mdn = 5), and slow phone-in-hand movement (Mdn = 5), at $p = .031$ and $p = .028$.

There was a significant difference when participants were asked about the physical demand of using the movement and speed combinations to select a target, $\chi^2(9) = 32.74, p < .0005$. Participants reported significantly more physical demand for the fast head movement (Mdn = 3) compared with the fast dominant hand movement (Mdn = 5) at $p = .049$.

Although there were significant differences between responses to the confidence and mental demand Likert items, $\chi^2(9) = 30.31, p < .0005$ and $\chi^2(9) = 32.55, p < .0005$ respectively, post-hoc tests revealed no significant differences between speed and movement combinations after accounting for multiple comparisons.

3.3.7 Discussion

The results show that holding an object does not significantly affect the task success rate, nor does using the non-dominant hand – unlike other tasks such as writing in which the non-dominant hand performs significantly worse. This highlights the ability to abstract from specific body parts, and provides users with various means by which to successfully interact with the system in the event their preferred method of input can not be used (e.g. when performing other tasks). The lack of difference between hands could also suggest the algorithms do not pick up on the fine motor control of the user – is the task success rate linked to user behaviour or algorithm's ability to detect? We investigate whether this is the case in more depth in section 5, which would suggest it is due to user behaviour.

Interestingly, we observed that using the head achieved the highest task success rate for slow movements, but the lowest task success rate for fast movements. Many users reported that the fast head movement was "uncomfortable" and felt "unnatural". The low task success rate can be, in part, explained by the way some users performed this movement. The experimenter noted that during the trials, the output of the system (not seen by participants) was reporting that the participants' fast head movements were passing the Pearson threshold, however no activation occurred. This infers that the participants were following the motion of the target, however their movements were not circular enough to pass the circle fitting stage of the matching process, i.e. the movements were elliptical. This is further exaggerated because fast Orbits require more constrictive parameters, compared with slow Orbits, to avoid accidental matching with background movements.

The task success rate across all sizes and number of Orbits when taking into account participants' preference for type of movement and speed is 87%. This rises again to 97% for participants' preferred type of movement and speed across both sizes if we only consider selecting a target

from 2, 4, and 4 plus 4 Orbits (99% for 2, 96% for 4 and 95% for 4 plus 4). This demonstrates that, despite TraceMatch's generic approach, users can successfully interact with the system using their preferred type of movement.

We observed individual differences between participants depending on the type of movement used. The participant with the best overall task success rate across all variables had a task success rate of 90%, whereas the worst had a task success rate of 67%. However, for the participant with the lowest overall task success rate, the task success rate for their preferred movement and speed was 85% across all variables (100% excluding when they selected a target from 6 and 8 Orbits). This is an example of when a participant had a much lower task success rate for other types of movement and speed combinations, as can be seen by the low average, yet there was at least one type of movement for which they achieved a very high task success rate.

According to the responses to the Likert items, there were no significant differences found other than for the fast head movement. This validates the idea of abstracting from body part segmentation and providing the user with a choice of how to interact with the system or, in the event the user is performing another task, allowing the user to continue interacting with the system whilst holding an object. The different preferences we observed for the speed of the Orbits could easily be implemented using a "settings" option, allowing users to tailor the system based on their personal preferences.

When discussing their favourite type of movement, one participant thought that their preference would depend on whether or not they were in a social situation. The participant preferred the slow head movement due to the "low-effort" involved, however, they stated that in a social situation they would rather use the hand gesture. This is because they would only need to glance at the Orbit to be able to follow the target, thus allowing them to control the system whilst maintaining their interaction in the social situation. This is another advantage of abstracting from the use of specific body parts, allowing users to interact in different ways depending on the situation.

The activation time in TraceMatch consists of the time taken for the user to locate the control they wish to activate, to position their desired movement condition (e.g. raise their hand), to start synchronisation with the Orbit, and to maintain synchronisation for the required amount of time (i.e. half an Orbit). Before starting the synchronisation, users may wait and choose to start the movement at a salient point, e.g. when the target is at the top of the Orbit. Interestingly, one would expect the slower times to be around one second slower for the slow Orbits due to the extra time required to synchronise with half an Orbit, 1s and 2s for fast and slow Orbits respectively. However, we observed a difference of less than half a second, suggesting that the slower Orbits are easier to synchronise with than the faster Orbits, as all other factors that contribute to the acquisition time remain the same (e.g. finding the target and positioning the body part ready for interaction).

3.4 Choice of Movement for Interaction

To investigate RQ1.3, we studied how participants interacted spontaneously with the system using real-world applications, an Interactive Story and a Formula 1 Multi-screen application. We used the same participants and apparatus as the previous study. Participants were free to use any type of movement to interact with the prototypes. Participants were instructed to inform the experimenter in the event of an incorrect activation, i.e. the wrong Orbit is activated when trying to activate an Orbit, or a false activation, i.e. an Orbit is activated when not trying to activate an Orbit. The Orbits used for the prototypes had a radius of 50px and speed of 3 seconds per cycle.

3.4.1 Interactive Story

The aim of the first prototype was to assess how users interact with the system given an application in which the participant has minimal interaction. For this, we used an interactive video series about knife crime filmed from a teenager's point of view [16]. Participants are shown a film, during which they are offered a series of choices throughout which influence the outcome. In order to choose which route to pick users are presented with a textual description and associated Orbit to select an action, see Figure 3.10. Two additional Orbits were added to restart the story from the beginning, or replay the last video section. Participants were instructed to choose whichever actions they preferred, and that their choices were not being recorded. At the end of the story, participants were presented with four 5-point Likert items:

- I felt comfortable following the targets
- I felt confident following the targets
- It was not physically demanding
- It was not mentally demanding

3.4.2 Formula 1 Multi-screen Application

The second prototype was a Formula 1 Multi-screen application, see Figure 3.10. The aim of this prototype was to present an application to users with a large number of Orbits (8) displayed simultaneously on the screen. The interface allowed participants to choose between four different camera angles, a timing screen and the track layout. Controls for muting the sound and enlarging the main window were also included. Participants were instructed to select each Orbit at least once in any order and to watch the videos if they desired. Once the participant had finished interacting with the prototype, they completed the same Likert items used for the *Interactive Story* prototype.

3.4.3 Results

The task success rate for the Interactive Story application was 100%, out of 70 activations, see Table 3.4. The most commonly used type of movement for activating the controls was the dominant hand, which was used for 70% of the activations. The cup was the only type of movement that was not used by any of the participants. Four participants used more than one type of input throughout the duration of the interactive story.

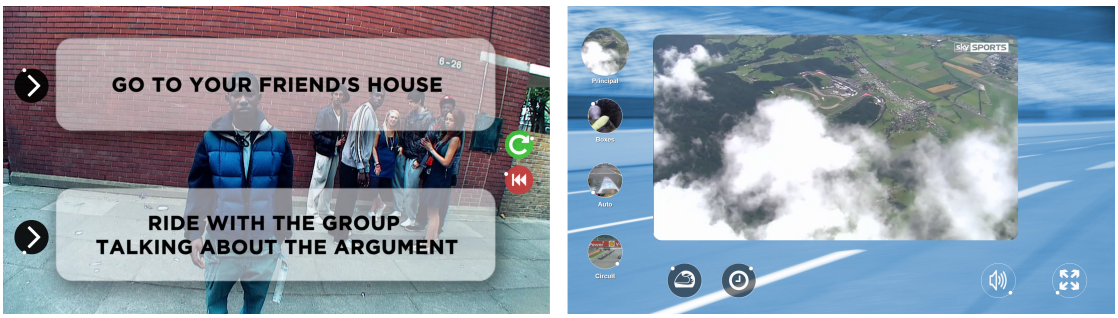

Figure 3.10: Prototypes for the second study. Left: Interface for the Interactive Story prototype with Orbits for selecting an action (left of the screen), and for restarting the story from the beginning or replaying the last section (right of the screen). Right: Interface for the Formula 1 Multi-screen prototype with Orbits for changing the main display (left), muting the volume (second from right), and enlarging the display to full-screen mode (right).

Table 3.4: Results for the second study, showing the different types of movement used to activate the Orbits and the overall task success rate.

		Interactive Story	Formula 1 Multi-Screen
Number of activations	Head	5 (7%)	18 (6%)
	Dom. hand	49 (70%)	223 (76%)
	Non. hand	6 (9%)	22 (8%)
	Phone	10 (14%)	28 (10%)
	Cup	0 (0%)	0 (0%)
	Foot	0 (0%)	2 (1%)
Total activations		70	293
Incorrect activations		0	8
False activations		0	0
Task success rate		100%	97%

According to responses from the Likert items, most participants felt comfortable (Mdn = 4.5) and confident (Mdn = 5) with the interactive story. Participants did not report any physical (Mdn = 5) or mental demand (Mdn = 5) with a bottom-two-box score of 0% for all Likert items relating to the interactive story.

When faced with multiple Orbits simultaneously during the Formula 1 multi-screen application, participants achieved a task success rate of 97% out of 293 activations. Three participants encountered one incorrect activation, and two participants encountered two incorrect activations. During the Formula 1 multi-screen prototype, participants used a wide variety of movements, including a participant who successfully activated an Orbit with their foot. The most frequently used type of movement was, again, the dominant hand which was used for 76% of all activations.

For the Formula 1 multi-screen, the majority of participants did not report any physical (Mdn = 5) or mental (Mdn = 5) demand, and reported that they felt comfortable (Mdn = 5) and confident (Mdn = 5). One participant disagreed that they felt comfortable, and thought the Formula 1 multi-screen application was mentally (2) and physically demanding (2). The participant reported that it was much harder to follow targets with a video in the middle of the Orbit (for the Orbits which previewed video content). We observed nine participants using multiple types of input when interacting with the Formula 1 Multi-screen prototype.

The type of movement predominantly used to interact with the prototypes was not the preferred type of movement of the participant in all cases. Two of the participants who preferred the cup object used their dominant hand to interact with the prototypes, with the third choosing to use their smartphone. Only one out of the three participants who preferred the head movement used their head to activate the Orbits for the majority of the time, the remaining two used their dominant hands for the majority of the time when interacting with the prototypes. One of the participants who preferred the phone-in-hand movement predominantly used their non-dominant hand to activate the Orbits during the prototypes.

3.4.4 Discussion

When users were given the freedom to interact with the system in a more natural setting, we observed that not all participants used their preferred type of movement reported during the first study. In the case of the head movement, this could be due to the increased speed (3 second per cycle) used for the prototypes, because all those who preferred the head movement also preferred the slower moving targets (4 seconds per cycle).

With the exception of one participant, those who preferred objects did not actively seek out their preferred objects to interact with the system, instead choosing to perform the movements without an object or with a different object. This demonstrates the flexibility when it comes to

choice of how to provide input and shows that users, although reportedly preferring one type of movement, can easily adapt to different types of movement.

One participant used the head movement, their preferred movement, during the interactive story, however switched to the dominant hand when interacting with the Formula 1 multi-screen application. We also saw similar behaviour with participants switching from their non-dominant hand in the interactive story, to their dominant hand in the Formula 1 multi-screen application. The story application is much more relaxed and requires less input than the multi-screen application, which could suggest that users change their type of movement depending on the context.

We observed four participants using different hands depending on the location of the Orbit they were trying to activate. When questioned about this, participants reported that they instinctively changed which hand they used based on the location of the Orbit on the screen. We also observed one participant who used their foot to activate the multi-screen prototype, in an explorative manner.

3.5 Multi-Level Input

To gain insight into how multiple targets on a single Orbit can be used for more expressive input we used two prototypes, a Video Control and an Information Popup application. We can display information with the Orbits themselves, e.g. through the use of background icons, but our aim here is to investigate whether additional information can be conveyed through the movement and colour of multiple targets orbiting around a single Orbit without the participants having prior knowledge of the functionality of the different targets. We used the same participants and apparatus as the previous study.

3.5.1 Video Control

The aim of the Video Control prototype was to simultaneously present Orbits with different sizes and speeds to the participants. For this, we designed a video controller which allowed the user to play, pause, skip forwards or skip backwards, see Figure 3.11. We chose skipping forwards and backwards to provide a non-continuous method of control, as opposed to rewinding or fast forwarding.

For skipping forwards, three clockwise Orbits were used with different speeds and sizes. The small, medium and large Orbits rotated with speeds of 4, 3, and 2 seconds per cycle respectively. The larger, faster Orbit skipped the video forward by 30 seconds, the medium sized Orbit by 15 seconds, and the smaller, slower Orbit by 5 seconds. For skipping backwards the Orbits operated the same way but rotated anticlockwise. The participants were not told about the functionality of the Orbits prior to their interaction with the prototype, however the number of seconds skipped was displayed in the middle of the Orbit when the control was activated, i.e. "5s" is displayed if the small, slow Orbit is activated.

Participants were asked to interact with the Video Control for two minutes. They were then verbally asked if they had understood the functionality of the Orbits for skipping forwards and backwards (i.e. the quicker the speed and larger the size the greater amount the video was skipped) when they first saw the controls, and after they had finished with the prototype. They were then verbally asked whether they thought the functionality of the different sizes and speeds made sense.

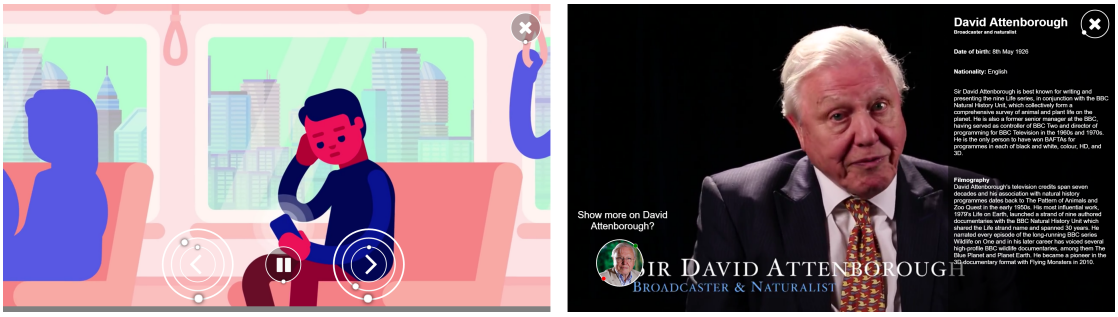

Figure 3.11: Prototypes for the third study. Left: Interface for the video control prototype with Orbits for skipping backwards (left), skipping forward (right), play/pause (middle) and for hiding the controls (top-right). Right: Interface for the information popup with Orbits for opening the popup (left) and closing the popup (top-right). The Orbit used to open the popup is shown for illustration and would not be visible at the same time.

3.5.2 Information Popup

The final prototype was used to test whether the colour of the targets could be used for selection or rejection of a popup. This was achieved using a prototype that allowed the participant to view additional information on a subject at four key points during a video, see Figure 3.11. The Orbits used for the Information Popup application had a radius of 50px and speed of 3 seconds per cycle.

For each key point, an Orbit with a red and a green target was displayed. If the participant selected the green target, the additional information would appear. If the participant selected the red target, the Orbit disappeared. We also varied how the red and green target orbited to find out which participants preferred. In two out of the four cases, the targets rotated clockwise with an offset of 180°. In the remaining two cases, the green target rotated clockwise and the red target rotated anti-clockwise. The order in which the participants were shown the targets was counterbalanced.

Following the video, participants were asked whether they had understood the functionality of the red and green targets when they first appeared, or at the end of the video. They were then asked which method of target rotation they preferred, i.e. same direction or opposite direction, and why.

3.5.3 Results

The video control prototype suffered the most incorrect activations (36), only three participants had no incorrect activations, resulting in a task success rate of 92% out of 425 activations, see Table 3.5. We observed four participants using more than one type of input to activate the controls, with the dominant hand being the most predominantly used type of movement accounting for 76% of all activations.

Ten participants reported that they understood the functionality of the video controls when they appeared, whereas seven did not understand the functionality until they activated the Orbits. Three participants did not understand the functionality of the controls, even after interacting with the prototype. The three participants who did not understand the functionality reported that they thought the different speed and sizes of Orbits were available to allow the user to choose whichever they prefer. Of the participants that understood the functionality, nine reported that they understood it because of the speed of the Orbits (i.e. the quicker the speed, the larger the effect) and six reported that it was because of the size (i.e. the bigger the size, the larger the effect). Two participants reported that the combination of speed and size led them to understand the functionality.

Table 3.5: Results for the third study, showing the types of movement used to activate the Orbits and the overall task success rate.

		Video Control	Information Popup
Number of activations	Head	30 (7%)	14 (10%)
	Dom. hand	325 (76%)	95 (70%)
	Non. hand	30 (7%)	5 (4%)
	Phone	36 (8%)	18 (13%)
	Cup	4 (1%)	2 (1%)
	Foot	0 (0%)	2 (1%)
Total Activations		425	136
Incorrect activations		36	0
False activations		0	0
Task success rate		92%	100%

Users achieved a task success rate of 100% out of 136 activations when interacting with the Information Popup prototype. The most frequently used type of movement was the dominant hand, accounting for 70% of all activations. Five users activated the Orbits with more than one type of input, including one participant who used their foot to activate an Orbit (a different participant to the one who used the foot in the second study).

Fifteen participants reported that they understood the concept of the green and red targets to open or close the information popup. The remaining five reported that they did not understand the concept until they had activated one of the targets. Sixteen participants preferred it when the red and green targets rotated in opposite ways, reporting that it required less effort to trigger the correct target because there was no chance of getting it wrong. Three participants preferred it when the red and green target rotated in the same direction because it was more aesthetically pleasing. One participant had no preference, reporting that it was more aesthetically pleasing for the same way but at the same time it was easy to select a target when they rotated in opposite directions.

3.5.4 Discussion

We have demonstrated that multiple input options can be expressed using multiple targets for the Orbits. This enables multiple targets to be located around a single Orbit which reduces the screen space required for the interface, and provides greater flexibility for designers.

It is interesting to note that the participants understanding of the video control functionality was predominantly due to either the size or the speed, but rarely both. This suggests that using a combination of size and speed is advantageous, because perception of the functionality of the different properties of the Orbits are not consistent across users. One participant noted that there could be an issue for the colour-blind with the red and green targets used in the information popup application, however they suggested simple tick and cross icons could also be used to convey the same information.

The ability to have multiple speeds of Orbits display simultaneously is desirable because it potentially allows for a greater number of Orbits to be displayed on the screen at any given moment. However, we noted that during the video control prototype a relatively large number of incorrect activations was a result of the users triggering a faster target that was in the process of overlapping a slower target, or vice versa. Whereas this might not be an ideal solution for increasing the number of controls, the task success rate remained above 90%.

3.6 Discussion

In this chapter, we have introduced and evaluated TraceMatch, a novel touchless interaction technique which abstracts from any specific body part. We have demonstrated that participants were able to select a target from eight Orbits (four in both directions) with an average task success rate of 88% across all sizes, speeds, and types of movement during a controlled experiment. The flexibility to successfully interact with a wide variety of different movement types is advantageous as we observed that participants had different preferences for their preferred type of movement. When users were provided the freedom to interact with TraceMatch in a naturalistic application context, users achieved a task success rate of >97% using standard Orbits with a variety of different movements, and >92% with Orbits using targets with different speeds and sizes displayed simultaneously.

We have studied TraceMatch in the context of Interactive TV, however the interaction approach can be used for a wide variety of devices, requiring only a webcam and a method for displaying the Orbits (not just limited to screens). We have shown that the size of the Orbits has no significant effect on task success rate, and that the colour, speed, and size of the targets of an Orbit can be used to convey additional information regarding its functionality for use with multi-level input. This affords designers the flexibility to design interfaces to constraints, and has wider implications, especially on smaller devices.

The intuitiveness and high discoverability of TraceMatch enables the technique to be extended into spontaneous interaction environments, such as public displays. Our studies have focussed on single users in the camera's field of view, however the generic nature of TraceMatch enables multi-user applications. TraceMatch's ability to abstract from a specific body part enables users to use any type of object. Specific applications, such as interactive games, could be developed for children in which their favourite toys, or objects relating to the game, could be used as input control. It also has the potential as an interaction technique for users for which conventional gestural input is not suitable, e.g. amputees.

Lastly, we identify four limitations in our studies. Movement correlation techniques, in general, are not ideally suited for continuous controls, e.g. changing volume, because they require the user to continuously follow the target for prolonged periods. For this reason, we opted for skipping forward and backwards rather than fast-forwarding and rewinding in the Video Control prototype. However, it is not intended that the TraceMatch control replaces existing input controls for television, i.e. the remote control, rather it compliments existing input methods by offering a method of low-effort gestural control for simple mundane tasks.

The second limitation is that we only consider circular motion. The extra stage of fitting a circle to the user's motion is required to reduce the chance of false activations as a result of detecting motion indiscriminately. TraceMatch can be extended to non-circular motion in the case of periodic Orbits by either replacing the circle fitting stage with the required shape or by using more sophisticated matching algorithms. Thirdly, the user performance results are inherently dependent on the algorithm chosen, and don't necessarily reflect participants' ability to follow the motion. Rather, they reflect the algorithms ability to detect the motion that users generate. Finally, the false positive dataset we collected represents the cases of watching TV or browsing the internet to gain indicative parameters for the system. However, there are many different types of motions users may perform which may result in an accidental activation, and these may vary depending on different contexts and or environments (e.g. large crowds). In section 5 we gather a more extensive false positive dataset consisting of semantic and spatial gestures, eliciting user movement more deliberately.

3.7 Conclusion

The aim of this chapter was to address whether movement could be used as the primary sensing principle for interaction (RQ1). We have demonstrated the feasibility of how movement can be used as the primary sensing principle, and how versatile motion correlation is, with TraceMatch (RQ1.1). TraceMatch demonstrates how we can abstract from specific body parts to provide users with greater flexibility to provide input by whatever means necessary. TraceMatch is a versatile technique that supports input by tracing of animated controls and requires only a single general purpose camera. It is able to detect motion as input that users can produce with little effort and in flexible ways – with their head, hand, or while holding an object. The technique lends itself to interaction with any form of device that is able to display controls in animated form.

Building upon this, we demonstrated how effective users are at providing input using different body parts, or when holding objects, through an experimental evaluation of how users interact with the TraceMatch system (RQ1.2). Our studies have shown that users are adept at providing input across body parts, and whilst holding objects, and has provided insights into how users may make spontaneous choices in the context of Smart TV applications (RQ1.3). Our results show that motion correlation with TraceMatch is optimal in terms of accuracy and selection time when selecting from up to 8 simultaneous objects. Users do not necessarily use their preferred type of input for interaction, as it will be based on context and in some cases the position of the Orbits on the screen. Across all our studies the main task was to interact with the system. One would expect the flexibility afforded by the system to be fully utilised when interaction is a secondary task. TraceMatch offers key advantages for low-effort interaction when performing mundane tasks, and has the ability to act as an input to a world of many devices. However, interaction extends beyond simple selection tasks which we have investigated in this chapter, and many tasks require more expressive control than is afforded by motion correlation in isolation. We explore how this can be addressed in the next chapter.

4

MatchPoint: Spontaneous Spatial Coupling Using Motion Correlation

If the principle of motion correlation is to be used as an input to a world of many devices, we must consider how it can be used to support more complex interactions beyond simple selection tasks. In the last chapter, we saw how motion correlation is limited in its ability to provide continuous input control. Changing a value, such as the playtime in Chapter 3.5, can be achieved with a user continuously following a target, and the system incrementing/decrementing the value whilst synchronisation is maintained. However, this is subject to under/overshooting the desired value, and any corrective action would require re-synchronisation with another moving target, making it less than ideal. To overcome this limitation, we investigate how motion correlation can be used to bootstrap spatial pointing (RQ2).

We introduce *spontaneous spatial coupling*, a hybrid technique of motion correlation and pointing that allows users to temporarily acquire a pointer, as illustrated in Figure 4.1. The interaction is initiated using controls which are presented to the user as moving targets that are differentiable by their movement. To activate a control the user matches its motion using any movement they can generate (e.g. a body part or an object they move). Upon synchronisation with a displayed target, a spatial coupling is created between the user's input and the control, with the established coordinate frame based on the preceding motion correlation phase. The spatial coupling is temporary for the purpose of a particular interaction.

Pointers can be acquired on-demand, using any type of movement captured by the input device,

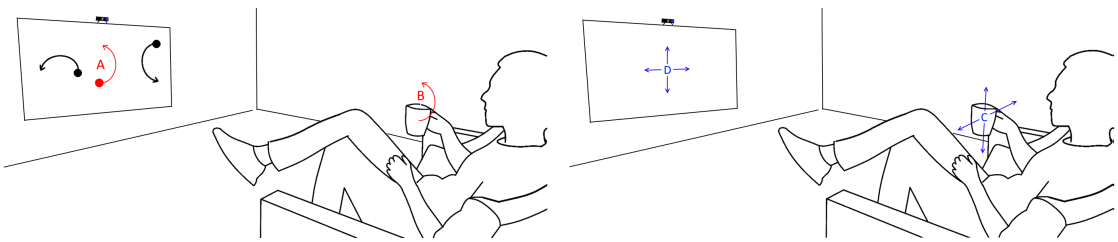

Figure 4.1: Spontaneous spatial coupling is a hybrid technique of motion-matching and pointing. Controls in the form of moving targets are presented to the user (A). When the user synchronises their movement with a target (B), a spatial coupling is created between the user's input (C) and the control (D). The technique enables ad hoc appropriation of any part of their body, or any object they hold, as a pointing device.

to be used for manipulation of individual controls or entire interfaces. Pointing has mostly been studied with applications that involve sustained interaction. In contrast, spontaneous spatial coupling is geared toward use in contexts where users interact on impulse, where interactions are short, or where interactions occur on the side of other activity. Such forms of interaction are becoming more typical as we engage with increasing numbers of devices in our environments, highlighting a need for users to have instant control "right here, right now" [134]. Spontaneous spatial coupling addresses this need with a low-effort method for instant, yet expressive control. Users can decouple from a pointer whenever they choose, providing the flexibility to change input in the case of fatigue, or for situational or contextual reasons. Everyday objects can be coupled to controls and left in place for prolonged periods, providing the opportunity to create unique tangible user interfaces. Multiple pointers can be instantiated, to support single users for bi-manual, multi-modal and multi-point interaction, and multiple users for simultaneous engagement and collaboration.

We operationalise this technique with *MatchPoint*, a computer vision-based implementation of spontaneous spatial coupling which builds upon TraceMatch. Matchpoint is highly deployable as it requires only a webcam as minimal hardware, and enables users to interact flexibly with minimal constraints on their pose and ways of providing input. The system is based on display of controls with orbiting targets, and accepts any form of movement in its field of view as matching input. We demonstrate how the initial motion correlation phase establishes a coordinate frame for spatial interaction, including the setting of the control-display gain (RQ2.1). MatchPoint can detect and track multiple pointing instances in parallel. Due to its generic approach to motion detection, the system does not require any calibration or training. As the system requires only a webcam, it can be deployed in many application domains, including on large displays, tablets, laptops, and smartphones.

We explore the unique design space created by spontaneous spatial coupling using practical examples built for MatchPoint (RQ2.2). We show how spontaneous spatial coupling can be used to create spontaneous pointing instances with any part of the body for interaction with the traditional cursor-based paradigm. We also show how multiple pointers can be created and decoupled using practical examples, either across users or with different inputs, and then used in parallel. These can be used to manipulate underlying controls with varying degrees of coupling, from unrelated pointers which can manipulate different objects, to tightly coupled pointers which allow for more complex interactions based on the spatial relationship between the two. Finally, we explore the unique affordances imbued when physical objects are used for the coupling process, resulting in ad-hoc tangible interfaces.

Finally, the notion of deriving the control-display gain from the initial motion correlation phase raises questions of how successfully users can point in the spatial stage (RQ2.3). We conduct an experiment based on the ISO 9241-9 multi-directional pointing task to investigate how successfully users can point across different body parts, or while holding an object, using MatchPoint. Previous literature suggests the hands are much more adept at pointing than the head, however we are also interested in the ability to point with objects-in-hand, to account for the flexibility afforded by accepting generic motion as input. We reflect on our results using previously reported metrics for the same ISO test, which illustrate that pointing at a distance with MatchPoint is as accurate as similar computer-vision based techniques.

Our aim in this work is to define spontaneous spatial coupling as a new interaction technique and to explore the opportunities it affords. We advance theoretical and practical understanding of the technique through the following contributions:

- Definition of the properties that define spontaneous spatial coupling.
- MatchPoint, a webcam-based implementation of spontaneous spatial coupling which uses generic computer vision processing to accept any form of input.

- An exploration of the design space with practical examples, built for MatchPoint, which demonstrate the versatility of spontaneous spatial coupling for different interaction techniques.
- Evaluation of MatchPoint as an input device for pointing using the ISO 9241-9 multi-directional pointing task.

4.1 Spontaneous Spatial Coupling

The interaction principle behind spontaneous spatial coupling is defined by five properties:

1. Distinct motions displayed to the user represent controls available for interaction;
2. A user's intent to interact with a control is expressed through movement corresponding to the control's motion;
3. The selection of a control is determined by the correlation between the system's output and the user's input;
4. Upon selection a spatial coupling is created between the user and the underlying functionality of the control;
5. The user is able to decouple from the control at will.

The interaction involves a phase of motion-matching followed by pointer control. The matching phase can be based on any shape of motion, and method for determining a correlation. Considerations for the design include how distinctive the motion is (to accidental matches), how easily and efficiently users can match it, and how reliably and robustly it can be detected.

Any type of input that produces motion can be used for both the matching phase and subsequent input, contrasting existing systems that are optimised for specific modalities such as tracking of hand gestures [200], head pose [252], or feet [267]. Prior work has also demonstrated spontaneous coupling of smartphones for pointing on public displays [10, 30], contrasting our work where users do not require any device. The matching phase results in a specific spatial coupling between the user and the control's underlying functionality. The coupling can be interpreted as a pointing device for which a cursor is instantiated, or as a device for describing gestural input. The output device needs to be able to present motion to facilitate the coupling, but once the user is coupled other types of feedback can be used for interaction (e.g. audio feedback when controlling the volume of a radio).

In the following, we discuss system design considerations to take into account when designing controls for spontaneous spatial coupling.

Control-Display Gain: The CD gain of the pointer can be set according to the size of the user's movement in the motion-matching phase. This approach assumes the user's movement range during the motion-matching phase is indicative of the movement range used when spatially coupled with a control. This may not always hold true and one might want to define the CD gain to increase the allowed range of movements, for example when manipulating objects on very-large displays.

Transfer Function: Absolute control maps provide a fixed gain so that the user's movement is directly mapped to the on-screen controls. To allow for greater precision, relative control maps, such as pointer-acceleration-based transfer functions, can be used in conjunction with techniques such as semantic pointing [27]. A drawback of relative control maps is the need for the input device to provide the ability to clutch, temporarily disabling the gesture, in order to allow the user to reposition themselves.

Pointer Starting Position: The user is in motion at the point of synchronisation with a moving target. If the system switched immediately into pointing mode, the pointer would move in

Figure 4.2: Initialisation of the MatchPoint tracker once a user is in synchrony with an Orbit. Left: One matched feature point showing its trajectory (green) with fitted circle (red and blue). Centre: The result after connected-component labelling of candidate pixels that matched the motion of the feature point (green), calculated using dense optical flow. Right: The region of interest of the object to be tracked (green).

continuation of the motion described for matching. For tasks such as parameter control or spatial selection, the user may wish to start the interaction from a well-defined position (e.g. the option currently selected). To allow this, the system can indicate when the match is detected, wait for the user to stop their matching motion, and enable pointing only after the matched input has become stationary. This may not always be required, for example when the pointer is used to represent an on-screen cursor using an absolute control mapping.

Pointer Termination: A range of mechanisms are possible for decoupling from a control. A pointer could terminate after completion of a single task for which it was instantiated, or after a pre-set time of inactivity. It is also possible to have the user explicitly trigger pointer termination, for instance using dwell or goal-crossing techniques [2]. If only one of the axes is used for input, users can use the other to signal task completion. If the coupling is used for gestural input, a specific gesture can be included for decoupling. Other than such generic techniques, specific implementations might afford device dependent decoupling mechanisms (e.g. using the depth axis of a depth sensor).

System Visibility: Moving targets are displayed to the user for acquisition and selection. The dynamic nature of the controls may be visibly distracting if the user is focussing on the display and has no intention of interacting with the system for prolonged periods of time. To overcome this the moving targets can be hidden from the user by assigning a specific control to hide the targets, or after a period of time with no input from the user. To display the moving targets, generic gestures can be used, e.g. moving an input in a full circle. There may be no need to hide the moving targets for applications where the user's main focus is not on the display.

4.2 MatchPoint

MatchPoint is a webcam-based implementation of spontaneous spatial coupling that accepts any type of movement as input. The system consists of two main processing pipelines (see Figure 4.3): the motion matching pipeline which allows the user to select a control and acquire a pointer; and the tracking pipeline which allows the user to manipulate the pointer by providing a spatial coupling between the user's input and the control. Instances of the tracking pipeline can run in parallel with the motion-matching pipeline to allow a single user to acquire multiple pointers, or for multiple users to acquire a pointer.

4.2.1 Motion-Matching

For motion-matching we use the TraceMatch processing pipeline, introduced in the previous chapter. TraceMatch uses *Orbits*, introduced by Esteves *et al.* [69], as input controls which consist of an orbiting target around a circular widget, a motion that is not likely to be reproduced

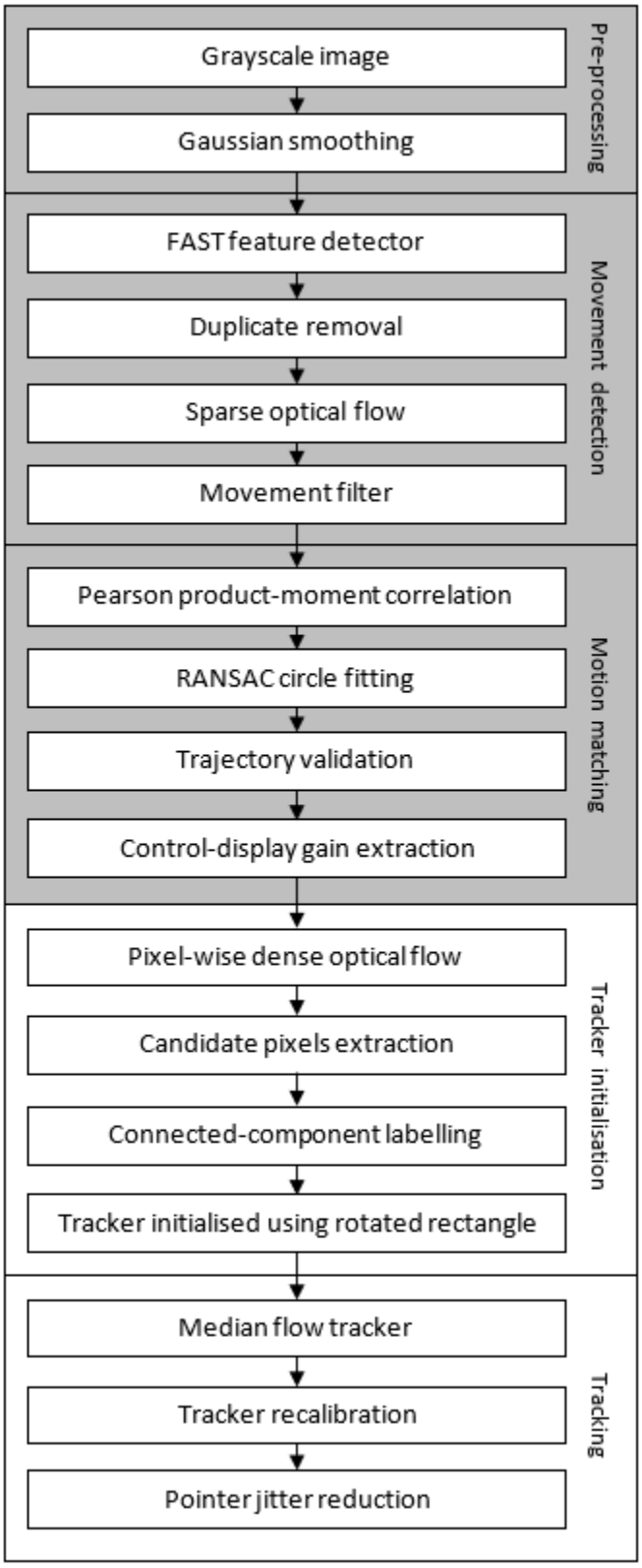

Figure 4.3: Processing pipeline of the MatchPoint system, which builds upon the TraceMatch system (in grey).

accidentally by the user. FAST feature points [224] are detected and tracked using the pyramidal Lucas-Kanade optical flow algorithm [152, 31]. Each feature point that exhibits a minimum amount of movement is compared to all of the Orbis currently displayed to the user. A match is confirmed if the minimum Pearson correlation coefficient between either the x or y axis of a feature point and an Orbit is above a minimum threshold, and if the feature's trajectory can be successfully fitted to a circle with the appropriate arc length using RANSAC.

Control-Display Gain

During the matching phase we also calculate the control-display (CD) gain. The output of the matching process returns a fitted circle for each matched feature point which indicates the ideal trajectory of the user's movement when matching the control (i.e. without any noise). We use this as an indication of the range of movement for which the user is comfortable using, as it takes into account the input used and distance from the camera, e.g. a head movement may result in a much smaller radius than a hand movement. The CD gain, CD_{gain}, is calculated by taking the reciprocal of the average radii, r, of the fitted circles from the matched feature points:

$$CD_{gain} = \left(\sum_{i=1}^{N} r^i / N \right)^{-1} \tag{4.1}$$

The CD gain is then multiplied by the appropriate distance depending on the context. Three ways which to calculate the appropriate multiplication factor are the size of the screen, size of the widget presented for the motion correlation phase, or size of the spatial control. Using the size of the screen ensures that the whole screen can be reached within a comfortable range, for example when using a screen cursor. Alternatively, one could use the size of the motion presented on the screen to calculate the gain, so that it forms a 1:1 mapping from the user's perspective. Using the size of the spatial control element (e.g. sliders, carousels) can be used to maximise the accuracy of the precision of the user's movement whilst in a comfortable range.

4.2.2 Spatial Coupling

To provide the spatial coupling between the user and the control we first initialise a region of interest (ROI) in the frame relating to the input, before tracking it in subsequent frames using a modified version of the Median Flow tracker [124]. We further process the output to remove pointer jitter introduced by image noise.

Tracker Initialisation

The output of the matching process is one or more feature points, however we ideally want to track as much of the body part, or object, which activated the control in order to improve downstream tracking performance. To do this we calculate the dense optical flow of the scene and compare the matched feature points' trajectories with the dense optical flow information, see Figure 4.2. First, we calculate the trajectories for all n matched feature points, $A = \{\mathbf{a}_1, \ldots, \mathbf{a}_n\}$, between frames t and $t-1$ using their respective x and y coordinates, where the match occurred at frame t:

$$\mathbf{a}_i = \left(x_i^{t-1} - x_i^t, y_i^{t-1} - y_i^t \right) \text{ for } i = 1, \ldots, n \tag{4.2}$$

We then calculate the Euclidean distance of each trajectory, $D = \{\|\mathbf{a}_1\|, \ldots, \|\mathbf{a}_n\|\}$, and the average angle of the trajectories, $\bar{\theta}$, using the trajectories' average unit vector, $\bar{\mathbf{a}}$:

$$\bar{\mathbf{a}} = \frac{1}{n}\sum_{i=1}^{n}\hat{\mathbf{a}}_i \tag{4.3}$$

Where $\bar{\theta} = \angle\bar{\mathbf{a}}$. Using the Farnebäck algorithm [72] we calculate the dense optical flow of the scene, $F = \{\mathbf{f}_1, \ldots, \mathbf{f}_N\}$, between frames t and $t-1$ for all N pixels, $P = \{\mathbf{p}_1, \ldots, \mathbf{p}_N\}$ where:

$$\mathbf{p}_i^t = (x_i, y_i) \text{ for } i = 1, \ldots, N \tag{4.4}$$

$$\mathbf{p}_i^{t-1} = (x_i + \Delta x_i, y_i + \Delta y_i) \text{ for } i = 1, \ldots, N \tag{4.5}$$

$$\mathbf{f}_i = (\Delta x_i, \Delta y_i) \text{ for } i = 1, \ldots, N \tag{4.6}$$

We then identify pixels that matched the motion of the matched feature point(s) by assessing whether the ith pixel satisfies the following equations:

$$(1 - th_d)\min\{D\} < \|\mathbf{f}_i\| < (1 + th_d)\max\{D\} \tag{4.7}$$

$$\bar{\theta} - th_\theta < \angle\mathbf{f}_i < \bar{\theta} + th_\theta \tag{4.8}$$

Where $\|\mathbf{f}_i\|$ is the magnitude of the dense flow at pixel i, $\angle\mathbf{f}_i$ is the angle of the dense flow at pixel i, th_d is a distance threshold between 0 and 1, and th_θ is an angle threshold measured in radians. For this implementation we use values of 0.25 and $\pi/8$ for th_d and th_θ respectively. This produces a mask of candidate pixels which is further processed to remove candidates resulting from image noise or background movement.

Finally, we use connected-component labelling to find candidate groups from the candidate pixels. We seed the connected-component labelling with $10px \times 10px$ areas around the matched feature points because errors can be introduced in the tracking of individual feature points during the motion-matching phase, i.e. a feature point is offset from the body part/object. We then fit a rotated rectangle around all the candidate groups connected to the matched feature point(s), resulting in the initial ROI for the tracker which will be tracked in subsequent frames. In the event there are no candidate groups, we initialise a minimum rectangular ROI $(30px \times 30px)$ around the matched feature points.

Tracking

The body part or object to be tracked may not exhibit smooth movements. The tracking should cope with unpredictable movements and changes in perspective of the body part/object relative to the camera. We experimented using different trackers, including Median Flow [124], KCF [52], MIL [9], TLD [125], and OLB [90]. Preliminary testing indicated that none were suitable for this application so we instead use a modified version of Median Flow to track a rotated rectangle.

The first modification we made was the selection of points to track at each iteration using Median Flow. Kalal *et al.* recommend using a grid of equidistant points or the use of a feature detector, such as FAST [124]. However, the body part or object that we wish to track may not fill the whole of the ROI, so instead we use a grid spacing based on central polygonal numbers (aka. the Lazy Caterer's sequence) to reduce the chance of the tracker getting stuck on background objects.

The second modification was to introduce a "recalibration" phase for the tracker, which accounts for changes in the size and perspective of the body part/object that is being tracked, whilst ensuring that the ROI covers as little as the background as possible. For recalibration we follow the same steps for initialising the ROI, however instead of using the matched feature point's

trajectory in Eq. 1 and 2 we use the trajectory of the ROI calculated using its centre, and instead of the matched feature points acting as the seeds to the connected-component labelling we use the centre of the ROI. As we have knowledge of the previous ROI we focus the search to a rectangular area around the centre of the ROI prior to calibration with width, $2 \times W$, and height, $2 \times H$, where W and H are the width and height of the ROI prior to recalibration. We remove the scaling of the ROI provided by the original Median Flow algorithm as this is performed during the recalibration phase.

The recalibration phase relies on the movement of the ROI, therefore we only perform the tracker recalibration when the magnitude of the ROI's trajectory is above a threshold, th_{ROI}, and at most every $t_{recalib}$ seconds to limit processing time. For our implementation we set th_{ROI} to 2 pixels, and $t_{recalib}$ to 500ms. The centre of the ROI is used as the reference point of the tracker (i.e. the point which updates the on-screen control). When we recalibrate the ROI, the centre may change which would cause a jump in the cursor from the perspective of the user. To avoid this we record the offset caused by the recalibration of the ROI and apply this to the output in subsequent frames so that the recalibration is unnoticeable to the user.

Reducing Pointer Jitter

Cameras are subject to image noise that affect tracker performance and result in unwanted movement of the pointer at the interface. This may be exacerbated by environmental factors, such as lighting, and the distance and input method of the user. For example, for smaller movements (e.g. head movements), there will be a lower signal-to-noise ratio and the jitter may be more apparent. Likewise, larger distances from the sensing device will reduce the signal-to-noise ratio. To reduce the effects of noise we take the average position of the ROI using a dynamic moving window when the Euclidean distance between the centre of the tracker from the previous frame to the current frame, d_t, is less than d_{MIN} pixels. The size of the moving window, N_B is calculated as:

$$N_B = N_{MAX} - \lfloor \frac{d_t \times N_{MAX}}{d_{MIN}} \rfloor \tag{4.9}$$

Where N_{MAX} is the maximum size of the moving window in frames, MatchPoint uses 10 frames. The moving window introduces input lag, therefore the value of N_{MAX} should be carefully considered based upon the frame rate of the camera. The value of d_{MIN} will vary depending on the quality of the camera, for our implementation this is set to 2.5 pixels using a Logitech C920 webcam.

4.3 Design Space

In the following we explore interaction techniques and applications that can be supported with spontaneous spatial coupling. We consider five cases:

- Single pointer → Multiple functionality – one pointer provides all the functionality in the interface;
- Multiple pointers → Multiple functionality – each pointer provides a different functionality;
- Multiple pointers → Single functionality – multiple pointers provide the same functionality;
- Parallel pointers – multiple pointers used in parallel;
- Tangible interfaces – creating temporary tangible interfaces using everyday objects.

The first case is the conventional case of “pointing as we know it". For all other cases, we present novel techniques and application demonstrators implemented with MatchPoint. All of the examples are applicable to both single and multiple users, as controls can be matched and spatially coupled in a non-exclusive manner.

4.3.1 Single Pointer → Multiple Functionality

Conventionally, touchless pointers use one pointer to control many on-screen input controls. This technique can be used with spontaneous spatial coupling, allowing users to decide how to provide input, and in the event of fatigue to desist pointing and resume with another type of input. If only one pointer is required, it is also possible to replace the motion-matching stage with a generic motion gesture, as there is no need to differentiate between different controls.

4.3.2 Multiple Pointers → Multiple Functionality

Multiple pointers can be used to provide different functionality. Users can acquire different pointers depending on the interaction to be performed, removing the need for a user to navigate through an interface using a single pointer. This also allows the CD gain to be mapped according to the size of a control, as opposed to size of the display. To demonstrate this concept, we developed two applications for different scenarios: a TV remote control for the living room, and a video player for use when engaged in other physical activity (e.g., in the kitchen).

TV Remote Control

Conventionally, a TV remote control is shared, and it must be passed from person to person. Gesture control for TV has been investigated in prior research, driven by users' desire to have instant control [134], but this has primarily focussed on library-based gestural techniques [45, 111, 118]. The level of interaction with TVs is increasing with the integration of “smart” features, while users primarily focus on the content displayed, or may watch it in the background whilst performing other tasks with minimal interaction. This provides an exemplar application space for MatchPoint.

The TV remote control application features controls for changing the channel up and down, muting the volume, changing the volume, selecting a channel from a list, and showing a TV guide (Fig. 4.4). Changing the channel up and down, and muting the volume, are binary choices and do not require spatial coupling, therefore these are implemented as motion-matching only controls.

Upon selecting the volume control, a one-directional slider is presented to the user (Fig. 4.5, left). The control waits until the user's input is stationary prior to displaying the control, allowing the user to change the volume relative to the current volume of the TV. Movement in the y-direction sets the volume level, and movement in the x-direction either cancels the control (when user moves left), or confirms the new volume level (when user moves right).

The channel selection control displays a one-directional carousel control (Fig. 4.5, right). This also waits for the user's input to be stationary prior to activating to ensure the user starts searching the channels from their currently selected channel. Movement in the x-direction scrolls through the channels, dwelling on a channel displays the programme details, and movement in the y-direction either cancels the control (down) or changes the channel to the current selection (up).

For the TV guide the user is presented with a cursor for navigation (Fig. 4.6). Dwelling on a programme displays the details in the upper-most box, which can be selected to present the user

Figure 4.4: TV remote control prototype, showing Orbits for: channel up and down (left), TV guide and channel select (middle), and mute toggle and volume control (right).

with a number of options depending on the programme, including watch now, set a reminder, start a recording, or close the TV guide. The user can also close the guide by dwelling at the edge of the interface.

Video Player

Building upon the TV context we developed a video player application. Video guides for practical tasks, such as cooking or car maintenance, are popular on platforms such as YouTube. Users will often watch the videos in-situ on a portable device, such as a laptop, tablet or mobile phone, whilst performing a task. Interactions with the video guide will be relatively sparse, such as pausing and rewinding, as the user is primarily focussed on performing the actions demonstrated in the video guide. Spontaneous spatial coupling allows the user to perform other tasks whilst interacting with the video player, e.g. using cooking utensils in the kitchen whilst cooking.

The video player features controls that allows the user to play or pause, change the playback, navigate to a specific time, and mute or change the volume (Fig. 4.7). The controls for play/pause and muting the volume are binary choices, therefore we use motion-matching only controls.

The video playback control allows the user to trigger playback of the video through movement in the x-direction, while movement in the y-direction exits the control (Fig. 4.8). Upon activa-

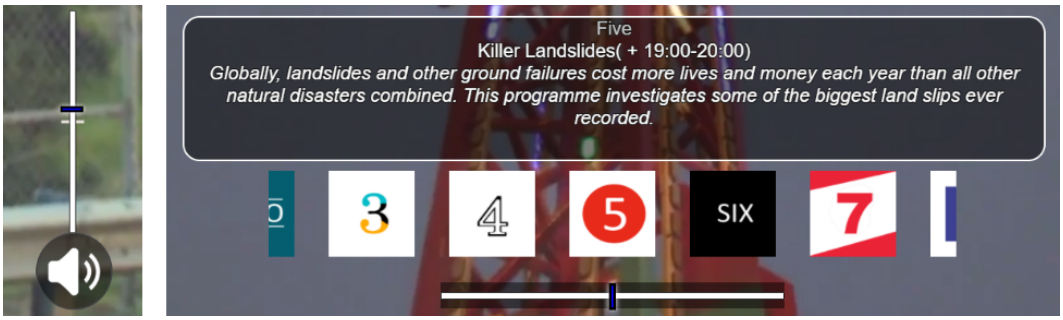

Figure 4.5: Volume slider which can be controlled with movements in the y-axis (left), and channel selection control showing the programme details and slider to indicate the position of the user's movements in the x-axis which controls the carousel (right).

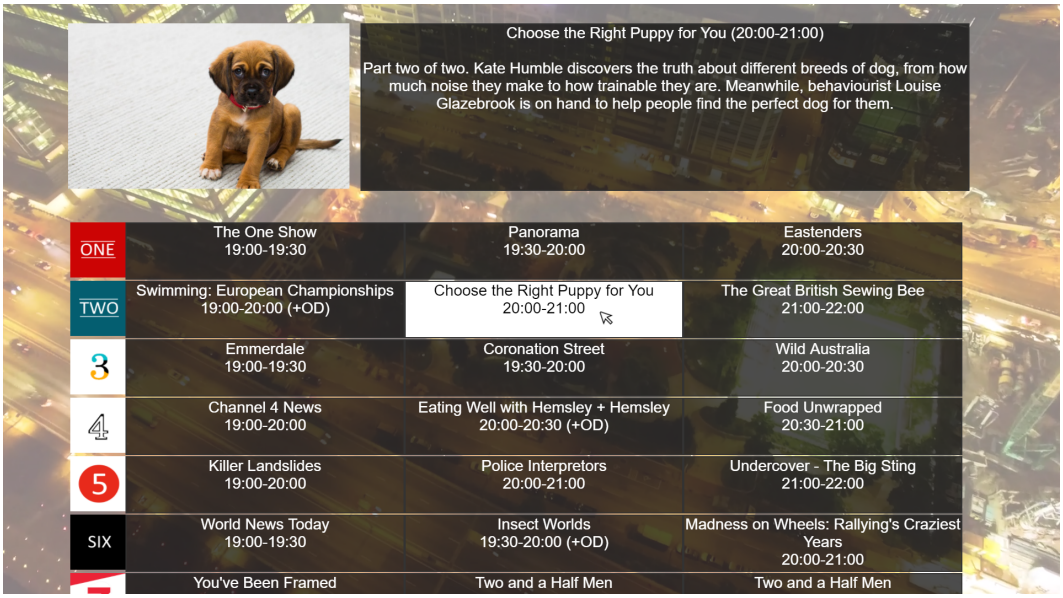

Figure 4.6: TV remote control prototype, showing the TV guide.

tion the control waits for the user's input to become stationary, ensuring that the play button is selected when the interaction starts.

The control allowing users to navigate to a specific time in the video uses movement in the x-direction to select the time, and movement in the y-direction to either confirm the selection (by moving up) or cancel the control (by moving down). Upon activation the user must pause their motion briefly so that search can resume from the current time in the video.

4.3.3 Multiple Pointers → Single Functionality

In the above examples multiple users can interact with the controls, but only one can interact with a specific control at any given time (e.g. two people can't change the volume at the same time). In some instances, it may be desirable for multiple users to acquire a pointer with the same functionality, for example to contextualize their discussions [191].

Whiteboard Pointing Prototype

A scenario in which this would be beneficial is a meeting room where users are remote from the display but would like to indicate a position on the screen. For this we designed a "whiteboard" pointing prototype featuring a control to allow users to acquire a cursor (Figs. 4.9 and 4.10). Upon selection, users acquire a cursor with a unique colour to allow multiple people to control a pointer and provide input to a shared space as and when required.

4.3.4 Parallel Pointers

Users can acquire multiple pointers at the same time. The functionality of the pointers used in parallel may be:

- Unrelated – control of one pointer does not affect the other(s)
- Loosely coupled – the pointers affect the same object, but can be used individually
- Tightly coupled – interaction results from the relationship between the pointers

Figure 4.7: Video control prototype showing Orbits for: play/pause (left), video progress bar and playback (middle), and volume control (right).

Figure 4.8: Playback control for the video control prototype where movements in the x-direction select different playback options.

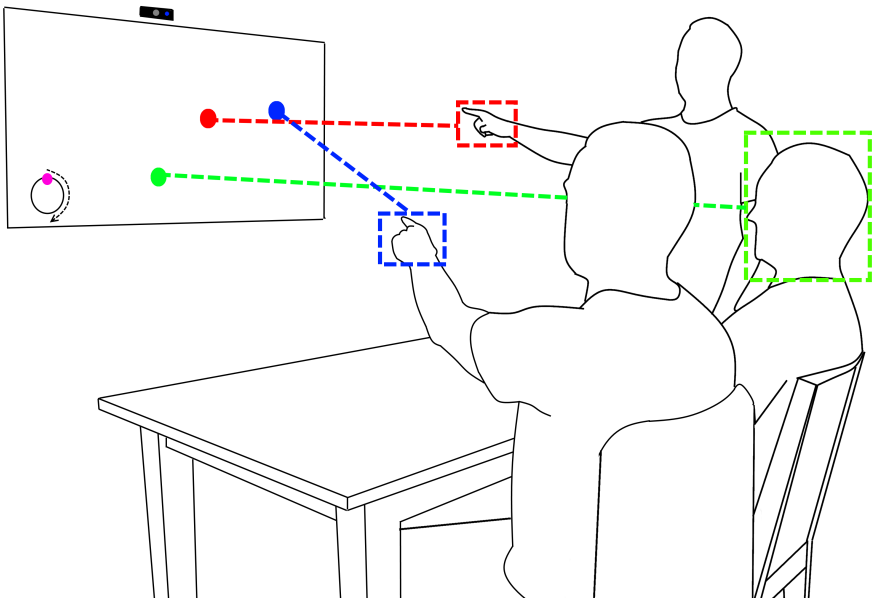

Figure 4.9: White board pointing prototype demonstrating multiple cursors (green, red, and blue). Dotted lines and rectangles represent the input and user controlling the cursor. The Orbit (pink) shows the colour of the next cursor to be generated.

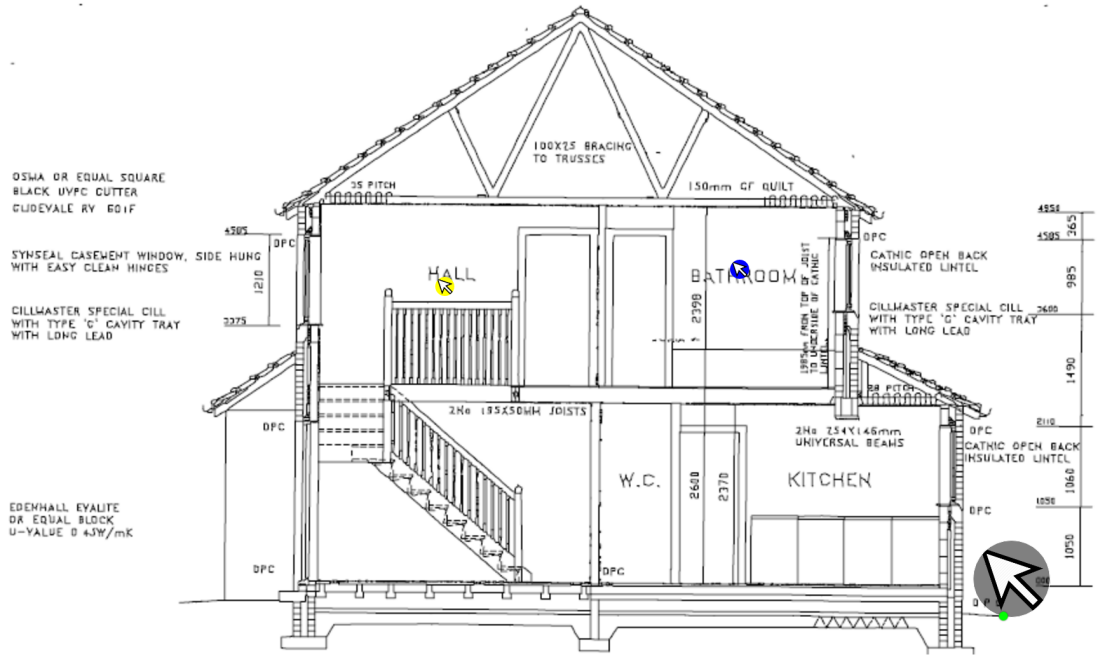

Figure 4.10: Interface of the white board pointing prototype demonstrating multiple cursors (yellow and blue) which allow users to highlight different parts of the display. The Orbit (green) shows the colour of the next cursor to be generated.

For unrelated and loosely coupled pointers, parallel usage allows users to complete tasks in less time, or to manipulate one control based on the state of another. Tightly coupled pointers offer more complex interactions by using the pointers' spatial relationship with each other, however they must be used at the same time and can not be used individually.

Object Manipulation Prototype

To demonstrate loosely and tightly coupled parallel pointing, we created two prototypes for object manipulation (Fig. 4.11). The first is a multi-modal prototype designed to be used with any type of input. It consists of two loosely coupled input controls: one to pan the object, and the other to zoom. The pan control uses the x and y directions of the user's first input to position the object. The zoom control uses movement in the y-direction of the user's second input to zoom in or out, using an acceleration-based transfer function to control the level of zoom. Movement in the x-direction is used to enable clutching (by moving to the left) or exits the control (by moving to the right).

The second prototype demonstrates bi-manual input, designed specifically for the hands using two tightly-coupled controls. Object manipulation is supported in a way that is familiar from touch-screens: the distance between the hands determines the zoom, the centre point between the hands determines the pan position, and the positions of the hands relative to each other determines the rotation.

The bi-manual prototype requires both controls to be activated in order to enable the parallel pointing interaction. However, it provides the user with three functions (pan, zoom and rotate) using only two pointers, contrasting the multi-modal approach which only provides two functions (pan and zoom). Note that the system does not identify body parts, and that it would be possible for users to acquire the controls with other than the intended hands. To avoid this, an interface should convey to the user if a specific type of input is required (e.g. showing icons of the hands).

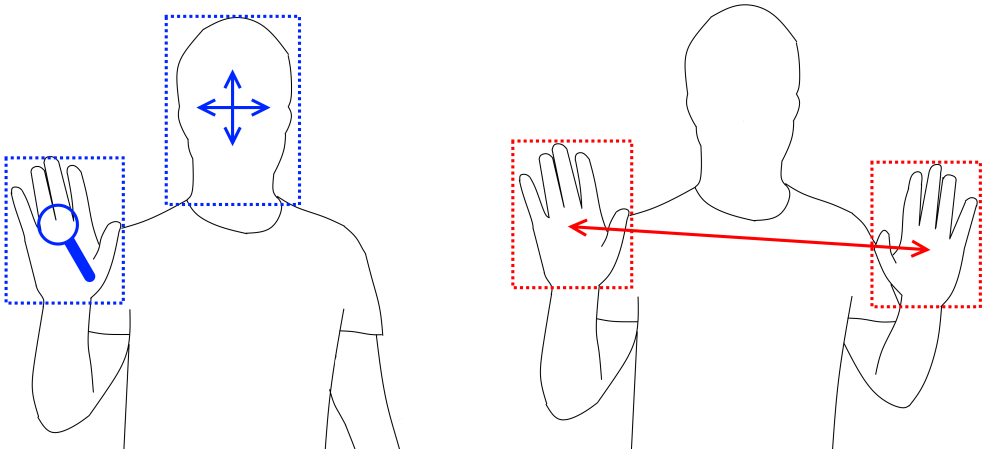

Figure 4.11: Two prototypes for parallel pointing. Multi-modal pan and zoom (left): one input controls the pan (the head), the other controls the zoom (the hand). Bi-manual pointing (right): the centre point between the hands determines the pan position, the distance determines the zoom, and the angle determines the rotation.

(a) **(b)**

Figure 4.12: Interfaces designed to demonstrate the parallel pointing manipulation of an image: (a) a multi-modal technique whereby each orbit corresponds to a specific command for loosely coupled input, and (b) a bi-manual technique whereby the orbits are used for tightly coupled control.

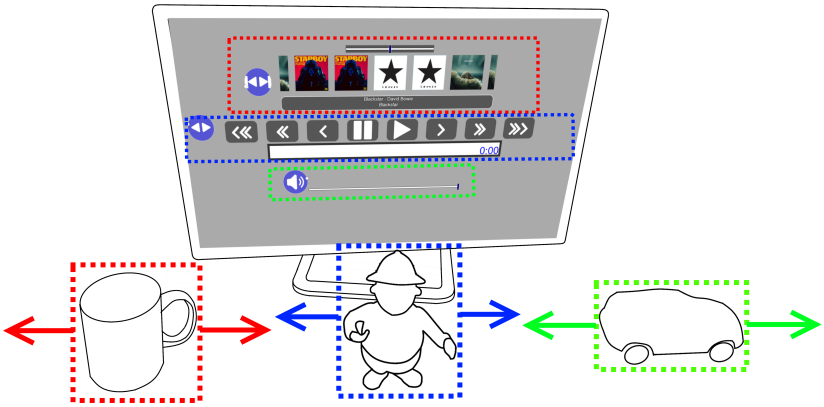

Figure 4.13: A tangible interface created with MatchPoint. The cup controls the playlist, the toy figure controls the playback, and the toy car controls the volume. Moving the objects left and right changes the value of the respective control.

4.3.5 Tangible Interfaces

Other than creating a coupling with a part of their body, users can also move a tangible object in synchrony with a displayed motion. The result is a spatial coupling between tangible object and control. The use of objects in this way presents a distinct case, as they become tangible intermediaries between user and control. This has interesting affordances, as a user can leave an object stationary in between interactions, with a persistent coupling. Objects can also afford specific manipulations depending on their shape and weight, for example nudging, rolling and tilting.

Graspable Music Player

To demonstrate this concept we developed a tangible music player interface (Fig. 4.13). Three controls are displayed to the user allowing them to change the playlist, volume, or the playback. Once the user couples a physical object to a control the system waits for the user to position the object into its starting position. The control is not fully activated until the object remains stationary for an extended period of time (e.g. 4 seconds). All input controls in this example utilise movement in the x-axis to manipulate the respective control, e.g. moving the toy car left lowers the volume, moving it right increases the volume. The objects remain paired with the controls until they are removed from the camera's field of view.

4.3.6 Summary

We have demonstrated different configurations of spontaneous spatial coupling, with application prototypes grounded in real-world applications that showcase multiple pointers being used in parallel, multiple users concurrently pointing, and the unique affordances of using tangible objects for spontaneous spatial coupling. The exploration of interaction techniques shows how spontaneous spatial coupling allows the user to define an interaction space on a per-interaction basis. The use of motion correlation to define the CD gain presents interesting design choices, and in the prototypes presented we based the CD gain on individual controls so that users could interact within a comfortable range of movement. This was motivated by the use of the head as input during development which, in contrast to the hands, has limited range of movement. By limiting input in the spatial domain to that used in the motion correlation phase, we ensure users are within a comfortable range. Depending on the context, designers may decide to adapt the CD gain to the screen or as a 1:1 mapping as discussed previously. Whereas the presented

techniques and prototypes demonstrate what is possible with spontaneous spatial coupling, the use of dynamic CD gain motivates an exploration of how it affects user input.

4.4 Multi-Directional Pointing Task Evaluation

The ability to dynamically adjust the control-display gain of the pointing phase based on the size of the movement during the motion correlation phase enables seamless pointing between inputs without the system requiring any knowledge of the body parts providing input. In order to assess how successfully users adapt to the CD gain being dynamically set based on the motion correlation phase we conducted a two-directional Fitt's Law study, based on the ISO 9241-9 standard, to understand the performance of MatchPoint as an input device for pointing and to provide insights into pointing performance with different body parts, and whilst holding an object. The CD gain is calculated using the technique proposed in section 4.2.1, with the size of the Fitts' circle being used as the range of movement. We compare three movement conditions (head, hand and cup-in-hand) to investigate their throughput and other pointing characteristics in a simulated living room environment. We chose a simulated living room environment to test how the system performed when the user was at a larger distance (>2m) from the input device.

4.4.1 Task

Ten circular targets of diameter W were displayed in a circular configuration with a radius of $A/2$. In order to avoid possible confounds we used Guiard's Form x Scale design [95], with three levels of W (50px, 100px, 200px) and one level of A (900px), resulting in three unique index of difficulty (ID) values of 4.24, 3.32, and 2.46 respectively. When calculating the throughput we use the effective index of difficulty, ID_e, by measuring the effective values of A and W, which take into consideration the speed/accuracy trade-offs participants make when completing the task:

$$ID_e = \log_2(A_e/W_e + 1) \tag{4.10}$$

Where A_e is the average movement distance observed [249], and W_e is the standard deviation of endpoints, defined as:

$$W_e = 4.133 \times SD_{x,y} \tag{4.11}$$

Where, $SD_{x,y}$ is the bivariate endpoint deviation, defined as [295]:

$$SD_{x,y} = \sqrt{\frac{\sum_{i=1}^{N}\left(\sqrt{(x_i-\bar{x})^2+(y_i-\bar{y})^2}\right)^2}{N-1}} \tag{4.12}$$

Dwell was used as the selection process, with a dwell time of 240ms to simulate the time taken to press a button [227]. We also measure the number of target re-entries, as defined by MacKenzie *et al.* [156].

4.4.2 Participants and Apparatus

We recruited 12 participants to undertake the study (mean = 28.1, SD = 3.9). Six participants were female, and one was left-handed. None of the participants had used the MatchPoint system prior to the study.

The study was conducted in a simulated living room environment, using a Samsung 55" Smart TV (1920 x 1080) as the display. An unmodified off-the shelf webcam was used as the input

device to the MatchPoint system and captured a 640 x 480 region of interest from a 1920 x 1080 frame. The region of interest was used to ensure that only movement relating to the study was captured by the webcam. Participants were seated on a couch 2.23m from the TV (based on a TV size to viewing distance calculator). For the cup-in-hand input participants were asked to hold a cup half-filled with water to simulate holding a drink.

4.4.3 Procedure

At the beginning of the study participants completed a demographics questionnaire and were introduced to the MatchPoint system. They were instructed to relax and work comfortably whilst performing the tasks as quickly and as accurately as possible. No instructions were given regarding how to hold the cup or position the hand when matching and pointing.

For each movement condition, participants undertook three sets (one for each ID), where a set consisted of five blocks of 10 target selections. The first block was used as a warm-up, with the remaining four being used for data collection. At the start of each block the users acquired a cursor by matching the motion of an Orbit with the specified body part or object. Participants were instructed to let the researcher know if they required a break in between blocks due to fatigue. A Latin square design was used to counterbalance for movement conditions, and the order in which the IDs were presented and the starting position of the first target were randomised. The movement time was only measured after the first target was selected. The study ended with a brief verbal discussion to gain feedback on the system.

Excluding the warm-up blocks the study involved 12 participants $\times$ 3 movement conditions $\times$ 3 IDs $\times$ 4 blocks $\times$ 10 trials per block $=$ 4320 trials.

4.4.4 Results

Movement times, shown in Figure 4.14, were analysed using a one-way repeated measures ANOVA to determine whether any statistically significant differences existed for the different movement conditions. The data was normally distributed, as assessed by boxplot and Shapiro-Wilk test, $p > .05$, and the assumption of sphericity was not violated, as assessed by Mauchly's test of sphericity, $\chi^2(2) = 3.903, p = .142$. The test revealed a significant difference between movement times ($F_{2,22} = 73.166, p < .001$). Post-hoc pairwise comparisons with Bonferroni corrections revealed the movement time of the head (3.12s) was significantly higher compared to both the hand (2.37s) and cup (2.54s), at $p < .001$. There was also a significant difference between the movement time of the hand and cup, at $p = .035$.

The grand throughputs for each movement condition were calculated using the mean of means approach [249]. A one-way repeated measures ANOVA was used to determine whether there were statistically significant differences between throughputs for different movement conditions. The data was normally distributed, as assessed by boxplot and Shapiro-Wilk test, $p > .05$, and the assumption of sphericity was not violated, as assessed by Mauchly's test of sphericity, $\chi^2(2) = 1.856, p = .395$. A significant difference between movement conditions was revealed ($F_{2,22} = 54.617, p < .001$). Post-hoc pairwise comparisons with Bonferroni corrections showed that the throughput of the head (1.03 bits/s) was significantly lower compared with both the hand (1.35 bits/s) and cup (1.25 bits/s), at $p < .001$. There was not a statistically significant difference between the throughputs of the hand and the cup, at $p = .059$. Using linear regression, we then developed Fitts' Law models of the form:

$$MT_{input} = a + b \cdot ID_e \tag{4.13}$$

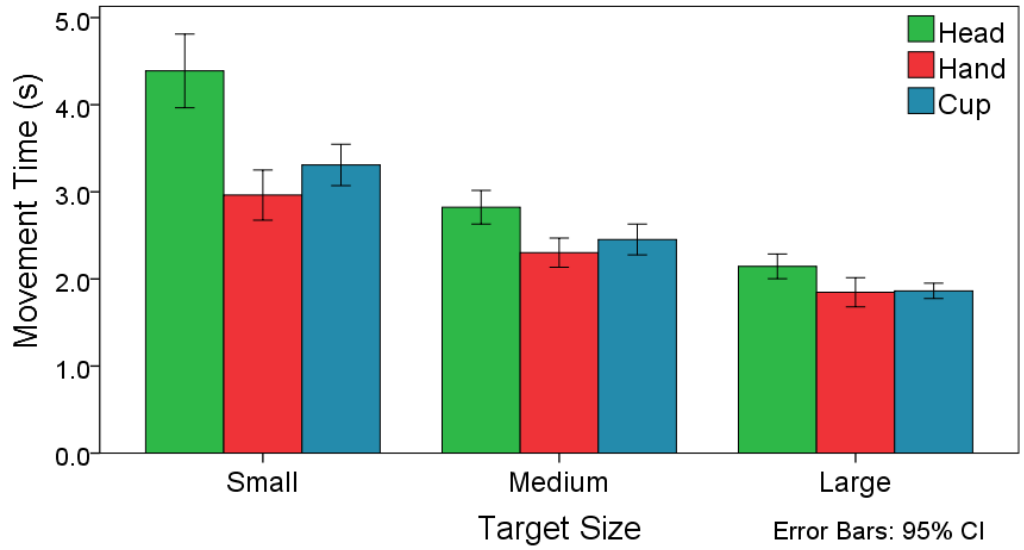

Figure 4.14: Movement times for each target size and movement condition.

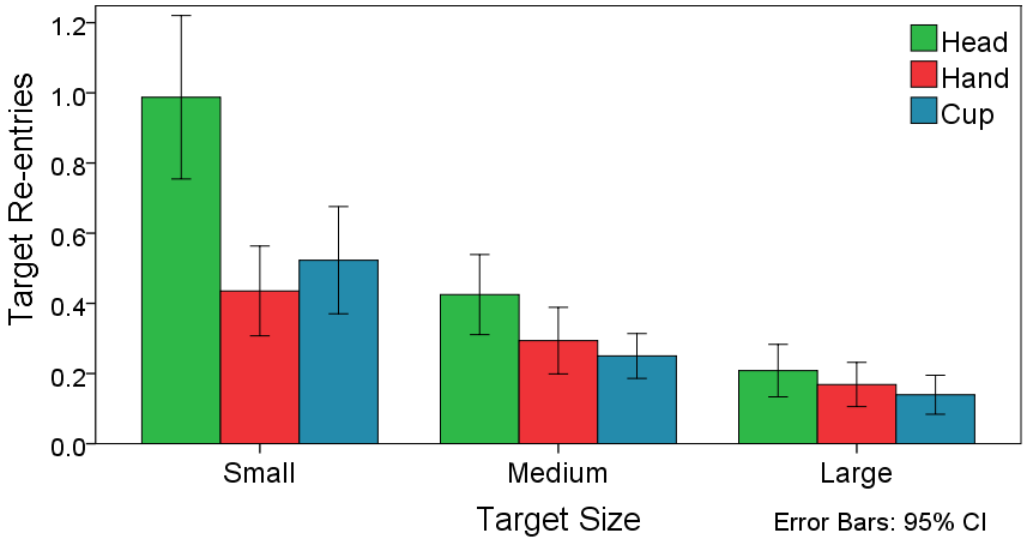

Figure 4.15: Target re-entries for each target size and movement condition.

Where, MT_{input} is the predicted movement time for a movement condition, measured in seconds. The parameters for a and b, and the R-squared model fit values are given in Table 4.1.

We were interested to see whether a statistically significant difference existed between target re-entries for different movement conditions and target sizes (Fig. 4.15). For this, we used a two-way repeated measures ANOVA. The data was normally distributed, as assessed by boxplot and Shapiro-Wilk test ($p > .05$), and in all cases the assumption of sphericity was not violated, as assessed by Mauchly's test of sphericity ($p > .05$). We discovered significant main effects for movement conditions ($F_{2,22} = 24.836, p < .001$), size ($F_{2,22} = 44.341, p < .001$), and a significant interaction for movement condition $\times$ size, ($F_{4,44} = 11.472, p < .001$). We further analysed the main effects using Bonferroni corrected pairwise comparison of means.

For the main effect of movement condition, we observed that the head had a significantly higher number of target re-entries (0.541) compared with both the hand (0.299) and cup (0.304), $p < .001$. There was no statistical significant difference between the hand and the object, $p = 1.0$. The main effect for size showed that smaller targets (0.649) resulted in a higher number of target re-entries compared with medium targets (0.323), which in-turn resulted in a higher number of target re-entries compared with larger targets (0.172). In all cases the results were statistically significant at $p < .005$.

Input	*a*	*b*	R^2	TP (bits/s)
Head	-1.000	1.345	0.766	1.03
Hand	0.417	0.624	0.556	1.35
Cup	-0.118	0.857	0.795	1.25

Table 4.1: Fitts' Law parameters, model fits, and throughput (TP) for each movement condition.

To analyse the movement condition $\times$ size interaction, we performed a one-way repeated measures ANOVA for each size to assess which movement conditions, if any, caused a significant difference to target re-entries. For all sizes, Mauchly's test of sphericity was not violated. There was no significant difference between movement conditions for the large size, however there were significant differences for both the medium ($F_{2,22} = 5.513, p < .011$) and small sizes ($F_{2,22} = 22.546, p < .001$). Post-hoc Bonferroni corrected pairwise comparisons revealed the head had a significantly higher number of target re-entries compared with the cup for medium target sizes, at $(p = .011)$, and both the hand and cup for small target sizes, at $(p < .005)$. No false motion-matching activations occurred during the study.

4.4.5 Study Discussion

The larger throughput and number of target re-entries for the head may be a result of both user and system performance. The head is used much less frequently for everyday pointing tasks compared with the hand, and the smaller range of movement at a distance of over 2m from the camera equates to fractional changes per pixel between frames. The moving window used to reduce cursor jitter for very small movements introduces a time lag which could have affected the user's fine-grain pointing performance. In future work, this could be alleviated using augmented cursors, which have been shown to improve target acquisition of smaller targets for gestural interaction [62].

Table 4.2 details prior work which used the ISO 9241-9 multi-directional pointing task to examine the throughput of input devices for head and hand gestures. A direct comparison of devices cannot be made due to the variability of participants, distances to the input device, measurement of endpoint deviation, selection of IDs, and difference in selection techniques. However, it appears that MatchPoint has a similar throughput to the first version of the Microsoft Kinect when used at larger distances. This would suggest that the dynamic setting of control-display gain had little impact on user performance when pointing. It is also important to note that we used a dwell time of 240ms to simulate the time taken to press a button. This may be suitable for some tasks, but for others a larger dwell time may be needed to reduce false detections when a user hovers over a control.

We observe very low R^2 values for the Fitts' law models, see Figure 4.16. Based on prior research one would assume the physiological nature of the hand and arm movements of the users follow Fitts' law, and we therefore look to technological reasons for the low values. The original reasoning behind the introduction of effective index of difficulty was to account for the speed-accuracy trade-off, and assumed that users who were less accurate were performing the

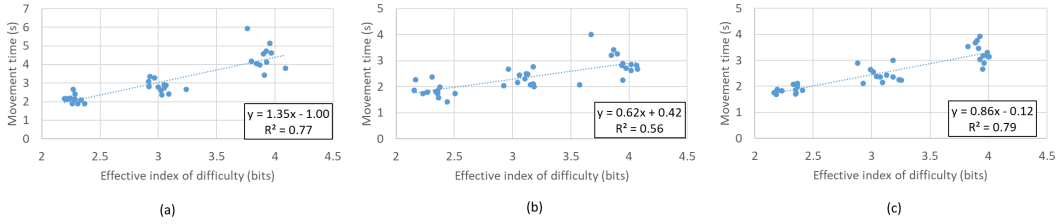

Figure 4.16: Line graphs of movement time versus effective index of difficulty for (a) head, (b) hand, and (c) cup.

Distance to Device	Study	Movement Condition	Input Device	TP (bits/s)
<1m	[35]	Hand	Leap Motion	2.8
	[48]	Hand	MS Kinect v1	1.42
	[117]	Head	Head-mounted marker	1.61
	[200]	Hand	MS Kinect v1	2.1
	[278]	Head	Optical IR markers	1.40
>1m	[24]	Head	MS Kinect v2	2.3
	[24]	Hand	MS Kinect v2	2.45
	[148]	Head	MS Kinect v1	0.75
	MatchPoint	Head	Webcam	1.03
	[235]	Hand	MS Kinect v1	1.19-1.38*
	MatchPoint	Hand	Webcam	1.35
Unknown	[135]	Hand	Wii Remote	2.97
	[227]	Hand	MS Kinect v1	0.5-2.0**
	[235]	Hand	Swiss Ranger 4000	0.75-1.57*

Table 4.2: Touchless input devices used in previous studies that used the ISO 9241-9 multi-directional pointing task to assess throughput (TP) for hand and head gestures. * indicates a range of throughputs due to different selection techniques. ** indicates estimated throughput.

movements more quickly [249]. With MatchPoint, this may not be the case. We observed a relatively large number of target re-entries across the inputs (especially in the case of the head), which in combination with the dwell time used could affect the movement time. Figure 4.16 shows more variability for the higher index of difficulties. This may have been due to users struggling more with fine-grained movement, in which the technique is more prone to jitter. Although a moving average filter was used to suppress the jitter, it inherently introduces lag, and therefore users have either jitter or lag to contend with – both of which could contribute to the lower goodness of fits.

4.5 Discussion

Spontaneous spatial coupling can support wide-ranging applications by enabling flexible touchless input over a distance. At the core of the concept is the motion-matching phase – it empowers users to simultaneously select the function they wish to control, and the input to use (implicit in their action). The selections made by the user, and additional contextual information such as scale and range of motion observed in the matching process, in turn enable input to be uniquely tailored to the context. As shown, this encompasses tracking of the specific input of choice as a pointing device, calibration of the control display gain based on context, and the possibility to map input in a task-specific manner to parameters of the selected function.

The dynamic appropriation of "anything the user can move" as a pointing device presents a new design opportunity, inviting exploration of mappings that might not be general purpose but fitting for specific contexts. Our design space exploration shows the concept extends to spatial coupling of multiple controls at a time, by one or multiple users, with body parts and/or objects. This opens up a compelling design space, for which we have provided an initial framing and demonstrated a range of novel techniques.

MatchPoint provides a highly deployable implementation of spontaneous spatial coupling. The

system requires only an off-the-shelf RGB camera, and uses low-cost computer vision techniques that are able to track input without the need for recognition of objects or body parts. Other sensing modalities could be considered for spatial coupling, for example depth sensors to extend motion-matching and pointing into 3D, or inertial measurement units, to leverage sensors that are widely deployed in mobile and wearable devices.

MatchPoint's ability to accept any form of input is compelling as it enables users to choose a form of input that is convenient in a given context. As shown, users perform well with MatchPoint for pointing over a distance, but the way in which a user provides input can affect performance – raising the question of when to design for flexible choice versus specific types of input. Based on its ability to accept any form of input, MatchPoint could also be deployed as an accessibility device, to provide users who can not operate a conventional mouse with a flexible alternative.

The motion-matching phase in MatchPoint is based on circular motion, adopted for the purpose as it provides uniformity to acquisition of controls. However, the system could be extended to support matching with any shape of motion by using a generic model fitting approach. Matching against any type of motion could provide designers with additional opportunities, for example selection of graphical objects for manipulation by tracing their outline, or use of polygonal motion paths as corners could help users synchronise.

There are several limitations in the current implementation of MatchPoint. The tracker used for spatial coupling does not handle occlusion, and simultaneous motion could result in tracking errors when feature points are detected for body parts connected to the user's desired input (e.g. tracking of the elbow when using the hand). If multiple people perform the exact same motion at the same time the system might also attempt to track their combined movements. The system may also attempt to track the hand when it is removed from a physical object when creating tangible interfaces. These limitations could be overcome by incorporating object recognition and segmentation of body parts, which would also open up the possibility for designers to present different interfaces for a control depending on which input is being used.

4.6 Conclusion

In this chapter we have demonstrated how expressive motion correlation can be for interaction when combined with spatial input (RQ2). Spontaneous spatial coupling is a powerful concept for touchless input as it empowers users to dynamically appropriate any part of their body, or object they hold, as a pointing device. The concept leverages motion-matching as an intuitive method for users to select a control while implicitly creating a spatial coupling that is tailored to the context, supporting the acquistion of a pointer as and when needed (RQ2.1). The concept also opens up an entirely new design space for interactions that leverage spontaneous coupling of multiple controls at a time, by one or multiple users, with different body parts, or with objects as tangible intermediaries (RQ2.2). MatchPoint is a systems contribution that provides a complete implementation of spontaneous spatial coupling. The system lends itself to wide deployment as it only requires an off-the-shelf camera and computer vision for detection, matching and tracking of motion input. The system is able to take input of any form, and adapts the control display gain to provide users with a comfortable input range. We demonstrate that users are adept at providing input with different body parts, and whilst holding an object, and that the throughput of MatchPoint using only a webcam is similar to that of the Microsoft Kinect v1 (RQ2.3).

5

In-depth Analysis of Motion Correlation with Body Movements

Chapter 3 demonstrated the versatility of motion correlation when movement is used as the sensing principle, and Chapter 4 showed how motion correlation can be used to establish co-ordinate frames on a per-interaction basis which enables unique opportunities for interaction. However, one of the primary factors of whether a technology is adopted include user experience and users' expectation of sensing robustness, which we address in this chapter. In general, gesture detection and recognition using computer vision techniques is non-trivial with confounding factors such as lighting, scale, occlusion, etc. Users expect 100% accuracy and are adept at detecting events that do not align with their expectations. Another important consideration when designing an input method for touchless gestures is understanding its robustness to the Midas touch problem. Ideally a system should detect a user's intent to interact, whilst rejecting accidental activation. Motion correlation has the potential to reduce the Midas touch problem associated with touchless gestures because detection is based on both spatial and temporal properties of the user's movement. The predominant technique used in the motion correlation literature, and in this thesis, relies on variations of the Pearson product-moment correlation. This positions the detection problem as one of matching two trajectories, where there has been a plethora of work in other communities based on time series classification [238], and trajectory mining [282, 304]. This raises the question of whether or not the Pearson product-moment correlation coefficient is the best detection algorithm for motion correlation.

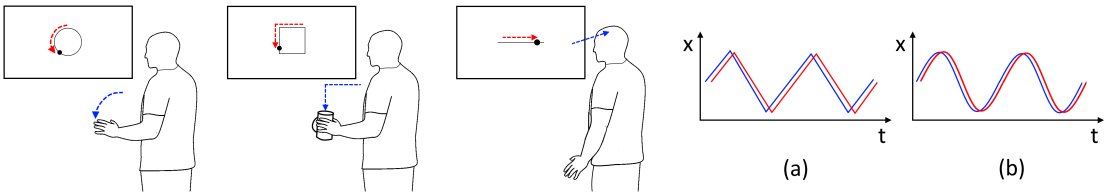

Figure 5.1: In this chapter, we evaluate how well users synchronise with external stimuli in the context of motion correlation under different conditions. Movement is presented on displays using four different shapes (circle, square, horizontal and vertical lines), and both (a) linear and (b) simple harmonic motion. Users are tasked with following the movement with four inputs (head, dominant hand, non-dominant hand, and cup-in-hand). After extracting the ground truth of the user's movements, we analyse how well they are able to synchronise to inform both interface and algorithm design.

To select the most appropriate algorithms from the literature, and to better design new algorithms, we look towards an understanding of how users synchronise with target movement. This is currently lacking in the context of motion correlation using body movements. Sensorimotor synchronisation literature has studied how we synchronise with external stimuli, however work most closely related to the context of motion correlation is not directly applicable because they have focussed on one-dimensional movement of the hand (e.g. [36]), use visual feedback of the user's position (e.g. [222, 42, 250]), or use two-dimensional movement synchronised to discrete stimuli such as a metronome (e.g. [255]). Work on motion correlation interfaces also provides potential insights of how well we can synchronise. PathSync found higher recognition rates when users synchronised with squares, hypothesising that the corners provided saliency and increased user's ability to synchronise [40]. Freeman et al. found that users preferred, and had higher success rates with, one-dimensional target movements in comparison to circular movements [80]. In Chapter 3, we showed that users struggled synchronising with the fast head movement, and overall found higher success rates with slower moving targets. However all of these insights are based on an inference that detection rates equate to user's ability to synchronise with a moving target. They fail to take into account any deficiencies in the detection process – either user (e.g. poor synchronisation) or sensor (e.g. input lag) induced. This raises the fundamental question of how well can users synchronise with moving targets as measured in the context of motion correlation (RQ3).

The underlying detection process can only be as good as users' ability to synchronise with the motion. So far, we have presented targets to users as circular orbits which move with a uniform speed. Previous work has also investigated how the hands can match targets moving with constant velocity for squares, diamonds, and one-dimensional lines [40, 81]. This motivates an investigation into both how well these shapes can be matched using other inputs, but also if we can better design target movement for body movements (RQ3.3). It has been shown that hand movements exhibit simple harmonic motion when performing simple reciprocal pointing tasks [94]. After all, when we move our body we must accelerate and decelerate a mass. Based on this knowledge, we posit that by presenting target movement as simple harmonic movement we can leverage the kinematic properties of how we move our bodies to help users synchronise more accurately with targets.

In order to answer these questions we follow a data-driven approach where we first collect data on eight users, see figure 5.1. Algorithms should be sensitive enough to detect when a user synchronises with a moving target, but specific enough as to not detect movement that does not correspond to a moving target. We therefore gather two datasets, one on users matching target motions which forms our *true positive* dataset, the second we collect on users performing body movements in the form of spatial and semantic gestures which we use as a *false positive* dataset. The goal of a detection algorithm is to maximise the detections in the true positive dataset, whilst minimising detections for the false positive dataset. After collecting the data, we first analyse user behaviour to gain insights on how well they can synchronise against the moving targets under different conditions (RQ3.1). To inform algorithm design, we extract the temporal difference between the user's gesture and the corresponding position in the target trajectory to see how users lead or lag the target and how this evolves over time hl(RQ3.2). We also extract additional information on user behaviour to inform interface design, including the size of the gesture, and time taken to start. We use insights gained to select and evaluate eight algorithms from both the motion correlation and time series comparison literature, and perform an extensive search of the parameter space (RQ4).

Our aim in this work is to provide a foundation for understanding motion correlation at a deeper level, to help inform both algorithm and interface design. We advance theoretical and practical understanding of motion correlation through the following contributions:

- A deeper understanding of how users synchronise with moving targets in the context of

motion correlation across different target conditions to inform both interface and algorithm design;
- A demonstration of how simple harmonic motion can be used to support users synchronise with target movement by leveraging the kinematic properties of how our bodies move;
- An extensive parameter search of eight algorithms, demonstrating the robustness of motion correlation as a technique and providing parameters for practitioners.

5.1 Data Collection

To better understand how users follow a moving visual target with body movement, and to collect data for algorithm evaluation, we collected data of participants mimicking the motion of a moving target with varying properties:

- Shape: circle, square*, vertical line*, horizontal line*;
- Type of movement: linear, harmonic*
- Speed (Time per cycle): 2s, 4s
- Direction: clockwise, anti-clockwise

Note, only shapes marked with * use the harmonic type of motion, and for the circle we use uniform circular motion. The movement of a target moving with simple harmonic motion oscillates sinusoidally, and is defined by:

$$x = x_0 \cos(\omega t + \varphi) \tag{5.1}$$

Where x_0 and φ are constants, and ω is the angular frequency of the oscillations. We investigate users' ability to synchronise with moving targets using four Movement Conditions (head, dominant hand, non-dominant hand, and a cup). We also capture users performing a number of common semantic and spatial gestures that may be used during everyday life, see Table 5.1. These form a *false positive* dataset with which we assess how robust motion correlation is against movements not intended for interaction. Gaining an ecologically valid false positive dataset that covers the wide variety of accidental movement one would expect to find in real-world applications is non-trivial due to amount of data required and ethical concerns surrounding in-the-wild data capture. By acquiring data on semantic and spatial gestures we gather common movements which could cause false activations, and against which we evaluate the robustness of motion correlation.

5.1.1 Participants and Apparatus

Eight participants (5M/3F) aged between 21 and 40 years (mean= 27.9, s.d.= 6.25) were selected to take part in the study. All of the participants were right-handed. Four participants had no

Table 5.1: List of communicative gestures used to detect robustness of algorithms to false positives from accidental movement

Input	Communicative Gesture
Head	Nod; Shake; Look at objects
Hands	Thumbs up; Thumbs down; Unsure; Okay symbol; Stop; Go away; Hurry up; Finger wave; Hand wave; Rub together
Body	Shrug shoulders; Stand and sit; Get comfortable; Walk around; Pick up objects; Point at objects

experience with motion correlation input techniques, one had experienced gaze-based smooth pursuit motion correlation, one had experienced body-based motion correlation, and two had experienced both gaze- and body-based motion correlation techniques.

The data collection took place in a lab with a 55" Samsung Smart TV (1920×1080) and a couch placed 2.23 m from the TV (based on a TV size to viewing distance calculator). An unmodified, off-the-shelf Logitech C920 web camera was mounted on top of the TV. The camera captured a 960×720 region of interest within a 1920×1080 image at 30 frames per seconds to control that only movement related to the study was captured. Participants were seated on the couch. We extract body movement using a combination of a web camera and skeletal extraction using OpenPose [284, 245, 39, 38], rather than a Kinect because preliminary testing revealed this approach was more stable – in particular when the arm was perpendicular to the camera.

5.1.2 Procedure

After filling out demographics information and signing a consent form participants were seated comfortably in the middle of the couch facing the TV. They were then instructed to perform a series of semantic and spatial gestures aimed towards the experimenter sat off camera, the order of which was chosen randomly and verbally communicated to the participant, see Table 5.1. They were told they were free to perform the gesture as they wished as the primary goal was to elicit movement. If they did not understand what the gesture meant the experimenter verbally described the context in which the gesture would be used, e.g. stop – "if you were to non-verbally indicate to someone to stop what they were doing". The participants were recorded for the duration of the gestures (including time in between).

Following this, participants performed four blocks of following target motion, one for each movement condition. The order of movement condition was counter-balanced using a Latin square and the remaining factors were randomised. The cup was half-filled with water to simulate someone holding a drink. Each condition was performed two times. There is no notion of clockwise and anti-clockwise for one-dimensional shapes (vertical and horizontal), however we include these as dummy conditions which effectively increases the repetitions to four for these shapes. Participants were instructed to perform all movements using a comfortable range of movement. They were instructed to perform hand movements from the elbow, as opposed to the wrist or finger, to improve automatic detection of hand movements. They were instructed to rest their hands between trials, and that they could change the posture of their hand between, but not within, trials. The former instruction was to make sure the participant did not get tired, but also to ensure that each movement started from a resting position.

For each trial, the participant was instructed to follow the movement of the target as soon as it appeared on the screen. Each movement lasted for eight seconds, with a (≈ 4 s) break in between, and was shown on the TV with a path length (i.e. circumference in the case of the circle) of 400 pixels for two-dimensional movements, and 200 pixels for one-dimensional. The starting point of the motion was randomised for each trial. A recording was taken for the duration of the eight seconds. Users completed all trials for a movement condition before moving onto the next. After each movement condition, participants were asked for their preference on shape, speed, and type of movement. After all trials were completed we then asked for participant preferences again as a whole, and which movement condition they preferred to use. In total, each participant performed $4 \times (4+3) \times 2 \times 2 \times 2 = 224$ movements.

5.1.3 Movement Extraction

For understanding how well users synchronise with moving targets, and for evaluating algorithms, we need to extract user movement from the recording. Both user and target movements

are spatiotemporal trajectories in which each spatial position is recorded at a specific time. A user's trajectory $\tau_u = \{\mathbf{P}(t_0), \mathbf{P}(t_1), \ldots, \mathbf{P}(t_n)\}$ is an ordered sequence of spatiotemporal positions, $\mathbf{P}(t) = \binom{x(t)}{y(t)}$, representing the user's position in the camera's image plane at a given time stamp. Likewise the target's trajectory, τ_t, is made up of spatiotemporal position representing the target's position on the display. The system used for recording uses a polling mechanism. Each time a frame is captured by the camera, the corresponding position of the target on the display is recorded alongside it, along with the timestamp of when this was recorded.

In order to analyse user movement we extract both head and hand movements from the recorded videos. This is performed using the OpenPose multi-person keypoint detection library [284, 245, 39, 38]. Each video was post-processed with OpenPose, extracting the body keypoints for those trials involving hand movements, and face keypoints for head movements. For hand movements we use a single keypoint of the wrist from the OpenPose library, as we found this more stable than the finger-keypoint detection which constantly dropped frames. Each keypoint has an associated confidence level from the OpenPose library (0-1). If this is above 0.4 then we assume the joint position is accurate, if not then we linearly interpolate between valid points. For head movements we use the facial keypoints reported from OpenPose. We select those keypoints corresponding to the nose as these should remain relatively stable irrespective of a user's facial expression or speech, unlike eye or mouth features. In the event all of the features of the nose are not found with a confidence over 0.4, then we initialise a median flow tracker from the last known valid position to interpolate the missing frames. We chose the median flow tracker because of its error reporting capabilities in the event the lack of features is due to the face no longer being visible in the frame (e.g. in the case of the false positive dataset). As the cup is a non-deformable object subject to mainly translational movements, we use a computer vision object tracker. We manually initialise a bounding box and then track the object using a discriminative correlation filter tracker with channel and spatial reliability (CSR-DCF) [154].

For the remainder of the analysis it is important to note that there is sensor lag involved in the data collection system. When capturing users, there is camera input lag - a time delay between when the photons hit the camera lens and when they are sent to the PC for processing due to exposure of the photosensitive sensor and the on-board encoding of the image. We measured the lag of the camera used to perform the recording at approximately 100-150 ms. There is also input lag for the TV used to display the motion – a delay from when the PC sends the command to display an image to when the image is actually displayed on the screen. We reduced this by setting the TV to "game mode", which according to the manufacturer reduces input lag to 33 ms. In terms of synchronising user input with system output these effects cancel each other out, so we expect the difference to be between $\approx$70-120 ms. For the remainder of this work we perform the analysis on the basis of detecting user motion using a webcam and the inherent input lag involved. Results are not intended to represent millisecond measured performance metrics of human motor synchrony. Instead, we use them to inform design of applications where we find the same sensor limitations present.

5.2 Understanding Synchronous Body Movements

In this section, we extract ground truths of how capable users are at synchronising with moving targets. In Chapter 2.4.2, we discussed previous work in the sensorimotor synchronisation literature which has studied our ability to synchronise with external motion. In Chapter 3, we investigated users' ability to synchronise using algorithms which are intended for the matching process. Previous work in the motion correlation literature have also drawn insights of how well we can synchronise with movement [69, 40, 81]. However, in these cases any analysis is limited by the algorithm's capabilities and not necessarily how the user synchronises with the motion. In

contrast, we look at extracting the ground truth offline, independent of any particular detection algorithm intended for real-time use.

Previous work that has studied gaze-based smooth pursuit interactions has done so using calibrated gaze to detect where the user is looking, and to compare the absolute position of the gaze point to the target's trajectory. Analysis has then focussed on how the detection algorithms cope under different conditions [208, 63]. However, touchless gestures are ephemeral and do not leave behind any record of their path. Knowing where the user is compared to the target is non-trivial because there is no known mapping between input and output space, and the user's internal point of reference may change over time. Instead one must infer at what point a user is in their gesture in both time and space.

Ultimately, the understanding of how users follow moving targets can help us better design algorithms for detection by providing insights into how much users lead or lag a target, and how this varies over time. It also provides insights about which movements to use for motion correlation interfaces based on understanding users' preferences and capabilities. In particular, we aim to answer three fundamental questions:

- How successfully are user's able to follow the moving target over time?
- How long does it take user's to start synchronising with a target stimuli?
- Do the properties of the target movement affect the size of the user's movement?

The first question is required for us to understand how much users lead or lag a target over time, depending on different target conditions. For this we extract the *user-to-target difference* – the temporal difference between where the user is in their gesture, and when the target was displayed on the screen at the corresponding position. When aggregating the user-to-target difference across multiple users, we need to extract the time taken to start the gesture (*time-to-start*) so that signals can be temporally aligned. This is defined as the time between the start of the trial when the user is at rest, until they start following the target, but does not necessarily involve steady-state synchronisation with the moving target. In addition to temporally aligning signals, we use the time-to-start to investigate how long it takes to start a gesture based on different shapes or target speeds because it is analogous to the acquisition time of an input device (e.g. picking up a remote control), and indicates how long it takes for a user to be in a position to affect control. To answer the final question we extract the *size of gesture* in order to provide an indication of how much effort is required to perform the gestures, measured in pixels relative to the camera. In particular we are interested in the difference across shapes, which we use to provide additional insight into interface design and user preferences. Specifically, we measure the length of the user's trajectory when completing the first cycle of the shape (i.e. one rotation in the case of the circle).

5.2.1 Ground Truth Extraction

The first step in extracting the ground truth is to extract the start of the gesture. Once we have extracted the start gesture for each trial, we can temporally align all the trials for comparison of the difference between user and target movements across users and movement conditions. We calculate the difference between target and user movements by using a moving window ellipse fitting approach for circular shapes, and a vertex and edge extraction approach for square and one-dimensional shapes. Prior to analysis we filter all positional data of the user with a zero-lag third order Butterworth filter with a critical frequency of 3Hz.

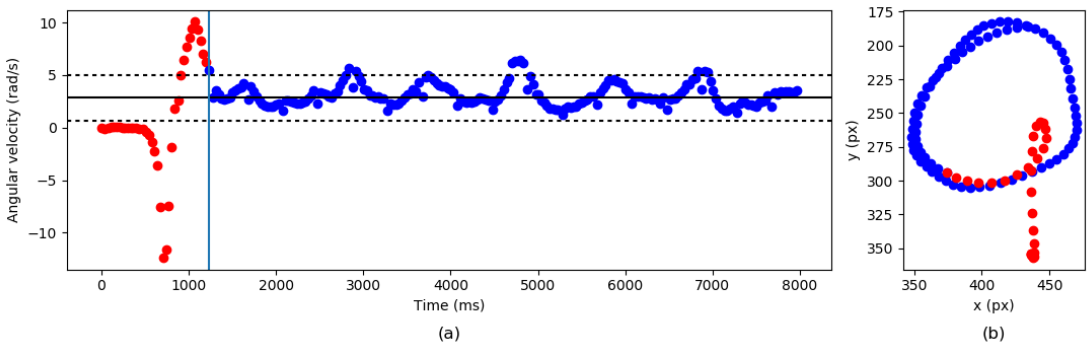

Figure 5.2: An example of the gesture start calculation for circles: (a) angular velocity of the user's motion with horizontal lines showing median and three median absolute deviations; (b) first four seconds of the trajectory. Blue points indicate valid part of the trajectory, red points indicate movement prior to the gesture.

Circle

For circular shapes we use least-squares ellipse fitting [76] over a moving window which returns the centre, width, height and angle of the ellipse fitted to the data points. For each data point in the user's trajectory we can calculate the angle relative to the centre of the fitted ellipse to infer where in the circle the user is, and compare this to the target's motion. This is measured in degrees, however we convert it into milliseconds to compare across the different angular velocities of the targets. When fitting an ellipse to the user's trajectory the window size is an important factor which affects the centre of the ellipse and therefore inference of where the user is relative to the target motion.

The window size of the ellipse fitting process is chosen based on the arc length of the target movement, which takes into account the different speeds of the target's, and hence the user's, movement. We calculate the angular velocity of the user's movement for a variety of different arc lengths, ranging from 0.5 to 1.2 in 0.025 increments. An important property of the angular velocity is that it is independent of the size of the circle the user is tracing. For each trial, we choose the arc length that minimises the variance of the user's angular velocity. The goal of this is to ensure that any variation in angular velocity, and hence relative position of the user, is due to their movement and not the ellipse fitting process. The initial portion of the user's trajectory will contain motion that is not related to the user following the target, but rather to the initial resting phase and synchronisation with the target. We therefore discount the first 3 seconds when calculating the arc length so that only motion corresponding to the motion-matching phase is used.

Once we have calculated the appropriate arc length to use, we calculate the starting position of the user's gesture. We define the starting point as the time where the user starts performing a stable circular gesture, e.g. after the hand has moved from its resting position to start following the circle. It is important to note that the starting point calculated is not necessarily the point at which the user matches the movement of the target - because there may be catch-up or slow-down behaviour to synchronise after the gesture has begun. In order to calculate the starting point we once again look towards the user's angular velocity, in particular outlier values which indicate the user is not performing a stable circular movement. At the start of a gesture outliers can be caused by either the user moving their hand in a ballistic fashion to start the gesture from a point of rest, thus increasing their angular velocity, or due to the ellipse fitting process fitting very large "circles" to the user's movement as they are viewed as straight lines due to the ballistic movement of moving the hand into a position to gesture. We calculate outliers of the angular velocities by computing the median and median absolute deviation (MAD) [146]. The median

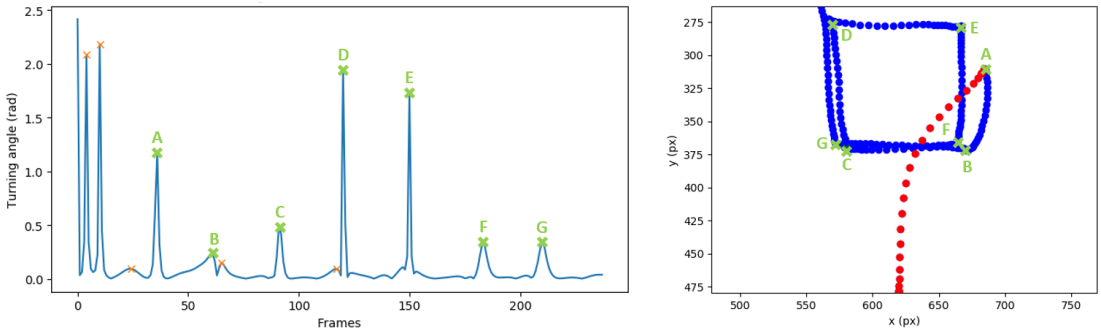

Figure 5.3: An example of the peak detection process to extract corners. Green corners labelled alphabetically are the final corners detected, with orange corners found in the peak detection removed as anomalies or duplicates.

and MAD are more robust measures to outliers than using the mean and standard deviation. The starting point of the gesture is then defined as the first data sample which is not an outlier, and in which the proceeding 30 samples ($\approx 1s$) are also not outliers, see Figure 5.2.

Squares and lines

For the remaining shapes we extract vertices and edges in order to match the user's movement to the target's. We begin by extracting the velocity vector, $\mathbf{V} = \frac{\Delta P}{\Delta t}$, of the trajectory and pass it through a zero-lag third order low-pass Butterworth filter with a critical frequency of 2.25Hz. The *direction* of the velocity vector, $\angle\mathbf{V}$, indicates which way the trajectory is moving at any given time point. For horizontal and vertical shapes, we are only interested in finding the vertices as a result of changes in direction of the velocity vector in the respective axis (e.g. the x-axis for horizontal movements) and so discard the other axis. We use a peak finding algorithm to find local maxima of the differences between the directions of the velocity vectors which correspond to candidates for the vertices of the trajectory, see Figure 5.3. We take a liberal approach to peak detection as any false positives will be filtered out down-stream in the next stage.

For each edge, e between two vertices, v^i, we calculate its length, $\|e\|$, and the angle of the edge defined as:

$$\angle e = \text{atan2}(v^y_{i+1} - v^y_i, v^x_{i+1} - v^x_i) \tag{5.2}$$

Where v^x_i is the x co-ordinate of the ith vertices, and v^y_i is the y co-ordinate.

Using these two metrics we filter out vertices that are either anomalous or duplicates, see Figure 5.3. We define an anomalous edge as one which is smaller than 10% of the average of all the edge lengths, and in the case of the square where the angle is $180° \pm 45°$ relative to the previous edge (as we are only interested in right-angled corners). These are recursively removed until no anomalies are found. The second data cleansing phase is to remove "duplicate" vertices, which we define as when two consecutive edges have the same angle within $\pm 45°$. These are also recursively removed until no duplicates are detected.

After extracting vertices and edges for both user and target trajectories, we match the vertices of the user's trajectory to the corresponding vertices in the target's trajectory. To achieve this we iterate backwards through the edges and look to maximise the number of sequential matches between target and user edges based on the direction of the edge. We do this in reverse order because at the start of the movement the user is in a resting position and therefore not in sync with the target.

To remove movement unrelated to following the target we define the start of the gesture as the first user vertex which is mapped to a target vertex. It is important to note that the start of the

gesture may have occurred somewhere on, or at the start of, the previous edge, however we have no way of knowing where the user started the gesture relative to the movement of the target without a whole edge consisting of two vertices. Thus, the first point at which we can be certain of the user's intention relative to the target movement is the first matching vertex.

The difference in time between each data sample in the user's trajectory to the respective point in the target's trajectory determines how much the user leads or lags the target. To calculate this, we calculate the user's position in the edge proportional to the length of the edge, and find the nearest corresponding point in the target's trajectory according to the same criteria. The difference in time stamps between these points indicates how much the user leads or lags the target.

5.2.2 Results

In this section we look at the general trends for users synchronising with the target by analysing how quickly they are able to achieve synchronisation, how much they lead or lag once synchronisation is achieved, and how large their movements are whilst synchronising. After calculating the time difference between the user and target at every data point in the user's trajectory, we calculate the time difference between the user and target in bins of 100ms for comparison across users and the different conditions. For each 100ms we record the mean difference (negative values indicate the user is lagging) and 95% confidence interval. When aggregating the results we removed any trials that did not have four seconds worth of synchronisation data after the start of the gesture had been detected. This could be due to errors in the ground truth extraction process, or in the event of the slow horizontal and vertical movements because of insufficient complete edges to analyse. In total we removed 46 out of 1,792 trials (3.5%) - 7 square (1.4%), 24 horizontal (4.7%) , and 31 vertical (6.1%). We then aggregated the data across all participants, directions, and repetitions for analysis.

Time-to-Start Gesture

The time-to-start gesture, see Table 5.2 indicates how long it takes users to begin the gesture, and is important for temporally aligning participants' movements across trials in our later analyses. For the time-to-start gestures we make statistical comparisons based on how we extract the start of the gesture. The time-to-start gesture for the line-based movements, including the square, are worst case scenarios based on the first vertex detected. We therefore look at the circular shape separately. We also look at the square shape independently of the horizontal and vertical lines, because the edge length of the line-based movements are twice that of the square, i.e. for fast square the vertices are every 0.5s, whereas for fast lines they are every 1s.

Table 5.2: The mean time to start the gesture across all participants, measured in seconds. The standard error is given in brackets. (H) indicates harmonic movement.

	Slow				Fast			
	Head	Dom	Non	Cup	Head	Dom	Non	Cup
Circle	0.42 (0.05)	1.08 (0.12)	1.19 (0.11)	1.05 (0.16)	0.66 (0.09)	1.20 (0.06)	1.33 (0.10)	1.14 (0.11)
Square	0.95 (0.05)	1.14 (0.05)	1.19 (0.05)	1.14 (0.04)	0.90 (0.08)	0.94 (0.04)	1.03 (0.04)	0.99 (0.07)
Square (H)	1.06 (0.08)	1.19 (0.04)	1.10 (0.04)	1.14 (0.06)	0.86 (0.04)	1.00 (0.04)	1.00 (0.04)	1.05 (0.05)
Horizontal	1.44 (0.07)	1.30 (0.09)	1.44 (0.10)	1.56 (0.12)	0.97 (0.02)	1.09 (0.02)	1.06 (0.06)	1.17 (0.03)
Horizontal (H)	1.31 (0.06)	1.12 (0.04)	1.27 (0.06)	1.28 (0.07)	0.94 (0.04)	1.06 (0.02)	1.00 (0.05)	1.14 (0.07)
Vertical	1.35 (0.06)	1.56 (0.07)	1.66 (0.17)	1.72 (0.08)	1.00 (0.05)	1.26 (0.08)	1.38 (0.08)	1.34 (0.08)
Vertical (H)	1.44 (0.06)	1.54 (0.15)	1.72 (0.08)	1.74 (0.08)	0.94 (0.02)	1.16 (0.04)	1.08 (0.04)	1.18 (0.03)

We compared the time-to-start the gesture for the **circle** with a two-way repeated measures ANOVA (Movement Condition×Speed) and Bonferonni-corrected posthoc tests. No cell in the design violated the assumption of normality according to the Shapiro-Wilks test, and Mauchly's test of sphericity revealed no interaction violated the assumption of sphericity. We found a significant main effect of Movement Condition ($F_{3,21} = 27.359, p < .001$) which revealed that the head (0.54s) was significantly quicker to start the gesture than the dominant hand ($1.14s$), non-dominant hand ($1.26s$), and the cup ($1.10s$) all at ($p < .001$), with no significant differences between the hand-based inputs. There was also no significant main effect of Speed between the fast (1.08s) and slow (0.94s) movements at $p = 0.055$. The head is the quickest input due to it being already 'in position', unlike the hands which were at rest between trials.

For the **square** shaped movement we compare the time-to-start gesture using a three-way repeated measures ANOVA (Movement Condition×Speed×Type), where Type is either linear or harmonic, and with Bonferonni-corrected posthoc tests. No cell in the design violated the assumption of normality according to the Shapiro-Wilks test, and Mauchly's test of sphericity revealed no interaction violated the assumption of sphericity. We find main effects for Movement Condition ($F_{3,21} = 5.994, p = .004$) and Speed ($F_{1,7} = 60.690, p < .001$). The head (0.94 s) was significantly quicker than the dominant hand (1.07 s) at $p = 0.013$, yet there were no statistically significant differences with the non-dominant hand (1.08 s, $p = 0.06$) or cup (1.08 s, $p = 0.211$). There were no differences between the hand-based inputs. The faster movements (0.97 s) were significantly quicker to start than the slower movements (1.11 s) at $p < .001$. It is worth noting that the difference in time-to-start for slower movements will be affected when detecting the first vertex due to the longer edge lengths compared with the faster movements. There could be a ≈0.25 s difference between fast and slow movements if we assume the starting point at which users actually start the gesture is equally distributed across the edge length prior to the first vertex we detect.

We compare the **horizontal** and **vertical** movements in a four-way repeated measures ANOVA (Shape×Movement Condition×Speed×Type). No cell in the design violated the assumption of normality according to the Shapiro-Wilks test, and we use Greenhouse-Geisser correction when the assumption of sphericity is violated. We found main effects for Movement Condition ($F_{1.391,9.735} = 6.898, p = .019$), Speed ($F_{1,7} = 1150.91, p < .001$), Shape ($F_{1,7} = 49.569, p < .001$), and Type ($F_{1,7} = 9.881, p = .016$). There were also statistically significant two-way interactions between Movement Condition×Shape ($F_{3,21} = 5.72, p = .005$) and Speed×Shape ($F_{1,7} = 9.451, p = .018$). Finally, we found a three-way interaction between Speed×Shape×Type ($F_{1,7} = 7.157, p = .032$). Both the head (1.17 s, $p = .001$) and dominant hand (1.26 s, $p = .002$) were significantly quicker than the cup (1.40 s), with no differences between the non-dominant hand (1.33 s). Harmonic movements (1.25 s) were significantly quicker than linear movements (1.33 s), $p = .016$. Horizontal movements (1.20 s) were also significantly quicker to start than vertical movements (1.38 s) at $p < .001$. Similar to the square, the faster movements (1.11 s) were significantly quicker to start than the slower movements (1.47 s) at $p < .001$, but there could be a ≈0.5 s difference between fast and slow movements if we assume the starting point at which users start the movement is equally distributed across the edge length prior to the first vertex we can detect.

Size of Gesture

We compared the size of gesture across all shapes using a three-way repeated measures ANOVA (Shape×Movement Condition×Speed). We combine the movement type (linear/harmonic) with the shape to compare across all shapes including the circle. We use Bonferonni-corrected posthoc tests for pairwise comparisons. No cell in the design violated the assumption of normality according to the Shapiro-Wilks test, and Mauchly's test of sphericity revealed no interaction

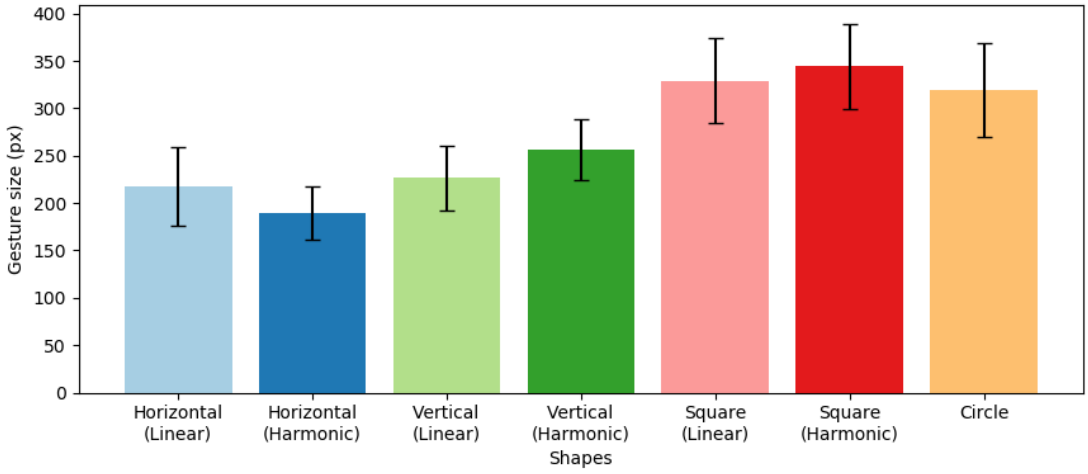

Figure 5.4: The size of the participants' movements (measured in pixels) for the first cycle of the gestures, showing that one-dimensional targets require less movement than two-dimensional despite the duration of the gestures being the same.

violated the assumption of sphericity. We found main effects for Shape ($F_{6,42} = 61.905, p < .001$) and Movement Condition ($F_{3,21} = 92.231, p < .001$), and two-way interactions for Shape×Movement Condition ($F_{18,126} = 8.508, p < .001$). Unsurprisingly, the head (63 px) results in much less movement than the dominant hand (337 px), non-dominant hand (338 px), or cup (338 px) at $p < .001$. Note how all hand-based inputs are within 1 px of each other, demonstrating consistency amongst the hand-based movements. The main effect for Shapes offers some interesting insights, see Figure 5.4. Both variants of the square (linear: 329 px, harmonic: 344 px) and the circle (319 px) show users gesture significantly larger than the one-dimensional shapes at $p \leq .019$. There was no statistically significant difference between horizontal movements (linear: 218 px, harmonic: 189 px) and vertical movements (linear: 226 px, harmonic: 256 px).

Synchronising with the Target

We quantify the average and range of the aggregated user-to-target differences across all participants. Based on visual analysis of Figures 5.5-5.8 we exclude the first half second after the start of the gesture when calculating the aggregated user-to-target difference so as to remove the initial synchronisation phase because users are not necessarily synchronised at the start of the gesture. We do not calculate this for the slow circle due to the high variability between movement conditions.

Circle: Across all movement conditions, participants lagged the fast moving circular target by an average of 183 ms (range = 47 ms), see Figure 5.5 (a). However, Figure 5.5 (b) shows the slow movements demonstrate much more variability than their fast counterparts, and participants overtake the target prior to slowing down in order to sync back up. Despite requiring less time to start the gesture with the head, there appears to be additional time required to achieve steady-state tracking of the target (at approx. 600 ms) compared with the hand-based inputs. Also, the head initially leads the target unlike the hands which lag. It is also interesting to note how closely the dominant and non-dominant hands follow each other for the slow movement. In contrast, the cup movement does not appear to overshoot the target as much yet follows a similar path from around 2200 ms. This could be due to the added weight of the cup which makes the overshoot less pronounced as it requires more energy to accelerate.

Square: Figure 5.6 (a) and (b) shows the starting user-to-target difference for fast movements

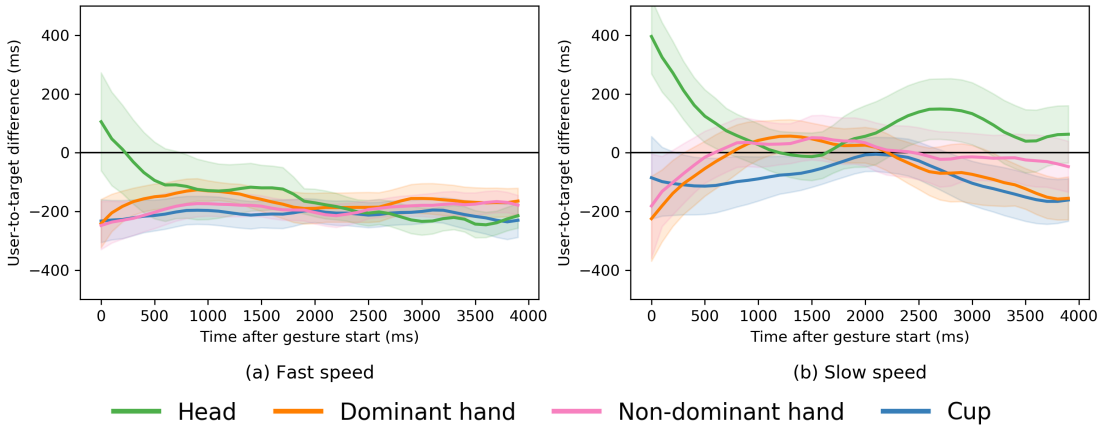

Figure 5.5: Synchronisation between the user and the target for the circular shape, measured in 100ms bins. Shaded areas show the 95% confidence interval. Negative values indicate the user is lagging the target, positive values indicate the user is leading.

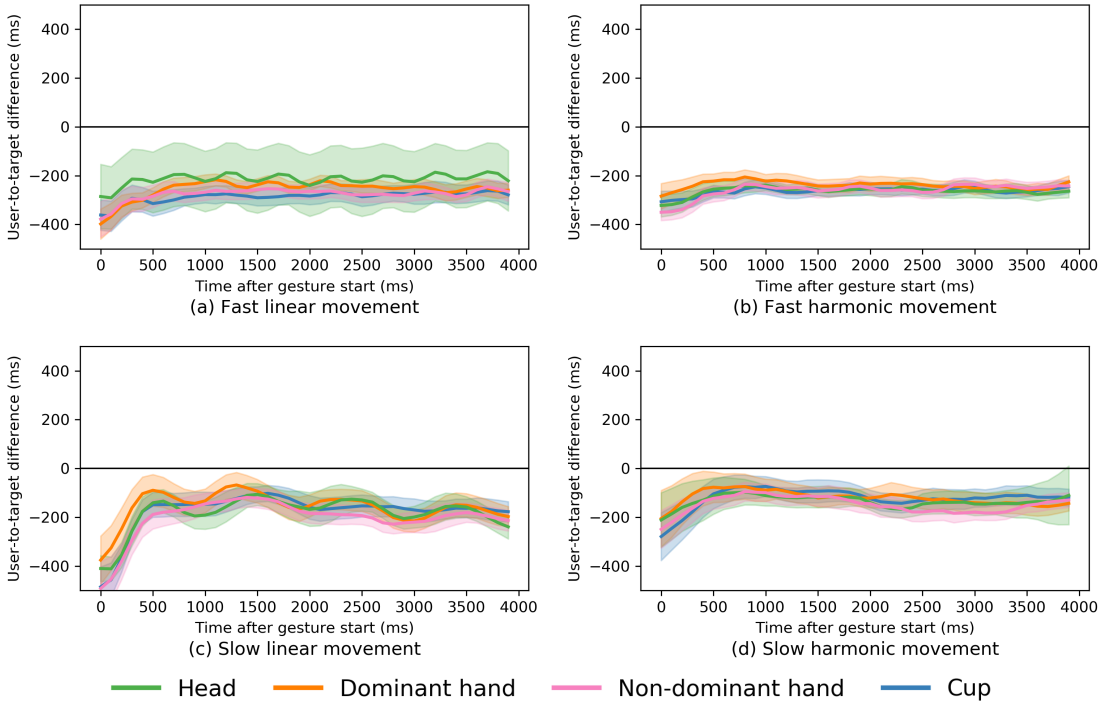

Figure 5.6: Synchronisation between the user and the target for the square shape, measured in 100ms bins. Shaded areas show the 95% confidence interval. Negative values indicate the user is lagging the target, positive values indicate the user is leading.

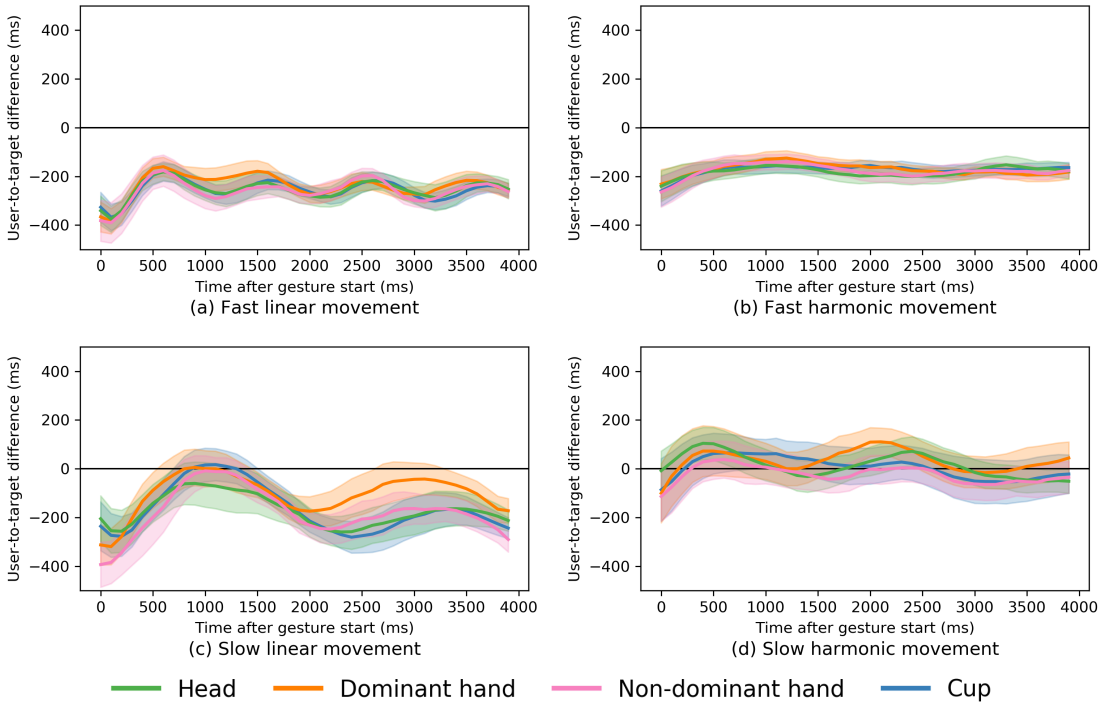

Figure 5.7: Synchronisation between the user and the target for the horizontal shape, measured in 100ms bins. Shaded areas show the 95% confidence interval. Negative values indicate the user is lagging the target, positive values indicate the user is leading.

was slightly smaller for harmonic (-316 ms) compared with linear movement (-356 ms), with near identical total average differences between linear (-250 ms) and harmonic (-251 ms). The head shows large variance across participants for the fast linear movement and one can see clear oscillations that correspond to the edge length of the square (with a cycle of 0.5s). Three participants signalled the square was particularly uncomfortable to perform with the head. Note however, that for harmonic movement we observe very little variability across the movement conditions.

Figure 5.6 (c) and (d) shows the difference in starting user-to-target difference was much more pronounced at the slow speed between the types of movement, with users being much closer to the target with harmonic movement (-236 ms) compared with linear (-441 ms). The average user-to-target difference shows users can follow the slower target more closely than their faster counterparts, however there was only a small difference between the different types of movement (Linear: -159 ms, Harmonic: -125 ms). In the case of the linear movement we see oscillations in the signal that correspond to the corners of the square (with a cycle of 1s), demonstrating the harmonic nature of our movements. At the corners the user maximally lags the target, before accelerating towards it to a local minima at the middle of the edge, and decelerating again until the next corner. This is reflected in the higher range of the average user-to-target differences for the linear (102ms) movement compared with the harmonic (63 ms). In contrast, the harmonic movement shows little oscillation.

Horizontal and Vertical: For both horizontal and vertical movements we observe three distinct differences between the linear and harmonic target motions that are much more pronounced than with the square, see Figures 5.7 and 5.8. Firstly, users are much more tightly coupled with the harmonic targets at the beginning across all movement conditions for both horizontal (Fast linear/harmonic: -354 ms/-249ms, Slow linear/harmonic: -287ms/-77ms) and vertical (Fast linear/harmonic: -386 ms/-247ms, Slow linear/harmonic: -357 ms/-22 ms). Secondly, based on the average user-to-target difference users appear to be more in sync with the harmonic motions

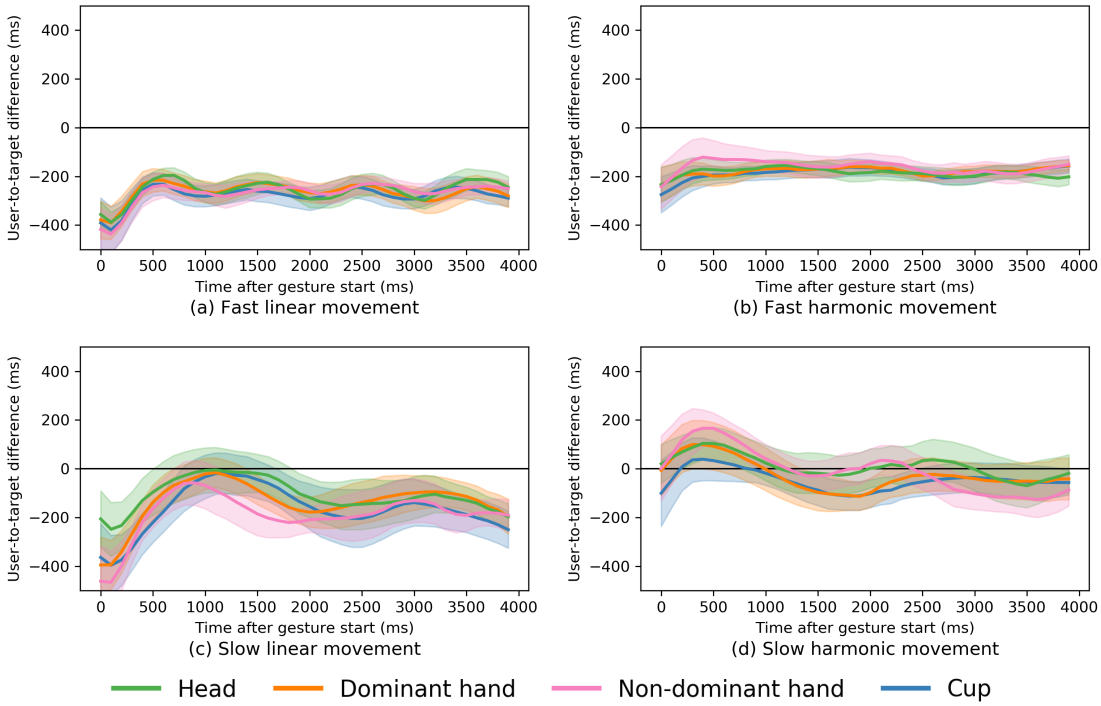

Figure 5.8: Synchronisation between the user and the target for the vertical shape, measured in 100ms bins. Shaded areas show the 95% confidence interval. Negative values indicate the user is lagging the target, positive values indicate the user is leading.

than with linear. For fast horizontal movements the average difference was 71 ms (Linear: -242 ms, Harmonic: -171 ms), and the difference doubled for slow movements at 147 ms (Linear: -139 ms, Harmonic: +8 ms). We see a similar trend for the vertical movement with an average difference of 81 ms for fast (Linear: -256 ms, Harmonic: -175 ms) and 101 ms for slow speeds (Linear: -127 ms, Harmonic: -21 ms). In all cases users appear to be able to synchronise more closely with the harmonic targets. Finally, similarly to the slow square, we observe oscillations in the signals for the linear movements types at all speeds, and for the harmonic movements at slower speed. The oscillations for the linear movement types align with the turning points of the lines, similar to those seen in the slow square, and are indicative of the harmonic movements we naturally make. For the horizontal movement, we see much larger range of values for the aggregated user-to-target differences at both fast (Linear: 114ms, Harmonic: 42ms) and slow (Linear: 215ms, Harmonic: 109ms) speeds. The range of differences of the vertical movement is nearly doubled for the fast linear movement (68ms) compared with harmonic (35ms), however there is little difference between the ranges for slow movements (Linear: 176ms, Harmonic: 173ms).

5.2.3 User Preferences

When asked about their overall preference across all movement conditions all of the users preferred the faster harmonic targets using their dominant hand, except one who preferred the fast cup movement (P2) - who stated that it "felt nice having something in my hand" and was "satisifying" to perform. For shapes, five preferred the horizontal movements, two preferred the circle, and one preferred the square. One of the main comments participants made regarding their preferences for the faster movements was that if felt "more natural", and less straining. Interestingly, one user commented that they thought the fast movement "saved time" despite all trials lasting eight seconds - suggesting that the perception of time taken was affected by the target speed. People noted that harmonic motion also felt "more natural" and "fluid". P1 noted that the har-

monic motion felt like it gave them "mini-breaks" when doing the motion, and that this was more pronounced during the faster movements. Participants felt that they were always "catching up" with the linear movements, and that it was tiring, whereas the harmonic motion felt "in sync". P3 described the harmonic motion as "dancing" with the target. Users commented that it felt more natural to "point" with the hands compared with the head, and chose the dominant hand because they use it more. Although most users preferred their dominant hand overall, some noted that for prolonged usage they would prefer the head because it involved less effort.

When asked about the dominant hand, all users preferred the fast harmonic movements. The most popular choice of shape was the circle (3), followed by the horizontal line (2) and square (2), with one person preferring the vertical line. For the non-dominant hand all the users preferred the fast harmonic movements, although one user said the they preferred the slow speed equally (P7). The circle was once again the most popular choice (3), followed by the horizontal (2) and vertical (2) lines, with one person preferring the square. For cup movements, all participants preferred harmonic movements except P4, who was also the only participant to prefer the slower movements. Three participants preferred the horizontal line, followed by the vertical line (2) and the circle (2), and only one participant preferred the square. With the cup one participant stated that they preferred the vertical movement because it was like "drinking or cheering", whereas P4 stated that vertical movements were harder because they had to "lift" the object.

The head movement proved significantly different in terms of preferences. Five people preferred the faster movement, two preferred the slower movements and one said it was dependent on the shape (P2). P2 said that there was little difference for slow and fast for the circle and the lines, however they preferred the slower movements for the square. Six participants preferred the horizontal movement, with two preferring the circle. All the participants preferred the harmonic movement with the exception of P4 who preferred the linear movement with the head. Two participants commented that the square movement was particularly difficult to perform with the head (P2 and P6). P6 noted that with horizontal movements it was easier to maintain gaze on the target whilst performing horizontal movements compared with vertical, a similar sentiment shared by P4 who stated they "lose the target" with vertical movements. This also indicates users performing (relatively) large head movements. Interestingly, P2 and P6 had the largest average head movement during the trial.

5.2.4 Discussion

Our analysis highlights the strengths of motion correlation as an approach which abstracts from specific body parts and provides insights across the different movement conditions. Across all shapes, speeds and movement types there is little to separate type of input, especially dominant and non-dominant hands. Unlike other manual tasks in which one hand may outperform the other, both hands are equally adept at following the motion of a target, making it a suitable technique for spontaneous interaction with either hand.

Our findings provide interesting insights into interface design, showing movement conditions are affected by targets differently. For faster speeds we observed similar user-to-target difference between the circle and square for the fast targets, and in general, we see little difference for one dimensional line-based movements across all the movement conditions. In Chapter 3.3 we observed users struggling with the fast circular movements when using the head, and in this study we observe users struggling with the fast linear square movement with the head, however there were no issues with the harmonic movement. The head is compelling because it can be used in settings where the hands are either unavailable or busy with other tasks, and it is quicker to start gesturing with compared to the hands. Thus, harmonic movements should be used for applications in which users may be more likely to use head movements. Alternatively, it may be beneficial to use slower two-dimensional movements, or one-dimensional

movements which were more generally preferred. We also note the similarity between the fast two-dimensional square and slow one-dimensional targets, due to the same edge length. Future work could shed light on how users match motion with faster moving one-dimensional targets.

We also observe that users prefer faster targets, but can synchronise more closely with slower. In contrast to our findings in Chapter 3.3 where there was a 50/50 split in preferences for slow versus fast, all participants preferred the faster movements. This could be for two reasons: the first is that the sample size was smaller and therefore we did not encounter those who preferred slower movements. Another is that the previous study was an interactive task, and thus the performance of the algorithm may have impacted participants' preference, i.e. slow movements had a higher success rate. From the data, we observe that the faster movements provided more stable readings across participants compared to slower movements across each shape. For both one-dimensional shapes and the circle we see large variability which could indicate that the movements were too slow for participants. Previous research on motor behaviour has shown that users avoid moving slowly, and instead attempt to move closely to their preferred tempo [264]. This could explain the variability as users overtake and catch up the target due to their desire to move more quickly, and is also reflected in participants' preferences for faster movements.

Despite their preference for faster targets, users were more closely synchronised with slower targets, albeit with more variability. For circle movements we observe evidence that slower target movements result in lower times to start the gesture (although not statistically significant). In addition, if we subtract the estimated difference between fast and slow movements for line-based movements (based on the start movement being equally distributed along the edge prior to the first detectable vertex) the starting times for slow movements are less than those for fast movements. Whereas there is no physical explanation for this phenomena, it could be due to less cognitive load in detecting the target movement and deciding how to coordinate one's body movement to "intercept" and catch the target.

Participants preferred one-dimensional shapes, but two-dimensional are more robust. The one-dimensional movements are smaller than their two dimensional counterparts, which could have contributed to them being the preferred choice because they were both cognitively and physically easier to follow. However, there are similar properties between the fast one-dimensional lines and the slow square. The edge length of these shapes are the same (1 s), but interestingly users are more in sync with a slow square than the fast lines. This could be due to the 2D nature of the square movements where it is visually easier to distinguish the corner to synchronise with. For both one- and two-dimensional shapes users unanimously preferred harmonic movements overall, which have shown to have distinct advantages in both user preference and users' ability to synchronise with the target.

Finally, the results from analysing how we are able to follow the motion of a moving target provides interesting insights to inform algorithm design. Firstly, users are never exactly in sync with the target and may both lead or lag the target. Whereas harmonic movements demonstrate closer synchronisation with the target, there is always an offset that must be taken into account which varies across users. Secondly, the leading or lagging of the target varies over time, and in the case of slow movements we see evidence of users overshooting the target and leading it, prior to dropping back. This was most apparent for circular gestures, where evidence in the sensorimotor synchronisation literature would suggest that because the circle lacks any perceptible events, synchronisation would be more variable and show less error correction [254]. Harmonic movements for faster targets also demonstrate much less variability than their linear counterparts, which may provide a more stable signal when comparing user and target trajectories. The variability is also dependent on the type of input used. Uniquely, the head leads the target for both circular movements prior to synchronisation, whereas hand-based movements lag the target. Work in psychology has shown that there are two dominant techniques when intercepting a target, one is to catch the target from behind, whereas the other is to

intercept from the front [179]. This has implications for algorithm design, as one must be aware of users both leading and lagging the target in the initial movement phase which may vary when a user achieves synchronisation.

5.3 Algorithms

Given the insights from how well users can synchronise with targets, we explore the sensitivity and specificity of algorithms for motion correlation based on the matching of user and target trajectories. We perform an extensive search of the parameter spaces of eight algorithms, both from the motion correlation literature [273, 40, 269], and from the wider time-series data mining and matching literature [282, 54].

Detecting whether a user's motion coincides with a target's movement involves two stages: the first is determining a measure of similarity (or dissimilarity) between the two movements; and the second is determining whether that similarity passes a threshold to be able to say that the user's movement intentionally matched the target's. Algorithms must be sensitive enough to detect when users are matching the motion of a moving target, whilst being robust enough to reject accidental motions which are not intended for interaction. In the case of motion correlation there are a number of constraints that approaches must satisfy. The first is the requirement for the approach to be real-time. Secondly, one must match the spatiotemporal properties (shape, speed, direction and phase) of both user and target trajectories. The matching of these properties allows for simultaneous motions moving in the same direction at the same speed to be presented to a user and successfully differentiated. Finally, the approach should be scale-invariant. Both user and target trajectories should not have to be in the same coordinate space for matching.

Broadly speaking, there are two categories for comparing time-series trajectories: lock-step and elastic [282], see Figure 5.9. Lock-step measures compare the *i*th data point from one trajectory with the *i*th data point from another trajectory. This one-to-one rigid mapping can be sensitive to noise and misalignments in time, however it implicitly assumes that the trajectories are temporally aligned with similar speeds - an inherent assumption of motion correlation. In contrast, elastic measures allow for one-to-many and one-to-none comparisons to find an optimum match. This can be used to find similar trajectories with an offset (such as the case with leading/lagging), but also in the case where the speeds of the trajectories vary over time, e.g. due to compression.

5.3.1 Lock-step Measures

Lock-step measures compare trajectories based on one-to-one mappings. They are computationally simpler than elastic measures and are the only type of algorithm that have been explored

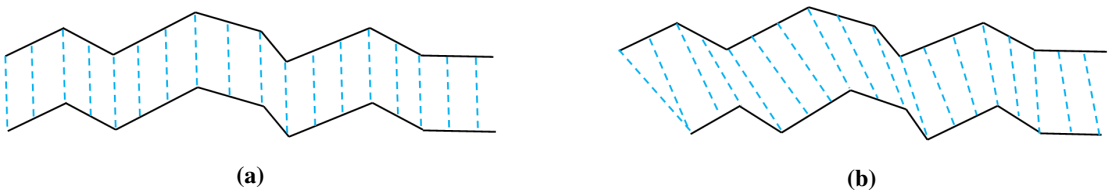

Figure 5.9: Illustrations of (a) Lock-step measure showing the rigid one-to-one mapping, and (b) Elastic measure showing one-to-many mapping against a signal that has been compressed and shifted. Note we show them separated on the y-axis for illustration, however these would usually be normalised.

in the motion correlation literature. We further categorise lock-step measures into correlation-based and distance-based metrics.

Correlation Measure

Correlation measures report the relationship between two variables as input, where +1 implies a positive correlation, -1 implies a negative correlation, and 0 implies no correlation. Earlier work on motion correlation techniques utilised the Pearson correlation coefficient, a measure of the linear correlation between two axis [73, 273, 70]. The Pearson correlation coefficient, r_{xy}, is defined as:

$$r_p(A,B) = \frac{\sum_{i=1}^{n}(a_i-\bar{a})(b_i-\bar{b})}{\sqrt{\sum_{i=1}^{n}(a_i-\bar{a})^2}\sqrt{\sum_{i=1}^{n}(b_i-\bar{b})^2}} \tag{5.3}$$

Where a and b are one-dimensional time-series of length n with paired values, $\bar{a}$ is the mean value of a, and $\bar{b}$ is the mean value of b.

In one of the earlier works on motion correlation, Fekete et al. used the Pearson correlation by summing the value of both axis [73], however they found this was outperformed by other distance-based metrics. Vidal et al. introduced an alternate approach by ensuring the minimum of the two axes Pearson correlation were above a threshold, thus ensuring that both axes were correlated with the respective axis of the target [273]. Carter et al. identified a flaw in the algorithm when comparing line-based movements, because the coefficient is ill-defined with no movement in one of the axis [40]. They proposed to rotate the data by calculating the PCA in order to maximise the variance in both axis prior to calculating the Pearson coefficient. In Chapter 3 we identified that for body movements the Pearson correlation does not take into account the aspect ratio of the movement as the axes are calculated independently. We demonstrated that by incorporating a model-fitting approach we minimised false positives, however this was limited to circles and each new shape would require a relevant model.

Whereas the Pearson's correlation coefficient compares the linear relationship between two time series, i.e. it looks for a proportional change, the Spearman's rank correlation coefficient looks at the monotonic relationship, i.e. they change together but not necessarily at a constant rate. Although ordinal data is most commonly used as input to the Spearman correlation coefficient, the Spearman correlation coefficient is non-parametric and, unlike the Pearson correlation coefficient, does not assume the data is normally distributed. The Spearman rank correlation coefficient is calculated in the same way as the Pearson, except it uses the ranks instead:

$$r_s(A,B) = \frac{\sum_{i=1}^{n}(R(a_i)-R\bar{(a)})(R(b_i)-R\bar{(b)})}{\sqrt{\sum_{i=1}^{n}(R(a_i)-R\bar{(a)})^2}\sqrt{\sum_{i=1}^{n}(R(b_i)-R\bar{(b)})^2}} \tag{5.4}$$

Where $R(a_i)$ is the rank of the ith value of a and $R\bar{(a)}$ is the mean rank of a.

Minkowski Distance Measures

Distance-based metrics measure the similarity of two time-series based on the distances between pairs of points. The most common distance metrics are the family of Minkowski distances, defined between a pair of points, a and b, as:

$$d_m(a,b) = \left(\sum_{i=1}^{d}|a_i-b_i|^p\right)^{\frac{1}{p}} \tag{5.5}$$

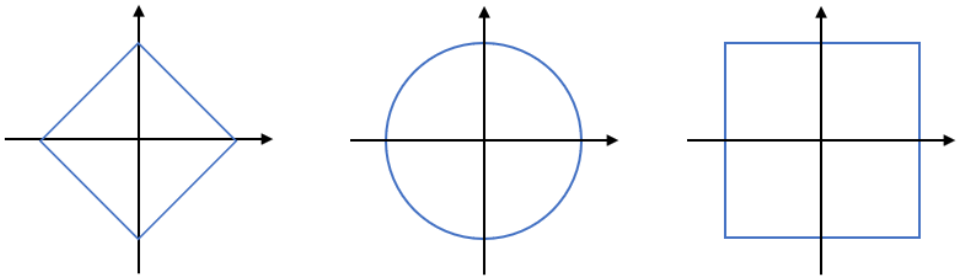

Figure 5.10: Visual illustration of the most common Minkwoski distances showing all points that are unit distance from the centre for Manhattan (left), Euclidean (centre), and Chebyshev (right).

Where d is the number of dimensions (in our case two), and p is the order of Minkowski distance. The most commonly used p-values correspond to common distance measures of Manhattan ($p = 1$), Euclidean ($p = 2$) and Chebyshev ($p = \infty$), see Figure 5.10 for a visual representation. The Euclidean distance is the most intuitive of these measures, and the square of the standard euclidean distance is often used in loss functions (e.g. root-mean squared error) because it places greater emphasis on larger errors. Once a distance has been found for each pair of points between the trajectories, the results must be agglomerated. The most common type of agglomeration functions are the sum, mean, or root mean of all paired values. For example, the similarity measure based on the average Euclidean distance between each pair of points in the trajectories A and B is calculated as:

$$d_m(A,B) = \frac{1}{n}\sum_{i=1}^{n} \|a_i - b_i\| \tag{5.6}$$

where n is the number of points in each trajectory, and a_i is the ith point in trajectory A.

In the motion correlation literature, Velloso et al. introduced "2D correlation", and demonstrated that it outperformed the Pearson correlation-based methods for smooth pursuit eye movements [269]. Although motivated by the R^2 coefficient of determination, the algorithm as described in [269] is not a conventional measure of correlation the values are not confined between +1 and -1 like the aforementioned correlation metrics. The method utilises the Euclidean distance (as opposed to the sum of squares), after the trajectories have been normalised according to their z-score. The euclidean distance between each pair of points in both user and target trajectory is summed, prior to being normalised by the sum of the euclidean norms of the user trajectory.

5.3.2 Elastic Measures

Elastic measures remove the one-to-one constraint of the lock-step techniques, instead allowing for comparisons of one-to-many or one-to-none. The ability to stretch or compress one time series relative to the other reduces the effect of local time distortions, such as accelerations or decelerations.

Dynamic Time Warping

Dynamic Time Warping (DTW) allows local time warping in a non-linear manner. The output of DTW is the smallest sum of absolute distances required to "warp" one trajectory to the other. Prior to becoming a staple of the data mining community, DTW was traditionally used in the speech recognition community and has found application in a wide variety of domains, from gesture recognition [217] to real-time query-by-humming systems [305]. Originally introduced

for time-series by Berndt and Clifford, DTW can be computed using dynamic programming with $O(nm)$ complexity, and is defined as [25]:

$$DTW(A,B) = f(n,m) \tag{5.7}$$

$$f(i,j) = d(a_i,b_j) + \min \begin{cases} f(i,j-1) \\ f(i-1,j) \\ f(i-1,j-1) \end{cases} \tag{5.8}$$

Where $d(a_i,b_i)$ is a distance measure (e.g. Euclidean), $i = 1,\ldots,n$ and $j = 1,\ldots,m$. There are several constraints that can be placed to reduce the search space of warping paths. Monotonicity ensures that the matched points are ordered with respect to time, and continuity ensures that steps are confined to neighbouring points. The amount of warping between two time series can be constrained, not only reducing the computational cost required, but it has also been shown to improve similarity matching [282]. There has also been concern about the running time of DTW, however with lower bounding techniques and ever improving hardware, DTW has found application in many real-time applications [209].

The boundary conditions of the algorithm are commonly constrained to the endpoints, i.e. the first index of one time series must be matched with the first of the other, and the same applies to the last indices. This "anchoring" of start and end points was originally proposed to simplify the search space of warping paths. By relaxing these constraints, we can instead look at subsequence matching, first proposed in [180]. For our purpose this can be advantage as it removes the need for both user and target trajectories to be of the same length, instead we can search a larger portion of the target trajectory to account for the user trajectory to account for the leading/lagging element.

Longest Common Subsequence

The Longest Common Subsequence (LCSS) is based on the concept of *edit distance* and was originally used in the natural language processing community to find the maximum substring that is common between two strings. Vlachos et al. adapted the fundamental concepts of LCSS to time series similarity matching by introducing a parameter, ε [275]. If two data points are within ε they are thought of as equivalent, akin to being the same character in the string. In contrast to DTW, not all elements of both sequences must be matched which can make it more robust to outliers. Similar to DTW, a threshold that limits the amount of warping can be used to reduce computational complexity and improve performance, and we can also match subsequences.

The longest common subsequence of two trajectories, $LCSS_{\delta,\varepsilon}(A,B)$, is defined as:

$$\begin{cases} 0 & \text{if } A \text{ or } B \text{ is empty} \\ 1 + LCSS_{\delta,\varepsilon}(H(A),H(B)) & \text{if } d(a,b) < \varepsilon \text{ and } |n-m| \leq \delta \\ \max(LCSS_{\delta,\varepsilon}(H(A),B), & \\ \quad LCSS_{\delta,\varepsilon}(A,H(B))) & \text{otherwise} \end{cases} \tag{5.9}$$

Where H(A) represents the head of the trajectory, $H(A) = (a_1,\ldots,a_{n-1})$, ε is the threshold for determining whether two points are deemed "equal", and δ constrains the time warping. The similarity between two trajectories based on the LCSS is:

$$S_L(A,B) = \frac{LCSS_{\delta,\varepsilon}}{\max(n,m)} \tag{5.10}$$

Geometric Similarity Measures

Often used in the computer vision and graphics communities, geometric similarity measures measure the difference between two geometric shapes. The Hausdorff distance is used to match a set of points against a template shape [110]. It is defined as the maximum of the minimum Euclidean distances between a set of points to the nearest point in the other set:

$$d_H(A,B) = \max_{\forall a \in A} \min_{\forall b \in B} \|a-b\| \tag{5.11}$$

Hausdorff is an asymmetric distance, and the bi-directional Hausdorff distance is calculated as:

$$D_H(A,B) = \max(d_H(A,B), d_H(B,A)) \tag{5.12}$$

The Hausdorff distance does not take the order of the points into account. Another geometric measure is the Fréchet distance which takes into account the order of both sets of points [66]. Intuitively, this can be thought of as a man walking along one trajectory and a dog on the other. The Fréchet distance is the shortest leash that would be required assuming both the man and the dog can only walk forwards (at varying speeds) along the trajectories.

5.3.3 Normalisation

One of the compelling properties of motion correlation techniques is scale invariance between input and output spaces. That is the user and target's motions do not need to have the same scale, or be in the same coordinate space for a match to be detected. From an algorithmic perspective, scale invariance is inherent to the correlation-based algorithms, but for all others we can pre-process the data to remove the effect of scaling through normalisation. In this work, we investigate three normalisation techniques:

Z-score normalisation

Z-score normalisation transforms the data so that it has zero mean and unit variance. We use the maximum standard deviation for both axis to maintain the aspect ratio between the x- and y-axis:

$$\tau'_x = \frac{\tau_x - \bar{\tau}_x}{\max(\sigma_x, \sigma_y)} \tag{5.13}$$

where τ'_x is the transformed x-values of the trajectory, τ_x is the original x values, $\bar{\tau}_x$ is the mean of the original x-values, and σ_x the standard deviation of the x values. The y values would be transformed by replacing y for x.

Min-max normalisation

Min-max normalisation transforms the data so that all values are within the range $[0,1]$. To maintain the aspect ratio of the trajectory we use the axis with the largest range:

$$\tau'_x = \frac{\tau_x - \min(\tau_x)}{\max(\max(\tau_x) - \min(\tau_x), \max(\tau_y) - \min(\tau_y))} \tag{5.14}$$

Where τ'_x is the transformed x-values, τ_x the original x values, $\min(\tau_x)$ the minimum x value and $\max(\tau_x)$ the maximum x value.

Table 5.3: Table showing the algorithms and parameters used for the evaluation

Algorithm	Parameters	Parameter Combinations
Procrustes	Buffer size; Sweep	18
Pearson Correlation	Buffer size; Sweep; Rotate	36
Spearman Correlation	Buffer size; Sweep; Rotate	36
2D Correlation	Buffer size; Sweep; Normalisation	54
Hausdorff	Buffer size; Sweep; Normalisation	54
Minkowski	Buffer size; Sweep; Normalisation; Distance metric	216
Dynamic Time Warping	Buffer size; Padding; Normalisation; Distance metric; Window	648
Longest Common Subsequence	Buffer size; Padding; Normalisation; Distance metric; Delta; Epsilon	2592

Procrustes analysis

Procrustes analysis determines the optimal linear transformation consisting of translation, reflection, orthogonal rotation and scaling that transforms a set of points A to another set of points B, using the sum of squared errors as the goodness of fit criterion according to:

$$Minimize \left\| c(\tau' + \mathbf{v}\mathbf{x}^T)\mathbf{R} - \tau \right\|^2 \tag{5.15}$$

where τ and τ' are $(n \times 2)$ matrices that contain the co-ordinates of the trajectories of n data points for the target and user, respectively. n is the number of data points found in both trajectories. c represents the uniform scale factor ($c \in \mathbb{R}$). $\mathbf{R}$ is a (2×2) orthogonal rotation matrix, which handles rotation and reflection. x is a (2×1) vector which translates data points of P' with v, while $v = (11 \ldots 1)^T$ is a $(n \times 1)$ vector. By extracting the translation ($\mathbf{x}$), rotation ($\mathbf{R}$)), and scale (c) we can transform one set of points to optimally align the user's trajectory with the target trajectory. Although not a traditional normalisation procedure, we leverage its ability to align two trajectories before determining a similarity score based on different distance-metrics. Note that due to the rotation aspect the Procrustes algorithm can align trajectories temporally, as well as spatially.

5.3.4 Parameter Selection

In order to evaluate different matching algorithms on the dataset collected in Section 5.1 we performed an extensive search of the parameter space across eight algorithms, see Table 5.3. In total we explored a total of 3,654 different parameter combinations. We initially included the Fréchet distance using the algorithm proposed by Eiter and Mannila [66], but found it took too long for all but the smallest buffer sizes, because the algorithm is polynomial-time, and therefore do not include this in our analyses [34].

Lock-step

Based on our findings in the previous section we introduce a sweep through different starting points to account for a user leading/lagging, or due to sensor characteristics such as input lag. For the lock-step based approaches we introduce a "sweep" parameter which calculates the similarity between user and target trajectories at different temporal offsets. This is analogous to cross-correlation or convolving the signals, except we call it "sweep" because mathematically we neither cross-correlate or convolve the signals. For example, for a sweep of 10, the user trajectory starting at index i is compared to the target trajectory in the interval $[i-10, i+10]$, and the similarity threshold of the best match is returned along with the respective target index (e.g. lowest value for distance metrics, highest value for correlation-based metrics). We search

the whole interval because as noted in the previous section, there is variability in the amount of time a user leads or lags the target. For the Minkowski distance measure, we take the average of the distances over the buffer, so that it is comparable across buffer sizes.

We use the Procrustes algorithm for both normalisation and as an algorithm itself. For this, we use the output of the Procrustes analysis - the residual sum of squared errors, normalized according to a measure of the input trajectories. Although the Procrustes algorithm can align the signals temporally (due to the rotation component) it is still a lock-step algorithm because it performs one-to-one mappings.

Elastic

Knowing that there is a lagging or leading element involved, and that this may dynamically change over time we use subsequence variant matching, whereby anchor points are relaxed. However, we enforce that all points in the user's trajectory must be matched to at least one point in the target's trajectory in the DTW variants. For both LCS and DTW we use a "padding" parameter, which uses a larger buffer size for the target trajectory relative to the user trajectory. We include this to account for when the user's movement is leading or lagging a target. Consider a user and target trajectory of size 30, where the target is leading the user by 10 frames. If both user and target trajectories are optimally aligned spatially according to their matching indices, then only 20 frames of the trajectories are overlapping. If however, we pad the target trajectory by 10 frames at each end, increasing its trajectory size to 50 then we would expect to match all of the 30 frames in the user's trajectory.

We also include, δ, a parameter that constrains the maximum amount of time warping allowed in both LCS and DTW algorithms, such that when comparing index i from trajectory A with index k from trajectory B, the following must hold true:

$$\delta \geq |i-k| \tag{5.16}$$

In the DTW literature this is known as the Sakoe-Chiba Band [226], and it has two purposes. The first is that it constrains the amount of warping possible and can lead to more accurate results by preventing pathological warping (where a small subset of one trajectory matches to a large portion of the other) [226]. The second is that it can reduce computational complexity by reducing the number of calculations required. The LCS algorithm also requires the epsilon (ε) parameter which is the value used to determine whether two data points are a classified as a match. When there is no value of epsilon supplied (i.e. $\varepsilon = \text{None}$), it is calculated as the smallest standard deviation between the two trajectories as suggested by Vlachos et al. [275].

Posthoc Window

Inspired by hand gesture recognition, Carter et al. used the notion of a bi-threshold posthoc filter which has two thresholds: an initial threshold which must be initially met; and a second lower threshold which the match must remain above for a predetermined amount of time [185]. Velloso et al. demonstrated that a single level posthoc threshold, which looks to see if the signal is above the threshold for N_p frames, outperformed the bi-level threshold for use with smooth pursuit eye movements where the signal is noisier. In [269] the authors also suggested that "maintaining a high correlation would lead to user fatigue and consequently a drop in the similarity metric" for body movements. However, our analysis of body movements in the previous section shows this not to be the case. Instead what we see is fluctuations in a user's ability to synchronise with a target over time. In this work we forego the bi-level threshold and instead look at two single-level posthoc filters of 15 frames ($\approx 0.5s$) and 30 frames ($\approx 1s$), in addition to the case of no posthoc filter.

Table 5.4: Values for each of the parameters used in the evaluation

Parameter(s)	Values
Buffer size (Bs)	15, 30, 45, 60, 90, 120
Sweep (Sw); Padding (Pa)	0, 10, 20
Rotate (Ro)	True, False
Normalisation (N)	Minmax, Z-score, Procrustes
Distance metric (Dm)	Manhattan, Euclidean, Chebyshev, Squared Euclidean
Delta (δ)	∞, 10, 20
Epsilon (ε)	None, 0.1, 0.2, 0.3

5.3.5 Procedure

Our goal is to find the algorithm and its parameters that maximises the true positive rate, whilst minimising the false positives and activation time. Previous work has shown the properties of the target movement affect the optimum parameters, for example in Chapter 3 we demonstrated that the buffer size is dependent on the speed of a target in the case of circular movements. We also hypothesise that the optimum algorithm and parameters for one shape may not translate to all shapes. We therefore look to extract the optimum algorithm and its parameters for each speed and shape combination. We look across all movement conditions to perform the matching, as one may not be able to distinguish which body part, or object, is responsible for a motion at the sensor level (e.g. using optical flow techniques).

We used a python framework to implement the described algorithms, using NumPy and SciPy libraries for algorithms where available. All the algorithms we describe and implemented are index-based. For computational simplification, we assume the captured frame rate of the device was constant at 30 fps as we captured the video under lab conditions, and thus 15 frames represents 0.5 s. For real-world deployments, frame rates of capturing devices may vary and therefore time-based algorithms may be more appropriate. For example, the TraceMatch and MatchPoint systems use time-based windows, because the frame rate of a web camera may vary depending on the amount of light available. Due to the large amounts of data being processed and generated, we run the framework on a server of 56 CPUs running at 2.40GHz per CPU over a number of weeks.

We begin by calculating the true positives. For each trial we run each parameter combination for each algorithm against each true positive sample, i.e. each 8 second video of a user attempting to match the motion of the on-screen target. For each true positive sample, we calculate the similarity measure at each time stamp, on a rolling window basis, only once the buffer is full with data, see Fig 5.11. For example, an 8 second trial at 30 frames per second will have 240 frames. Assuming a buffer size of 30 means there will be 210 frames that will have an associated similarity measure. We found three trials that had erroneous data samples missing so we omit these for our analyses.

We then investigate the false positive rate of each algorithm-parameter combination. We generate "dummy" target movements for each shape-speed combination which are used as the targets for each video file of a user performing the semantic and spatial gestures. This is to simulate targets moving in the background whilst the users performed the gesture, although we note that these are simulated and users did not see any moving target when performing the semantic and spatial gestures. Due to such an extensive search space, we found it was impractical to run all parameter combinations against all the false positive data. We therefore extract the false positive rate in two stages. The first, calculated for all parameter combinations, uses the first one minute of false positive data from each participant using two simulated targets for each shape and speed

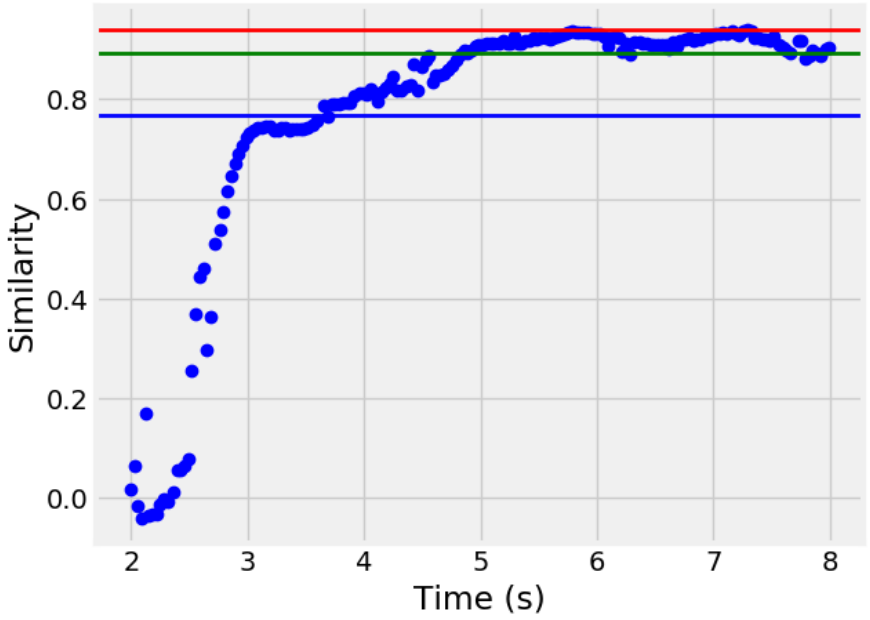

Figure 5.11: An example of the similarity calculations for one trial using the Pearson correlation with buffer size 60, PCA rotation, and zero sweep. The red line represents the maximum value achieved, the green the median, and the blue the mean.

combination with a 180°offset. This provides us with a lower-bound of the false positive rate of each algorithm-parameter combination, and we can discard any combinations that encounter false positives during this stage. To further reduce the amount of data processing for the false positive detection we only calculate a similarity measure when the average movement of the user's trajectory is greater than 1 pixel per frame. The second is a more verbose

Once we have similarity measures for all true positive and the first-cut false positives, we determine the appropriate thresholds to use. For this, we calculate the thresholds required to achieve a true positive rate of 100% (TP100) without a posthoc window, and for posthoc windows of 15 (≈0.5 s), and 30 (≈1 s). For example, when looking at the Pearson correlation algorithm we extract the maximum value for each true positive trial, see Figure 5.11, and then sort the values in ascending order. We can then determine the threshold required to achieve a 100% (minimum value) true positive rate or any arbitrary true positive rate (e.g. 95% TP rate = 5th percentile). We extract parameters for each parameter set, and for each shape-speed combination of target. Using these thresholds we calculate the corresponding false positive rate, and activation time (the time of the first detection) of each trial.

Once the true positive rate, false positive rate, and activation time has been calculated for each shape-speed combination, we rank the parameter combinations for each algorithm. This is achieved by extracting the top three parameter combinations by minimising the false positive rate, and then in the event of a tie minimising the activation times. Once we have the top parameter combinations, we run these on the full false positive data with four simulated targets at 90°offsets. This gives us a better indication of how well the algorithms (and motion correlation in general) are at rejecting false positives.

5.3.6 Results

Table 5.5 shows that we find an algorithm-parameter combination that yields a 100% true positive rate with 0% false positives for every shape-speed combination. This is achieved using both lock-step and elastic algorithms, with six of the eight algorithms featuring. Investigation of the

Table 5.5: Top performing algorithms that achieve a 100% true positive rate whilst minimising false positives, and ranked according to activation time for the circle (C), square (S), horizontal (H), vertical (V), and their harmonic (h) variants.

Shape	Speed	Algorithm	TP Rate	FP Rate	Activation time (s)	Buffe size	Threshold	Parameters	Posthoc
C		LCS	100	0	2.30	30	1	Dm: sqeuclidean; No: procrustes; ε: 0.3; Pa: 10.0; δ: 20.0;	15
S		Correlation2D	100	0	2.62	30	0.661	Sw: 20.0; No: procrustes;	30
S(h)		Correlation2D	100	0	2.27	30	0.705	Sw: 20.0; No: z-score;	15
H	2	Minkowski	100	0	3.03	45	0.187	Dm: chebyshev; Sw: 20.0; No: z-score;	15
H(h)		Spearman	100	0	3.23	45	0.982	Sw: 20.0; Ro: True;	15
V		Minkowski	100	0	2.84	45	0.199	Dm: cityblock; Sw: 20.0; No: z-score;	1
V(h)		Spearman	100	0	3.99	90	0.946	Sw: 20.0; Ro: True;	1
C		LCS	100	0	2.86	45	0.733	Dm: chebyshev; No: procrustes; ε: 0.2; Pa: 0.0; δ: inf;	15
S		LCS	100	0	3.20	60	1	Dm: sqeuclidean; No: z-score; ε: 0.3; Pa: 20.0; δ: inf;	1
S(h)		LCS	100	0	3.17	60	1	Dm: sqeuclidean; No: z-score; ε: 0.3; Pa: 20.0; δ: inf;	1
H	4	Procrustes	100	0	4.89	120	0.066	Sw: 20.0;	1
H(h)		Spearman	100	0	5.43	120	0.973	Sw: 20.0; Ro: True;	15
V		Pearson	100	0	3.59	60	0.967	Sw: 10.0; Ro: True;	15
V(h)		Pearson	100	0	3.60	45	0.969	Sw: 10.0; Ro: True;	30

Table 5.6: The false positive rate and activation time (in brackets) for each algorithm's best performing parameter combination which achieves a 100% true positive rate. Blue highlighted cells indicate the best performing algorithm for a given shape-speed combination, yellow indicate a non-zero false positive rate.

Shape	Speed	Proc	Pear	Spea	Corr	Mink	Haus	DTW	LCS
C		0.0 (2.44)	0.0 (2.65)	0.0 (2.74)	0.0 (2.33)	0.0 (2.44)	0.0 (2.53)	0.0 (2.83)	0.0 (2.30)
S		0.0 (3.09)	0.0 (3.15)	0.0 (3.51)	0.0 (2.62)	0.0 (2.86)	0.0 (3.25)	0.0 (3.31)	0.0 (2.67)
S(h)		0.0 (2.75)	0.0 (2.48)	0.0 (2.96)	0.0 (2.27)	0.0 (3.06)	0.0 (3.20)	0.0 (3.54)	0.0 (2.57)
H	2	0.0 (3.89)	0.0 (3.10)	0.0 (3.10)	0.0 (3.07)	0.0 (3.03)	8.3 (3.72)	2.1 (4.41)	2.1 (3.02)
H(h)		0.0 (4.24)	0.0 (3.26)	0.0 (3.23)	0.0 (4.01)	0.0 (3.55)	4.2 (3.98)	0.0 (4.56)	1.0 (3.40)
V		0.0 (4.03)	4.2 (4.47)	0.0 (5.64)	0.0 (4.05)	0.0 (2.84)	8.3 (3.85)	5.2 (5.12)	0.0 (3.43)
V(h)		0.0 (4.07)	0.0 (4.10)	0.0 (3.99)	2.1 (4.29)	6.2 (3.67)	8.3 (5.45)	3.1 (4.62)	4.2 (4.15)
C		0.0 (3.12)	0.0 (2.91)	0.0 (3.86)	0.0 (3.05)	0.0 (3.20)	0.0 (3.32)	0.0 (3.52)	0.0 (2.86)
S		0.0 (3.73)	0.0 (3.75)	0.0 (4.23)	0.0 (3.86)	0.0 (3.46)	0.0 (3.89)	0.0 (4.21)	0.0 (3.20)
S(h)		0.0 (3.19)	0.0 (3.23)	0.0 (3.50)	0.0 (3.45)	0.0 (3.17)	0.0 (3.68)	0.0 (3.85)	0.0 (3.17)
H	4	0.0 (4.89)	0.0 (4.90)	3.1 (4.71)	2.1 (4.82)	3.1 (5.32)	14.6 (4.83)	14.6 (3.62)	2.1 (4.34)
H(h)		14.6 (5.30)	1.0 (4.97)	0.0 (5.43)	5.2 (3.83)	7.3 (4.01)	14.6 (4.07)	12.5 (3.72)	10.4 (3.39)
V		12.5 (4.61)	0.0 (3.59)	1.0 (4.04)	0.0 (3.72)	2.1 (3.35)	35.4 (3.29)	5.2 (3.31)	2.1 (4.41)
V(h)		20.8 (3.57)	0.0 (3.60)	2.1 (3.48)	1.0 (3.06)	0.0 (3.62)	25.0 (2.71)	6.2 (5.45)	2.1 (3.21)

parameters shows how aligning input and output data both temporally and spatially is crucial in determining a good match. We see this reflected with the use of the sweep parameter which features in all of the top-performing lock-step algorithms. We observed higher thresholds for combinations in which the sweep parameter was used, demonstrating that it is able to match trajectories more closely. We also observe the Procrustes algorithm feature for both circle conditions, which is the only normalisation technique that aligns temporally because it rotates the trajectories relative to each other.

Table 5.6 shows how the parameters perform across all target conditions given a 100% true positive rate. Appendix A details all the results, including the parameter combinations required to achieve these results. For faster movements, we observe the Procrustes and Spearman algorithms can suppress false positives across all target conditions, followed by the Pearson, Correlation 2D, and Minkowski algorithms which achieve robust results across all but one of the fast conditions. For fast two-dimensional conditions all algorithms achieve robust results with zero false positives. Across the three fast two-dimensional conditions Correlation 2D has the lowest

average activation time (2.41 s), followed closely by LCS (2.51 s). For fast one-dimensional conditions the Minkowski-based algorithms achieve the best results over the linear movement types (2.93 s), whereas the Spearman algorithm achieves top results for the harmonic variants (3.61 s). Interestingly, despite harmonic motion being the preferred choice of movement type, it takes substantially longer on average to synchronise than their linear counterparts. We know from the previous section that users can synchronise more closely with harmonic movement, therefore we can infer that harmonic movement is more likely to occur as accidental motion and thus algorithms must be more robust.

For slow two-dimensional movements we once again observe robust results with all algorithms achieving zero false positives, yet LCS achieved the quickest activation time across all three conditions. From the previous section we observed that the slower movements exhibited more variability than the faster conditions, and LCS's ability to temporally warp the trajectories may be advantageous. Having said that, we do not see the same level of performance from DTW which is outperformed by most of the lock-step measures. The disparity between elastic measures could be due to LCS being more robust to "noise" in the matching process [275]. In contrast, the slow one-dimensional movements do not show the same level of robustness to false positives. According to Table 5.5, both the slow horizontal conditions in particular show much larger activation times than the other shapes. This is indicative that they are much more likely to suffer from false activations than their vertical counterparts and as a result they require much larger buffer sizes ($\approx$ 3 s). According to Table 5.6 there are very few algorithms that can robustly match the slow one-dimensional trajectories. This suggests that one must be much more careful to select the correct algorithm when designing interfaces with slow one-dimensional movements.

The shapes which were quickest to detect for both slow and fast targets were the circle (fast: 2.3 s, slow: 2.86 s) and harmonic square (fast: 2. 27 s, slow: 3.17 s). We see a marked improvement over the activation times for the circle in contrast to Chapter 3.3 (fast: 3.46 s, slow: 3.66 s). We also note the differences between fast and slow targets (circle: 0.56 s, linear square: 0.58 s, harmonic square: 0.9 s), corresponding to increased buffer sizes of 0.5 s, 1.0 s, and 1.0 s respectively. This contradicts our findings in the previous section which showed how participants were more in sync with slower moving targets, and in some cases started gesturing sooner – our results here demonstrate that faster targets are much quicker to detect. The additional buffer size, and hence time to detect, could be required in order to capture enough salient information of the shapes to ensure they are robust to false positives. Note how buffer sizes of 60 correspond to half the square for the slow shape, and 30 for the fast shape (fast linear actually required a buffer size of 45).

Rather than minimising false positives to zero, we also investigate a more "relaxed" approach by allowing a maximum of 5% false positive rate, see Table 5.7. As a result of relaxing the false positive rate we observe all activation times decreased across all shapes and speeds. This is most notable in the case of worst performing conditions from Table 5.5 - the slow horizontal movements (linear: -1.08 s, harmonic: -1.55 s) and the fast harmonic movements (horizontal: -0.67 s, vertical: -0.82 s). We also note the lack of posthoc windows in all but the slow vertical conditions, in contrast to Table 5.5 where the majority of top parameters feature the posthoc window. This demonstrates the effectiveness of the posthoc window at reducing false positives, and also suggests those most affected are indeed more likely to suffer from false positives.

5.3.7 Discussion

Our results demonstrate the robustness of motion correlation as an input technique. For all target conditions we find an algorithm that achieves 100% true positive rate without any false activations. This includes the more problematic cases such as the fast head movements with circles and linear squares, which were problematic for some participants in Chapter 3 and in this

Table 5.7: Algorithms with the lowest activation time that achieve a 100% true positive rate and a maximum 5% false positive rate for the circle (C), square (S), horizontal (H), vertical (V), and their harmonic (h) variants. Δ T represents the difference in activation time between these algorithms and the top-performing algorithms in Table 5.5.

Shape	Speed	Algorithm	FP Rate	Activation time (s)	Δ T	Buffer size	Threshold	Parameters	Posthoc
C		LCS	2.1	1.94	-0.36	30	1	Dm: chebyshev; No: procrustes; ε: 0.3; Pa: 10.0; δ: inf;	1
S		Correlation2D	4.3	2.24	-0.38	30	0.791	Sw: 10.0; No: z-score;	1
S(h)		Minkowski	3.1	2.25	-0.02	45	0.366	Dm: euclidean; Sw: 10.0; No: z-score;	1
H	2	Spearman	2.1	2.53	-0.5	45	0.986	Sw: 20.0; Ro: True;	1
H(h)		Spearman	2.1	2.56	-0.67	45	0.987	Sw: 20.0; Ro: True;	1
V		Correlation2D	3.1	2.81	-0.03	45	0.825	Sw: 20.0; No: z-score;	1
V(h)		Pearson	4.2	3.17	-0.82	60	0.982	Sw: 10.0; Ro: True;	1
C		LCS	4.3	2.53	-0.33	45	0.978	Dm: chebyshev; No: procrustes; ε: 0.3; Pa: 10.0; δ: inf;	1
S		Correlation2D	3.1	2.96	-0.24	60	0.739	Sw: 20.0; No: z-score;	1
S(h)		LCS	3.1	2.87	-0.3	60	1	Dm: cityblock; No: z-score; Pa: 20.0; δ: inf;	1
H	4	Spearman	4.2	3.81	-1.08	90	0.975	Sw: 20.0; Ro: True;	1
H(h)		Spearman	4.2	3.88	-1.55	90	0.981	Sw: 20.0; Ro: True;	1
V		Minkowski	4.2	3.23	-0.36	30	0.477	Dm: chebyshev; No: z-score;	30
V(h)		Minkowski	2.1	2.94	-0.66	30	0.478	Dm: cityblock; No: z-score;	30

study. This robustness to accidental activations is demonstrated without using any delimiting gestures - all algorithms were running for the duration of the false positive dataset (with the exception of when there was no movement defined by a 1 px threshold).

We have demonstrated the importance of optimally aligning input and output signals both temporally and spatially. For temporal alignment, this is shown by the top-performing algorithms being either a lock-step algorithm featuring the sweep parameter, or an elastic measures which temporally warp the trajectories. The success of the lock-step measures with the sweep parameter suggests the ability of the elastic measures to temporally warp the signals is offset by the lock-step algorithm's abilities to align signals using the sweep parameter. This may infer that the leading/lagging element is more important to the matching process than the fluctuation in user-to-target distance over time.

From a detection perspective, two-dimensional target trajectories appear to be more robust to false positive activations compared with one-dimensional movements. With the latter there were fewer algorithms which could suppress false positives, and this could have contributed to the higher average activation times. There is greater variability in two-dimensional movements due to the use of two axis, and accidental motion is less likely to be spatiotemporaly matched in both axes simultaneously. However, in the previous section we observed user preference for the fast harmonic one-dimensional movements. Our results suggest that they may be more subject to false activations, and require longer for a successful detection (on average), than the two-dimensional movements. For applications in which accidental activations are more likely (e.g. public displays, busy environments), two-dimensional shapes could provide additional robustness.

Our results also highlight the trade-off between false positive rate and activation time. By sacrificing the false positive rate we can reduce activation time whilst maintaining 100% true positives, especially in the case of the one-dimensional movements. This can therefore be viewed as a design choice: does one prioritise robustness to false activations or activation time? This choice will depend on the context of the application, and it would be possible to use two sets of parameters - an initial robust parameter set to minimise false positives, and a second parameter set optimised for activation time once the user has initiated interaction with an application. One possibility is to design interfaces such that two-dimensional shapes are used for initial interaction with/activation of the system, and once interaction has begun the system could transition to the more favoured one-dimensional targets with more "relaxed" parameters

that optimise for activation time.

There are two types of false positive: an accidental movement selecting a target (studied in this section), and selecting the incorrect target when many targets are available for selection (studied in Chapter 3.3). Our analysis in the previous section showed us how users perform catch-up and slow-down movements, and therefore we must consider how the presented algorithms may cope with the presence of simultaneous targets. Take the sweep parameter for example. With a sweep size of 20 frames ($\approx$ 0.6s) we search for movement in a 1.2 s window around the target. In Chapter 3.3 we showed that users are capable of differentiating about four targets in one direction separated by 90°– which translates to temporal differences of 0.5 s for the faster movements and 1 s for the slower movements. Using the sweep parameter would therefore require disambiguation when multiple targets are selected.

One way of achieving this is to output the index of the sweep parameter which achieves the highest similarity score to find the target closest to the user movement. Likewise, with the elastic measures we can output the warping paths to extract which target the user is more closely following. In the case of the Procrustes algorithm, its additional computational complexity can be offset by outputting the rotation required for optimal alignment which also has the advantage that only one target need to be compared with. For example, consider a circular target, if we find that the optimum rotation is 180°, then we can infer the user is following the target which is 180°offset to the (only) target we compare against.

We also observe the importance of algorithm and parameter selection based on different target conditions, and thus the context of the application. Although our results demonstrate zero false positive rates across the board, there was not a single algorithm which achieved zero false positives across all shapes at either speed. We observed all of the algorithms achieving zero false positives for the two-dimensional shapes, however we also see a variety of parameter combinations featuring in the top-algorithms (five parameter combinations out of six). Ideally one approach would be suitable across all target configurations, reducing the burden of designers to have to optimise parameters of the system, and allowing for a plug-in-and-play style approach. The correlation 2D algorithm performed well across the desirable different types of motion, and is our recommended algorithm based on activation time. Table 5.8 details the parameters required for different contexts. We have provided a foundation to begin with by exploring existing trajectory matching techniques, and the insights we have gained from both the previous section and this can be used to inform future algorithm design.

All algorithms are either scale invariant or normalise the input data, and we have demonstrated the abilities of the algorithms to match across all of the types of input. The algorithms presented are scale invariant insofar as that they do not require input and output coordinate spaces to be in the same dimensions. We demonstrated how the algorithms can pick up relatively small movement compared to the camera with head movements, however for extremely small movements (e.g. micro-gestures, very large distance of user from sensing device) scale invariance is limited as the signal-to-noise ratio decreases. The insights we have gained from this algorithm evaluation are transferable to other body sensing devices and application contexts which may involve users at different distances to the camera. Other skeletal trackers (e.g. Kinect) and optical flow based approaches (e.g. TraceMatch) output similarly smooth trajectories to those outputted by the OpenPose skeletal tracker, and hence we would also expect thresholds and parameters to be somewhat transferable, or in the worst case provide an initial starting point.

Future work and Limitations

We focussed on the most common techniques used in the literature that do not require training, however there exists other techniques that we omit, e.g. Edit Distance on Real Sequence

Table 5.8: Summary table showing the recommended types of motion and parameters for the Correlation 2D algorithm

Context	Speed	Type of Motion	Buffer Size	Threshold	Sweep	Normalisation	Posthoc Window
Robust against accidental activation	Fast	Circle	15	0.764	0	Procrustes	30
(e.g. activation gesture)	Fast	Square (Harmonic)	30	0.705	20	Z-score	15
	Fast	Circle	30	0.885	20	Procustes	1
Fast activation times	Fast	Square (Harmonic)	30	0.705	20	Z-score	15
(e.g. selection)		Horizontal (Harmonic)	60	0.753	20	Z-score	1
	Fast	Vertical (Harmonic)	90	0.671	20	Z-score	15

(EDR) [44], Time Warped Edit Distance (TWED) [161], or statistical methods such as normalised cross-correlation (NCC) with Bayesian estimation [240, 157] or machine-learning based approaches [181, 195]. As suggested by Williamson and Murray-Smith in their pioneering "Pointing without a pointer" paper, there is also scope to develop much more sophisticated algorithms using methods from information and manual control theories which can incorporate a mathematical model of the human behaviour. These models could also incorporate contextual knowledge of the application domain for matching, e.g. tailoring to a specific type of input or assigning higher probabilities of a match occurring at specific times.

We have shown the robustness of motion correlation to accidental activation generated by the hands and head using semantic and spatial gestures. However, this is an indication of the robustness to common movements one would expect to be encountered, and deployments over prolonged periods may reveal more information on the robustness of the system in-situ. For a system such as TraceMatch, which senses movement from any source, there is an increased likelihood that a false activation is encountered. Another aspect we didn't consider is how robust different shapes are at being matched against each other. For example, if a circular target and a square target with the same phase are on an interface can they be differentiated? The ability to display more than one shape not only increases the number of available targets, but gives designers flexibility in how they can design interfaces, with different shaped widgets being used for different purposes. We have also only looked at movements in the image plane of the camera – we assume the users are facing the camera and not at an angle. Future work should look at the effect of different angles on the detection capabilities, or take this into account using depth sensors.

Insights gained from this work may not only transferred to other body sensing devices, but also to other modalities such as eye gaze. We would not expect these results to directly translate because other sensors may exhibit different characteristics. Drewes et al. found that introducing a fixed offset into the eye tracking signal increased correlation, however it is not clear where this was due to the sensor or the characteristics of smooth pursuit eye movement [63]. Eye trackers have much higher frequency noise due to the need to detect smaller movements, however insights such as the parameter sweep or Procrustes approach may be beneficial to aligning signals. This would require a similar analysis to be performed on eye movements.

Finally, although we have performed an extensive search of the parameter space, we have not performed an extensive search of the thresholds. The thresholds we have presented are based on this dataset and there is a danger of overfitting, as our parameters are derived from this specific dataset. It would be prudent to investigate how applicable they are to a wider range of users and in different application scenarios. Assuming an algorithm yields 100% true positive rate with 0% false positives, there exists a range of thresholds that could satisfy this requirement. In this work, we chose thresholds on the boundary of ensuring 100% true positive rate, and hence the thresholds are as robust to false positives as possible (whilst maintaining 100% true positive rate). Alternatively, we could have selected thresholds on the boundary of yielding 0% false positive rates. Once again the implications of this depend on whether the priority is suppressing

false activations or enabling true detections.

5.4 Conclusion

In this chapter we have conducted an in-depth study into how users can follow motions with different body parts, or whilst using an object, which has provided interesting insights for both interface and algorithm design. By utilising our knowledge of how users move, we have demonstrated how we can better design target movement to suit user's body movement by using simple harmonic motion. By displaying the target movement as simple harmonic motion, it is both easier to follow and more preferred by users. We found that users showed preference for one-dimensional target movement, however our analysis on algorithms suggests two-dimensional trajectories offers compelling properties including lower activation times and more robustness to false positives. We have demonstrated that motion correlation is robust against accidental activations caused by common semantic gestures, and provide optimal parameters for a range of algorithms for practitioners of motion correlation interfaces using body movements.

6

Discussion

In this chapter we revisit our proposition that motion correlation is the third input paradigm for touchless gestures and reflect on the research questions posed in Chapter 1. We discuss how semantic and spatial input can be used to support motion correlation (and vice versa), and how to better design motion on the interface. We then look at application areas that could benefit most from motion correlation, and distil interface design guidelines from across our studies. Finally, we discuss future work, including challenges and directions for incorporating user feedback into the proposed systems, and the opportunities that arise if we extend the TraceMatch/MatchPoint systems, before finishing on the limitations of our work.

6.1 Research Questions

6.1.1 Can movement be used as the primary sensing principle for interaction?

The concept of motion correlation has been used for different input modalities, from a mouse [73], to gaze [273], to hand movements [40]. In Chapter 3, we showed how movement can be used as the primary sensing principle, by a wide range of sources which can generate movement. By leveraging the underlying concept of motion correlation, we investigated:

- **RQ1.1:** *Is it feasible to harness movement as the primary sensing principle using motion correlation to allow users to provide input more flexibly?* By combining feature detection and optical flow techniques, we demonstrated how sensing of arbitrary motion can be used as input with the TraceMatch system. This approach avoids detecting and segmenting specific users, creating a posture-agnostic input method which allows users to interact flexibly with any body part or object.
- **RQ1.2:** *How effective are users at providing input using different body parts, or when holding objects, to generate the required motion?* Using an abstract selection task, we discovered users can provide input across a range of body parts, and even whilst using objects-in-hand. The speed of the target motion affected the task success rate for participants, especially when using the head for input. Participants were able to select a target from eight Orbits (four in both directions) with an average task success rate of 88% across all sizes, speeds, and types of movement during a controlled experiment, and with a task success rate of >97% in a naturalistic application context. However, our results could be

conflated by the use of the detection algorithm, and in Chapter 5 we further investigated how users provide input by abstracting from any particular detection algorithm.

- **RQ1.3:** *How do users provide input spontaneously when given the opportunity to interact with different body parts and objects?* Given that users can provide input by any means necessary, we studied how they spontaneously chose to interact with a range of applications based on real-world contexts. Participants expressed preference for different methods of input, including input whilst holding a mobile phone or a cup, however the dominant hand was the most preferred method of input. Despite this, it was not used exclusively and choice of input was context dependent (e.g. participants used different hands based on the position of the on-screen controls). Follow-up discussions with participants also revealed how the head provided relaxed input which could have more application in different contexts.

6.1.2 How can motion correlation be used to bootstrap spatial pointing?

Prior to this thesis, previous work focussed on motion correlation for discrete selection of objects, however many interactions in the real-world require more precise input control. In Chapter 3.5 we explored how more expressive input could be achieved with varying speeds, sizes, and colour of orbit, however motion correlation is unsuitable for continuous control. In Chapter 4 we explored how motion correlation can be extended to better support a wider range of input for interaction with touchless gestures, with a particular focus on spatial gestures. Subsequently, we investigated:

- **RQ2.1:** *Can the control-display gain for spatial interaction be derived from motion correlation?* We demonstrated how the movement in the motion correlation stage encodes important information, with which we can define a coordinate frame for spatial interaction. Beyond defining the coordinate space, the motion correlation phase empowers users to simultaneously select the function they wish to control, and the input to use (implicit in their action). This enables unique opportunities for interaction which can be tailored on a per-interaction basis.
- **RQ2.2:** *What unique capabilities arise from combining motion correlation with spatial coupling?* By combining motion correlation and spatial input, we can leverage both of their advantages for unique interaction techniques. We implemented MatchPoint, a complete implementation of spontaneous spatial coupling, which we used to explore the design space for interactions that leverage spontaneous coupling of multiple controls at a time, by one or multiple users, with different body parts, or with objects as tangible intermediaries.
- **RQ2.3:** *What are the usability implications of dynamically setting the control display gain?* We explored how the dynamic appropriation of "anything the user can move" as a pointing device affects the user's ability to consistently provide input accurately. Using the webcam-based MatchPoint, users were able to complete a multi-directional pointing task from a distance with approximately the same throughput as the Microsoft Kinect v1. However, with MatchPoint the CD gain was dynamically set each time the user acquired a pointer, and users were able to complete the task using their head, hand, and whilst using an object, whilst the system had no underlying knowledge of which input method was being used. The ability to accept any form of input is compelling as it enables users to choose a form of input that is convenient in a given context, and invites exploration of mappings that might not be general purpose but fitting for specific contexts.

6.1.3 How do users reproduce trajectories in the context of motion correlation?

Our ability to synchronise with external stimuli is dependent on a number of physiological subsystems working in coordination. We have provided an understanding about how users synchronise with target motion in the context of motion correlation by addressing the following sub-questions:

- **RQ3.1:** *How can we extract the ground truth of how well users follow a target, independent of any particular detection mechanism intended for interaction?* Extracting the ground truth of how well users follow moving targets is non-trivial due to the ephemeral nature of touchless gestures. In Chapter 5.2.1, we detail two approaches for extracting the ground truth of how users follow both circular motion, and line-based motion. For circular motions an ellipse fitting model can be used, with the start of the gesture extracted based on anomalous angular velocities at the start of the gesture. For line-based motions, we use a vertex and edge-based approach to identify salient parts of the motion.
- **RQ3.2:** *How much do users lead or lag a target when synchronising with different shapes, and across different input methods?* Using the ground truth extraction approaches we analysed how well users can synchronise with target motion, aggregated over all participants. This revealed how different target conditions affect the way in which users synchronise with different methods of input. Faster targets were preferred by users, however slower target motions results in closer synchrony. We also observed how there was little difference between dominant and non-dominant hand, confirming the findings from Chapter 3.3 that motion correlation does not rely on fine motor control.
- **RQ3.3:** *Can simple harmonic movement better help users synchronise with target movement?* We showed how presenting target movement as simple harmonic motion helped users to synchronise more quickly, more closely to the target signal, and with less oscillations than linear movement. In addition, participants expressed a preference for following simple harmonic motion compared with linear across all methods of input, describing it as "more natural" and "fluid".

6.1.4 How robustly can we detect users matching motions?

The success of an input technique can be decided on its ability to accurately detect user intentions, and reject accidental activations. Whereas the previous question addressed how well users actually synchronise with targets, in Chapter 5.3, we demonstrate a 100% success rate of detecting users follow target motion, with 0% false activations against common semantic and spatial gestures. Two sub-questions shed light on this:

- **RQ4.1:** *Which algorithms are appropriate for detecting motion correlation?* We identified eight algorithms from the motion correlation and wider trajectory matching literature suitable for matching user motion to target motion. We focussed on those approaches which view the problem as matching two trajectories, and where a match is determined by extracting a similarity measure. We also identify several other approaches, including probabilistic and machine learning, which we did not investigate but have potential for future research and application.
- **RQ4.2:** *What are the best parameterisations for achieving optimal performance across different types of input?* We focussed on two specific contexts for extracting parameterisations for use with motion correlation: robustness against false activations (0% false positive rate), and a more lenient context for interaction which minimises activation time whilst ensuring a 5% false positive rate. Following an extensive search, we detail numerous parameterisations across different algorithms that satisfy these criteria.

6.2 Motion Correlation as the Third Input Paradigm

Our goal at the outset of this thesis was to position motion correlation as the third fundamental interaction paradigm for touchless gestures. The notion of motion correlation as the third input principle for touchless gestures has wider implications that extend beyond the research laboratory. Unlike the mouse and keyboard or touch-based input, there is no agreed upon means of interacting with devices remotely. In a world where computing is becoming more ubiquitous with internet-enabled devices, users are often faced with the issue of multi-device interaction. Our ability to follow motion is universal, and transcends languages and cultures, and motion correlation has the attributes for being a fundamental principle for interaction across devices. However, interaction across multiple devices may require inter-device communication to ensure uniqueness of trajectories is maintained to ensure there is no ambiguity over which device the user is addressing. We primarily focussed on simple selection tasks, intended for lazy input where the number of options available are limited (< 10), which could equally be used as activation gestures to first address a device prior to interaction, similar to the suggestion of Freeman and colleagues [81]. However, due to the time required for selection, motion correlation is not suitable for tasks requiring multiple sequential selections from a large vocabulary of input options, such as typing.

Throughout this thesis, we have demonstrated that motion correlation is a powerful and versatile input paradigm that abstracts from the need to provide input with specific body parts, objects, or devices. In Chapter 3 we introduced TraceMatch, a computer vision technique which uses generic motion as input without requiring user segmentation. Our subsequent studies highlighted the versatility of motion correlation, showcasing how users are adept at providing input with a variety of different types of input. Whereas we have focussed on sensing motion using computer vision techniques, other sensing modalities can be used to capture our fundamental ability to synchronise with motion on the interface.

Not only is motion correlation a powerful interaction paradigm in its own right, it lends itself to integration with other principles. In Chapter 4, we introduced spontaneous spatial coupling and showed how motion correlation can be used to support interaction beyond discrete selection. By combining it with conventional spatial input we increase the available design space for more expressive input, whilst maintaining the versatility to provide input by any means. In this case, there is a symbiosis between spatial input and motion correlation in which both techniques benefit from the other. Motion correlation has compelling properties for simple selection tasks which can reduce the gorilla arm effect as cursor-based selection requires the arm to be held aloft for longer periods [101]. Spatial input also benefits from a more seamless approach to defining the spatial mapping, which can now be defined on a per-interaction basis and can help to avoid ill-defined mappings. Similarly, motion correlation benefits from the ability of spatial input to provide continuous control in a precise manner. We can also look towards how semantic input can support motion correlation and vice versa.

Semantic gestures are powerful because they draw upon our natural communication skills and can be performed in an eyes-free manner. However, as discussed in Chapter 2, there are discoverability and memorability issues with ill-defined gesture mappings – motion correlation can bridge this gap. In Chapter 3.4 we showed how motion correlation could be applicable to a number of application scenarios featuring non-trivial mappings (e.g. selecting “multi-view” in the Formula One application). In future, touchless gesture sets could consist of both semantic and motion correlation-based gestures. Those gestures which are obvious to users can be provided semantically, with motion correlation used to “fill in the blanks” for those gestures whose mappings are non-trivial.

In contrast to semantic gestures, which are often tied to a particular body part, the ability to provide input using any body part or object provides opportunities to provide input seamlessly

whilst performing other tasks. Iconic gestures (e.g. the tracing of a spatial trajectory) also exhibit similar abstract properties, but without applying the temporal constraint of motion correlation gesture sets are limited in the one-to-one mapping of shape-to-input command. In contrast, motion correlation allows the reuse of a single shape multiple times by varying the phase or speed of the targets. However, movement on the interface required to display targets could be visually distracting for some applications. A small subset of semantic gestures (e.g. one for each body part) could provide an eyes-free activation gesture to trigger the motion used for selection.

Our analysis in Chapter 5 suggests the spatio-temporal element of motion correlation is crucial in being able to reject accidental activations from common semantic gestures which we use in everyday life, and which are more susceptible to being inadvertently detected. Likewise, iconic gestures with no temporal constraint (i.e. tracing a circle) will suffer from accidental activation from other movement in the scene as demonstrated in Chapter 5, and obscure activation gestures (e.g. a teapot [279]) may cause user embarrassment or be hard to remember. Thus, in addition to its discoverability, motion correlation can support semantic gesture systems by providing a more robust activation gesture which can be used to address multiple devices, such as demonstrated in Freeman et al.'s Do-That-There [81].

6.3 Motion on the Interface

Inherent to motion correlation, as discussed in this thesis, is motion on the interface. In Chapter 5 we discussed the kinematic aspects of motion on the interface, however the aesthetic design of the motion is also a vital component. In addition to providing discoverability and feed-forward capabilities for interaction, motion-based design in interfaces provide a number of advantages. It can be used to drive user attention, making it explicitly clear to the user what interactions are available, and be utilised by designers to indicate the flow of an application. We have shown that the properties of the motion shown can be used to convey underlying functionality of the control, for example by using colour or the speed, and provide intuitive controls. As opposed to traditional interface layouts (e.g. grid layouts), motion correlation allows for unique layouts and overlapping targets which can be used to indicate spatial and temporal hierarchical relationships between objects to support usability. This could be in the form of grouping similar functionalities in a design, or by using motion to highlight differences (e.g. clockwise for positive, anticlockwise for negative).

Design of the motion is also important for when users are not interacting with the system. Our peripheral vision is attuned to detecting motion, and our peripheral temporal sensitivity is almost equal to that of our foveal temporal sensitivity [169]. This has both positive and negative implications. Users could synchronise with simple one-dimensional movement patterns without having to look directly at the target. However, an important consideration for some applications is how to reduce motion when not required to avoid unnecessary distraction. As mentioned in the previous section, one approach could be to use a semantic gesture to activate a motion correlation interface. Alternatively, one could limit input to active regions of the sensor and only display motion on the interface when movement in the active region is detected. However, this removes the posture-agnostic element of a system like TraceMatch and constricts the way in which users can interact with the system. Another approach could be to limit distraction by design of the motion itself. We have only considered a simple design of target motion in this thesis (i.e. a moving dot), but the design of the target could be much more subtle, and even blend into the background based on the application scenario. It is important to note that motion correlation is feedback bound, and that a user needs to attend to the target visually to synchronise.

Another design consideration is the type of motion displayed. In this thesis we have demonstrated how simple harmonic movement, inspired by the way in which we move our body, can

be synchronised with more easily and is preferred by users. The use of naïve physics can be used in interfaces to add the illusion of mass, inertia, and/or springiness to digital widgets [115]. Using physics-based motion has been used in previous motion correlation interfaces with fish [273], flying insects [273], and shooting stars [199], where the movement of the targets mimicked their real-life counterparts. Aside from the harmonic motion introduced here, little is known about how we can utilise (simulated) physical properties of targets as an abstract concept in itself to help in perception and matching by making motions more predictable and aesthetically pleasing.

There exists other avenues in which we can show motion to users, in this work we have considered target trajectories in 2D space on a display. Prior work has also shown mechanical systems can be used to generate the motion required for users to synchronise with [271]. This may offer unique and novels methods of interaction, where the properties of the mechanical system could be naturally compatible with human motor system/perceptual/prediction capabilities, e.g. a swinging pendulum. Previous work in the psychomotor literature also illustrates how we are adept at synchronising with non-spatial cues, such as a flashing light [215], and work in the motion correlation literature has shown how users can synchronise with a metronome using micro-gestures [82].

6.4 Application Areas

In Chapter 2 we discussed the application areas most suited to touchless gestures. Some of those may not be suitable for motion correlation because of the visual attention required (e.g. automotive), however we believe the techniques presented here have applications in a number of different areas. We have primarily explored motion correlation in the context of a living room/television application space, but the concept of motion correlation can expand into other areas of *smart home* living, such as when cooking in the kitchen with messy hands or whilst using cooking utensils. In this case, display of the motion need not only occur on a display such as a TV, but could instead occur on equipment itself such as the hob or microwave. The surgical domain is another application which involves asepsis and could benefit from motion correlation techniques. Surgical instruments need not be put down to interact with the system, which could provide both simple selections (e.g. selecting a medical image), and spatial interaction (e.g. manipulating said medical image).

Motion correlation does not require fine-grained motor control, as shown by similar abilities across dominant and non-dominant hands, which presents opportunities to explore how it can be used as a means of providing input for those who are not capable of fine-grained motor control. We have alluded to the fact a system that detects generic motion could have implications for users who can not use traditional methods of touchless input. Both the TraceMatch and MatchPoint systems do not perform user detection or segmentation, and hence have no built-in knowledge of the human body. Thus, users with physical disabilities, such as amputees, could use the system unlike those which rely upon detection the hands. Likewise, the concept of motion correlation is suitable for sensing devices not suited for pointing, but equally it is suitable for users who are inaccurate or incapable of pointing precisely, and for those who do not have the dexterity or hands to perform complex semantic gestures. This makes motion correlation ideally suited to be investigated as an accessible input technique, but more work is needed to investigate how it can best support those who can not use traditional input touchless input methods.

6.5 Design Guidelines

The set of studies presented in Chapter 3, and the results from Chapter 5 give insight into design choices, from which we distil guidelines for the design of interfaces based on motion correlation

using body movements. We reflect insights on the following properties:

- *Shape* – What is the best shape to use?
- *Speed* – What is the best speed to use?
- *Number of Targets* – How many targets should be displayed simultaneously?
- *Size and Position* – Does the size and position of the targets make a difference?
- *Multi-Level Input* – How should one convey additional information using the Orbits for multi-level input?

6.5.1 Shape

Overall, this thesis has shown many shapes are suitable as input for motion correlation. Our analysis in Chapter 5 studied both one- and two-dimensional shapes and showed each has benefits and drawbacks. One-dimensional shapes are easy to perform across different types of input and users preferred them for their simplicity. However, from an algorithm perspective, designers have to be much more careful about parameter optimisation to avoid false positives. Two-dimensional shapes also have compelling properties. It is easier to reject false positives across all algorithms, and in Chapters 3 and 4 we showed how their button-like appearance can be used for inspiring designs. Uniform circular movement, the projection of which undergoes simple harmonic oscillation, is easy to perform across different types of input and economical in terms of effort [94], and was a popular choice for participants in Chapter 5. We also saw how square-based movement was more closely synchronized with than one-dimensional movement, and provides clear perceptible events with which the user can synchronise [254]. However, we also observed some users struggling with the fast head movements for two-dimensional shapes, and any application contexts where users are more likely to use their head for interaction could benefit from slow or one-dimensional movements. This highlights the importance of the speed of moving targets presented to users. Also, different body parts have preferential directions and axes of movement, for example the head has limited translational movement with more movement in the horizontal axis than the vertical. This should be taken into account when designing applications, for example when hands-free input is desirable. We have studied a small repertoire of basic movements which can be performed across body parts. There is a larger space of shapes for other research to explore.

6.5.2 Speed and Type of Movement

In this thesis, we have seen how both the magnitude and type of movement affects user preference and their ability to accurately synchronise with moving targets. In Chapter 5 we showed how presenting targets as simple harmonic motion was both preferred, and enabled better synchronisation, than targets presented with constant velocity. For body movement-based motion correlation interfaces there is little reason to pick constant velocity over simple harmonic motion other than the quicker average activation times witnessed in the algorithm section of Chapter 5 – most likely due to simple harmonic motion being more likely to be susceptible to accidental activation. Our studies have also shown that there are implications on the types of input that can be used based upon the speed of the targets. In Chapter 3.3, we observed participants did not perform well with fast head circular head movements, and in Chapter 5 we saw much higher variability for the linear fast square-shaped movement with the head. Designers must be aware of limitations with the use of fast targets which could inadvertently limit, or at least hinder, the ways in which a user can provide input.

The application context is also important when deciding the speeds of the moving target. In our studies, we purposefully filled the cup with (cold) water to simulate a drink. The ability

to provide input with any object presents potential dangers depending on the object used, for example gesturing with a hot cup of tea would be more dangerous at fast speeds. In Chapters 3.4 and 3.5 we observed participants using different types of movement to interact with the system in a context where they were not performing other tasks other than interacting with the applications. The varied use of different type of input can, in part, be explained by possible order and novelty effects of having completed the first study and using the system for the first time. However, in real-world deployments users may be performing other tasks with their hands when interacting with the system, such as eating, drinking, cooking, or in a public display context the user may have bags of shopping or be carrying or attending to a young infant. In this context, slower targets would allow users to fully utilise the principle of providing input by whatever means necessary.

In Chapter 3.3 the slower Orbits achieved a significantly higher task success rate across all movement types, and our analysis in Chapter 5 showed users could synchronise more closely with slower targets than their faster counterparts. However, slower movements showed more variability and we observed different preferences with regards to participant favoured speed, suggesting one speed does not suit all. This presents a conundrum to designers – what if the user performs better with one speed, but prefers the other? Here we invoke the popular saying – *"the customer is always right"*. If a user prefers faster targets then they should be able to configure the system in a personalised manner. This can be achieved using a "settings" option to allow the user to configure the speed on-the-fly to a configuration which they feel most comfortable using. Alternatively, the system could adapt in real-time to the user by analysing how well a user synchronises with target motions with different speeds, however this would not take into account their subjective preferences. For public displays users should not be expected to have to configure the interface before commencing interaction. An alternative approach would be to use two sets of targets with both slow and fast moving targets, however this could make the interface confusing. We therefore advise that the default speed of the targets are set to slow in this context, to maximise the way in which users can accurately provide input.

6.5.3 Number of Targets

Based on our findings in the first study, we would recommend a default maximum limit of eight simultaneous circular targets (four clockwise and four anti-clockwise) presented on an interface when using only one speed. Reflecting on our findings in Chapter 5, we observe that following circular targets results in higher variability compared with line-based movements, and therefore we can infer that the number of simultaneous line-based targets would be at least the same as circular targets. The speed of the targets is also a factor when considering how many should be displayed simultaneously. In Chapter 3.3, six orbits (allowing for 12 simultaneously) achieved a task success rate of 80% across all movement types for slow speeds, with seven participants (35%) achieving an average task success rate of at least 90% with six orbits across all movement types, and only three (15%) participants achieving a task success rate of less than 75%. In contrast, only two participants achieved a high task success rate ($> 90\%$) when using fast movement types. Analysis of how users followed motion in Chapter 5 suggests that users are able to follow slower moving targets more closely than faster moving targets, and due to the fact the time for one cycle of the gesture is longer it provides greater room for error when selecting amongst many simultaneous targets. Therefore we would only recommend increasing the capacity to twelve targets (six clockwise, six anticlockwise) when using slow moving targets and if the application necessitated the increased capacity.

The shape of the target is important when considering the number of simultaneous targets. Periodic one-dimensional motions have no notion of direction, thus limiting the number of targets given a specific phase offset compared with two-dimensional shapes. Therefore, one would

have to use (for example) both horizontal and vertical moving targets to achieve the same number of simultaneous targets as a circle/square with targets in both clockwise and anti-clockwise directions. The use of different speeds for the targets, such as shown in Chapter 3.5, also has the potential to increase the capacity of the number of simultaneously displayed Orbits, however one must take care in ensure the difference in speeds is sufficient enough to avoid false activations occurring due to the overlapping trajectories. We used a time difference of 1 second between the speeds, however a larger offset may result in a much lower false activation rate. We therefore recommend a minimum difference of at least 2 seconds between the speeds of the targets. If using this approach, a practical capacity of sixteen simultaneous targets could be displayed using two sets of targets with different speeds. Although we have only studied simultaneous circular targets in this work, there is also the possibility of multiple shapes being used to extend the maximum capacity, as long as the system can accurately differentiate the different trajectories.

6.5.4 Size and Position

In Chapter 3.3, we observed that the size of the targets did not affect task success rate. Smaller trajectories may take up less screen space, however a user's ability to accurately identify the trajectory should be taken into account for extremely small target trajectories. We predict there will exist a lower bound on the size in which the user can accurately distinguish the movement, although we did not see one in our study suggesting very small target trajectories are possible. This may vary depending on the shape, for example a square may have a smaller lower bound than a circle because of the corners acting as perceptible events.

With regards to the positioning of the targets, we observed interesting user behaviour in Chapters 3.4 and 3.5, with some users changing the way in which they provided input depending on the position of the target on the screen. Similar behaviour was noticed in PathSync, where they found a positive correlation between where users gestured and where the target appeared on the screen [40]. Based on this, one might consider placing the targets centrally where possible so that a user's input is not influenced by their position on the screen.

6.5.5 Multi-Level Input

In Chapter 3.5 we showed how colour, speed and size can convey additional information to the user for multi-level input. When using different speeds to convey additional information it is important to consider the difference in speed of the targets to avoid false activations that occur as a result of the trajectories overlapping. In contrast, size has been shown to have no significant effect on task success rate, therefore this is a "safer" way in which additional information can be conveyed. The downside to this is that not all users understood the implicit information conveyed through the size or speed alone, therefore it is important to utilise both properties. Some participants understood the concept of using colours to convey information, however this is limited and could pose an issue for users who are colour blind. Instead we recommend using icons for the targets of the Orbits to convey information more explicitly.

6.6 Future Work

In this section, we discuss future work and directions that arise from the insights and findings of our research.

6.6.1 Feedback and Reactivity

Providing feedback about an interaction is one of the basic heuristics of interaction design [186], and is important in gestural interfaces to let users know their actions are being sensed [19]. For motion correlation interfaces, feedback on the progress of a match could be provided on the target itself, on the path preview, or on an object of interest (e.g. if the path surrounds an object). Providing feedback for systems like TraceMatch, where any motion in a scene could be used as input, involves additional complexity. There may be cases where a motion in the scene starts mimicking the movement of one of the targets, not enough to cause a detection but enough to trigger feedback to be given. This could present a confusing situation for users because it is not obvious whether it is their movement, or movement of another object, that is causing the feedback. This could also increase the issue of the targets being visibly distracting, and hence subtle forms of feedback may be more suitable to avoid the user's attention being drawn to targets when accidental motion is causing feedback.

In this thesis, the moving targets we have studied have moved independently of user input. Feedback could also be provided to users by introducing a "reactive" element, in which targets change their motion according to user input. This could provide a number of advantages and opens up some interesting research questions. First let us consider the process of disambiguating multiple targets when many are simultaneously presented. As demonstrated in Chapter 3, the most robust way of differentiating targets is by using the phase. If the system detects a user is following a target, it could maximise the phase difference between its nearest neighbours to better help disambiguate which target the user is following. In this scenario we look at changing the motion of the targets around the target we believe the user is following, but we could also consider what would happen if we changed the target the user is following itself. This could be useful for further suppressing false activations, because the user must react to the system changing the motion, and acts as a method of visual feedback to the user of the system's detection of their movement.

6.6.2 Input Segmentation

In this thesis we have highlighted the benefits of abstracting from body part segmentation for input. We have shown how different types of input exhibit distinct differences in their ability to match moving targets, provide spatial input, and in user preference. In MatchPoint, we demonstrated how the properties of the matched motion can be used to inform control-display gain, thus accounting for the use of different inputs and range of movements. However, we do not perform user segmentation or object identification to identify what triggered the match. Adding an element of body part segmentation when a match is detected could increase the capabilities of these systems. Interfaces could be tailored to the type of input used to make the selection, for example if the head is detected a much simpler spatial interaction could be presented due to the higher difficulty in pointing.

In Chapter 4 we showed how the unique affordances of physical objects enable the spontaneous creation of tangible user interfaces. A system which detects what input was used could dynamically switch between spontaneous interaction of the body, to prolonged interaction with objects such as those shown in MatchPoint. The specific properties of an object (e.g. shape, weight) could also be inferred, and interaction in the spatial stage could utilise specific affordances (e.g. nudging, rolling, tilting) of the object, in addition to, or instead of, translational movement. The use of body part segmentation could also be used to provide input-dependent selections, either as a means to increase the number of available targets for selection or for entertainment purposes, such as a game for children in which they must use different objects depending on the context.

6.6.3 Multi-modal Interaction

There are exciting opportunities to combine body-based motion correlation with other modalities. As discussed in Chapter 2, gestures are commonly used in a multi-modal fashion to support speech. *Natural language interfaces* that can understand and interpret human speech have gained huge popularity in recent years due to advances in natural language processing (NLP) and cloud services, however it is difficult to refer to spatial locations with speech [49, 194]. Voice could be used to support motion correlation and spontaneous spatial coupling as a confirmation/cancellation mechanism, as opposed to the dwell we used in Chapter 4. In the case of motion correlation, the mimicking of motion could be used to disambiguate which target the users wants to select, with voice acting as the confirmation and/or cancellation. This provides additional robustness against accidental activations because users must both mimic motion and provide verbal confirmation. It also reduces/removes the need to establish a threshold for the matching process because we need only identify which target is closer to the user's movement.

This also prompts questions about how body movement-based motion correlation could be combined with other modalities, such as gaze. Although gaze has been used for selection with motion correlation, our eyes must follow the target to initiate sensorimotor synchronisation. Sensing both modalities could provide interesting opportunities – two modalities provides sensing robustness against false positives (are both the hands and eyes following?), and only following with the eyes could act as a different mode (e.g. as a pre-selection to display more information) with confirmation occurring only when both hand and eyes follow the motion.

6.7 Limitations

We have focussed on the use of general purpose cameras as a sensing device that have deep market penetration. However, a camera can only capture a scene in two-dimensions and requires line-of-sight of the users. Our studies have involved users performing gestures square on to the camera itself, in the optical axis of the camera. In real-world deployments it would be advantageous for the systems to accept input irrespective of the user's orientation to the camera. This could be achieved using depth sensors, however they are much less ubiquitous than their 2D counterparts. Vision-based systems also have potential privacy concerns that must be taken into account for in-the-wild deployments. However, it should be noted that by using generic motion as input, and not performing user detection or recognition, the systems we have developed are only subject to bad actors in terms of intercepting visual data.

In this thesis we have studied motion correlation and the systems we have developed in the context of providing input in a living room scenario. As discussed, motion correlation is compelling for interaction in public spaces due to its discoverability, self-revealing nature, and ability for simple gesture sets. Similar to the issue of revealing a gesture set to users, is the issue of revealing *how* a user should provide input to a system. In PathSync, the button-like appearance of the moving targets led some participants to think they were buttons, as touch-based input is ubiquitous in modern life [40]. To fully leverage the capability of systems like TraceMatch/MatchPoint, which allow input through motion, users must be aware that they can provide input by any means.

The studies in this thesis have been conducted under lab conditions. In-the-wild studies may reveal interesting insights into how users interact with the system over prolonged periods. For the systems we have developed, we have not fully explored how scalable they are in terms of the amount of motion they can process. In our studies, motion was limited to a sub-frame of what was captured by the webcam and to a single participant. In-the-wild deployments may feature multiple people, some intending to interact and others not. We would therefore expect more motion to be present in the scene, increasing the computational cost as more feature points require

processing and also increasing the risk of false positives. In these contexts more sophisticated approaches to processing motion, such as limiting the density of features detected or background subtraction masks, may be required to minimise computational cost and ensure the most salient motion of users is captured.

7

Conclusion

As computing becomes more embedded in our everyday lives we look towards more seamless interactive experiences. At the outset of this thesis we posited that motion correlation is a fundamental interaction principle which presents new opportunities for touchless gestures. In Chapter 2, we showed how motion correlation is a generic principle that can be applied across different types of input without a tight coupling between input and output spaces. The motions presented are self-revealing and highly discoverable, guiding the user through the interaction and reducing the need to design and remember complex gesture sets. We have leveraged and built upon these properties in this thesis.

TraceMatch showed how users can provide input flexibly in a posture-agnostic manner by using movement as the sensing principle for input (RQ1). Spontaneous spatial coupling demonstrated how motion correlation can bootstrap spatial input, enabling seamless and ad-hoc appropriation of pointers (RQ2). Our in-depth analysis has provided grounding into how adept users are at synchronising with moving targets (RQ3), and an extensive search of the parameter space across algorithms from both the motion correlation and time series comparison literature highlights the robustness of motion correlation to accidental activation, and provides optimal parameters and thresholds for future practitioners (RQ4). The design guidelines we distilled in Chapter 6 provide a firm foundation for the creation of motion interfaces. In conclusion, we have shown how motion correlation is a versatile and expressive technique with unique properties for touchless gestures, and how it can be integrated with, and enhance, existing techniques.

Our insights have also sought to guide further directions in which research into motion correlation can be expanded, both in general and with touchless gestures. We demonstrated how the kinematics of motion on the interface can be better designed to support users with touchless gestures by leveraging our innate ability to perform simple harmonic motion. We have scratched the surface on designing target motions based on how we know people move, or similarly based on what their limitations are. In the future, incorporating model-based approaches on how we move, or what our limitations are at run-time, provide interesting research opportunities. Equally important is the consideration of how best to design the visual aesthetics of motion on the interface, which can lead to minimally distracting interfaces that offer unique layouts compared to the traditional grid-layout of current interfaces. Feedback of the proposed systems is non-trivial, yet reactive elements of motion correlation interfaces, where the interface responds to the user's input, could further enhance the interactive experience.

The systems we have developed could provide unique opportunities in a variety of application spaces. Of particular interest are those areas in which interaction may be limited due to the

use of objects that are required for primary task completion, and where a flexible input method which does not rely on recognition of user or object is desirable. There are a variety of examples of this, such as cooking in the kitchen, performing surgery, using tools to follow a DIY video, where motion correlation has the potential to provide seamless input. Of particular interest is the accessibility space, where touchless input is currently lacking for some users. This would require a user-centred design approach to ensure that motion correlation, along with both semantic and spatial input, are utilised fully for the benefit of users. As computer vision and machine learning techniques continually evolve, so do the opportunities that arise from using motion as input. Systems like TraceMatch and MatchPoint could leverage advances in these fields for higher accuracy and precision, and to incorporate "intelligent" aspects whilst maintaining the flexibility and posture-agnostic sensing abilities, further adding to the systems' touchless capabilities.

Appendices

A

Algorithm Results

This appendix reports the results of the top-performing parameter combination of all algorithms in Chapter 5.

Table A.1: Top performing algorithms for the Procrustes algorithm that achieve a 100% true positive rate, whilst minimising both false positives and average activation time for the circle (C), square (S), horizontal (H), vertical (V), and their harmonic (h) variants.

Shape	Speed	Algorithm	TP Rate	FP Rate	Activation time (s)	Buffer size	Threshold	Parameters	Posthoc
C		Procrustes	100.0	0.0	2.44	15	0.051		30
S		Procrustes	100.0	0.0	3.09	60	0.152	Sw: 10.0;	15
S(h)		Procrustes	100.0	0.0	2.75	45	0.091	Sw: 20.0;	15
H	2	Procrustes	100.0	0.0	3.89	60	0.073	Sw: 20.0;	30
H(h)		Procrustes	100.0	0.0	4.24	90	0.049	Sw: 20.0;	1
V		Procrustes	100.0	0.0	4.03	90	0.107	Sw: 20.0;	1
V(h)		Procrustes	100.0	0.0	4.07	60	0.052	Sw: 20.0;	30
C		Procrustes	100.0	0.0	3.12	60	0.035	Sw: 20.0;	1
S		Procrustes	100.0	0.0	3.73	90	0.095	Sw: 20.0;	1
S(h)		Procrustes	100.0	0.0	3.19	60	0.034	Sw: 10.0;	1
H	4	Procrustes	100.0	0.0	4.89	120	0.066	Sw: 20.0;	1
H(h)		Procrustes	100.0	14.6	5.30	120	0.068	Sw: 20.0;	15
V		Procrustes	100.0	12.5	4.61	90	0.052	Sw: 20.0;	15
V(h)		Procrustes	100.0	20.8	3.57	45	0.114		30

Table A.2: Top performing algorithms for the Pearson algorithm that achieve a 100% true positive rate, whilst minimising both false positives and average activation time for the circle (C), square (S), horizontal (H), vertical (V), and their harmonic (h) variants.

Shape	Speed	Algorithm	TP Rate	FP Rate	Activation time (s)	Buffer size	Threshold	Parameters	Posthoc
C		Pearson	100.0	0.0	2.65	45	0.986	Ro: False; Sw: 20.0;	1
S		Pearson	100.0	0.0	3.15	45	0.913	Ro: True; Sw: 20.0;	30
S(h)		Pearson	100.0	0.0	2.48	30	0.954	Ro: True; Sw: 20.0;	15
H	2	Pearson	100.0	0.0	3.10	60	0.970	Ro: True; Sw: 10.0;	1
H(h)		Pearson	100.0	0.0	3.26	60	0.985	Ro: True; Sw: 10.0;	1
V		Pearson	100.0	4.2	4.47	120	0.796	Ro: True; Sw: 10.0;	1
V(h)		Pearson	100.0	0.0	4.10	60	0.968	Ro: True; Sw: 20.0;	30
C		Pearson	100.0	0.0	2.91	45	0.973	Ro: True; Sw: 20.0;	15
S		Pearson	100.0	0.0	3.75	60	0.924	Ro: True; Sw: 10.0;	30
S(h)		Pearson	100.0	0.0	3.23	60	0.978	Ro: True; Sw: 10.0;	1
H	4	Pearson	100.0	0.0	4.90	120	0.961	Ro: True; Sw: 20.0;	1
H(h)		Pearson	100.0	1.0	4.97	120	0.966	Ro: True; Sw: 20.0;	1
V		Pearson	100.0	0.0	3.59	60	0.967	Ro: True; Sw: 10.0;	15
V(h)		Pearson	100.0	0.0	3.60	45	0.969	Ro: True; Sw: 10.0;	30

Table A.3: Top performing algorithms for the Spearman algorithm that achieve a 100% true positive rate, whilst minimising both false positives and average activation time for the circle (C), square (S), horizontal (H), vertical (V), and their harmonic (h) variants.

Shape	Speed	Algorithm	TP Rate	FP Rate	Activation time (s)	Buffer size	Threshold	Parameters	Posthoc
C		Spearman	100.0	0.0	2.74	45	0.946	Ro: True; Sw: 20;	15
S		Spearman	100.0	0.0	3.51	60	0.921	Ro: True; Sw: 10;	15
S(h)		Spearman	100.0	0.0	2.96	60	0.958	Ro: True; Sw: 20;	1
H	2	Spearman	100.0	0.0	3.10	60	0.978	Ro: True; Sw: 10;	1
H(h)		Spearman	100.0	0.0	3.23	45	0.982	Ro: True; Sw: 20;	15
V		Spearman	100.0	0.0	5.64	120	0.884	Ro: True; Sw: 20;	30
V(h)		Spearman	100.0	0.0	3.99	90	0.946	Ro: True; Sw: 20;	1
C		Spearman	100.0	0.0	3.86	60	0.935	Ro: False; Sw: 20;	30
S		Spearman	100.0	0.0	4.23	90	0.926	Ro: True; Sw: 10;	15
S(h)		Spearman	100.0	0.0	3.50	60	0.949	Ro: True; Sw: 10;	15
H	4	Spearman	100.0	3.1	4.71	120	0.952	Ro: True; Sw: 10;	1
H(h)		Spearman	100.0	0.0	5.43	120	0.973	Ro: True; Sw: 20;	15
V		Spearman	100.0	1.0	4.04	90	0.977	Ro: True; Sw: 10;	1
V(h)		Spearman	100.0	2.1	3.48	45	0.872	Ro: True; Sw: 0;	30

Table A.4: Top performing algorithms for the 2D Correlation algorithm that achieve a 100% true positive rate, whilst minimising both false positives and average activation time for the circle (C), square (S), horizontal (H), vertical (V), and their harmonic (h) variants.

Shape	Speed	Algorithm	TP Rate	FP Rate	Activation time (s)	Buffer size	Threshold	Parameters	Posthoc
C		Correlation2D	100.0	0.0	2.33	15	0.755	Sw: 20.0; No: z-score;	30
S		Correlation2D	100.0	0.0	2.62	30	0.661	Sw: 20.0; No: procrustes;	30
S(h)		Correlation2D	100.0	0.0	2.27	30	0.705	Sw: 20.0; No: z-score;	15
H	2	Correlation2D	100.0	0.0	3.07	60	0.771	Sw: 20.0; No: z-score;	1
H(h)		Correlation2D	100.0	0.0	4.01	90	0.729	Sw: 10.0; No: z-score;	1
V		Correlation2D	100.0	0.0	4.05	60	0.713	Sw: 20.0; No: z-score;	30
V(h)		Correlation2D	100.0	2.1	4.29	90	0.671	Sw: 10.0; No: z-score;	15
C		Correlation2D	100.0	0.0	3.05	30	0.721	Sw: 10.0; No: z-score;	30
S		Correlation2D	100.0	0.0	3.86	60	0.461	Sw: 20.0; No: minmax;	30
S(h)		Correlation2D	100.0	0.0	3.45	45	0.772	Sw: 20.0; No: procrustes;	30
H	4	Correlation2D	100.0	2.1	4.82	120	0.755	Sw: 20.0; No: procrustes;	1
H(h)		Correlation2D	100.0	5.2	3.83	45	0.616	No: z-score;	30
V		Correlation2D	100.0	0.0	3.72	30	0.674	Sw: 10.0; No: z-score;	30
V(h)		Correlation2D	100.0	1.0	3.06	30	0.603	No: z-score;	30

Table A.5: Top performing algorithms for the Minkowski algorithm that achieve a 100% true positive rate, whilst minimising both false positives and average activation time for the circle (C), square (S), horizontal (H), vertical (V), and their harmonic (h) variants.

Shape	Speed	Algorithm	TP Rate	FP Rate	Activation time (s)	Buffer size	Threshold	Parameters	Posthoc
C		Minkowski	100.0	0.0	2.44	30	0.217	Dm: euclidean; Sw: 20.0; No: z-score; Pp: mean;	15
S		Minkowski	100.0	0.0	2.86	30	0.248	Dm: chebyshev; Sw: 10.0; No: z-score; Pp: mean;	15
S(h)		Minkowski	100.0	0.0	3.06	45	0.476	Dm: euclidean; Sw: 20.0; No: z-score; Pp: mean;	30
H	2	Minkowski	100.0	0.0	3.03	45	0.187	Dm: chebyshev; Sw: 20.0; No: z-score; Pp: mean;	15
H(h)		Minkowski	100.0	0.0	3.55	60	0.159	Dm: euclidean; Sw: 20.0; No: minmax; Pp: mean;	30
V		Minkowski	100.0	0.0	2.84	45	0.199	Dm: cityblock; Sw: 20.0; No: z-score; Pp: mean;	1
V(h)		Minkowski	100.0	6.2	3.67	60	0.356	Dm: cityblock; Sw: 10.0; No: z-score; Pp: mean;	30
C		Minkowski	100.0	0.0	3.20	45	0.313	Dm: chebyshev; Sw: 10.0; No: z-score; Pp: mean;	30
S		Minkowski	100.0	0.0	3.46	60	0.254	Dm: chebyshev; Sw: 20.0; No: procrustes; Pp: mean;	15
S(h)		Minkowski	100.0	0.0	3.17	60	0.182	Dm: euclidean; Sw: 20.0; No: procrustes; Pp: mean;	1
H	4	Minkowski	100.0	3.1	5.32	120	0.171	Dm: euclidean; Sw: 10.0; No: minmax; Pp: mean;	30
H(h)		Minkowski	100.0	7.3	4.01	60	0.096	Dm: sqeuclidean; Sw: 10.0; No: z-score; Pp: mean;	30
V		Minkowski	100.0	2.1	3.35	30	0.553	Dm: cityblock; No: z-score; Pp: mean;	30
V(h)		Minkowski	100.0	0.0	3.62	60	0.231	Dm: chebyshev; No: z-score; Pp: mean;	15

Table A.6: Top performing algorithms for the Hausdorff algorithm that achieve a 100% true positive rate, whilst minimising both false positives and average activation time for the circle (C), square (S), horizontal (H), vertical (V), and their harmonic (h) variants.

Shape	Speed	Algorithm	TP Rate	FP Rate	Activation time (s)	Buffer size	Threshold	Parameters	Posthoc
C		Hausdorff	100.0	0.0	2.53	15	0.445	Sw: 20.0; No: procrustes;	30
S		Hausdorff	100.0	0.0	3.25	45	0.782	Sw: 10.0; No: procrustes;	30
S(h)		Hausdorff	100.0	0.0	3.20	45	0.489	Sw: 20.0; No: procrustes;	15
H	2	Hausdorff	100.0	8.3	3.72	60	0.494	Sw: 20.0; No: z-score;	30
H(h)		Hausdorff	100.0	4.2	3.98	60	0.440	Sw: 20.0; No: z-score;	30
V		Hausdorff	100.0	8.3	3.85	60	0.459	No: z-score;	30
V(h)		Hausdorff	100.0	8.3	5.45	120	0.702	Sw: 20.0; No: procrustes;	30
C		Hausdorff	100.0	0.0	3.32	45	0.391	Sw: 20.0; No: z-score;	15
S		Hausdorff	100.0	0.0	3.89	60	0.731	Sw: 10.0; No: z-score;	30
S(h)		Hausdorff	100.0	0.0	3.68	45	0.306	Sw: 20.0; No: procrustes;	15
H	4	Hausdorff	100.0	14.6	4.83	90	0.593	No: z-score;	30
H(h)		Hausdorff	100.0	14.6	4.07	60	0.655	No: z-score;	30
V		Hausdorff	100.0	35.4	3.29	30	0.459	No: procrustes;	15
V(h)		Hausdorff	100.0	25.0	2.71	30	0.788	No: z-score;	30

Table A.7: Top performing algorithms for the dynamic time warping algorithm that achieve a 100% true positive rate, whilst minimising both false positives and average activation time for the circle (C), square (S), horizontal (H), vertical (V), and their harmonic (h) variants.

Shape	Speed	Algorithm	TP Rate	FP Rate	Activation time (s)	Buffer size	Threshold	Parameters	Posthoc
C		DTW	100.0	0.0	2.83	30	6.388	Dm: euclidean; δ: 10.0; Pa: 0; No: procrustes;	30
S		DTW	100.0	0.0	3.31	45	8.063	Dm: sqeuclidean; δ: 20.0; Pa: 10; No: procrustes;	30
S(h)		DTW	100.0	0.0	3.54	60	29.928	Dm: euclidean; δ: 10.0; Pa: 10; No: z-score;	30
H	2	DTW	100.0	2.1	4.41	90	15.114	Dm: euclidean; δ: 10.0; Pa: 0; No: minmax; AS:	30
H(h)		DTW	100.0	0.0	4.56	90	2.947	Dm: sqeuclidean; δ: 10.0; Pa: 0; No: minmax;	30
V		DTW	100.0	5.2	5.12	120	11.545	Dm: sqeuclidean; δ: 20.0; Pa: 0; No: z-score;	15
V(h)		DTW	100.0	3.1	4.62	90	8.442	Dm: sqeuclidean; δ: 10.0; Pa: 10; No: z-score;	30
C		DTW	100.0	0.0	3.52	45	7.416	Dm: chebyshev; δ: inf; Pa: 0; No: procrustes;	30
S		DTW	100.0	0.0	4.21	90	35.198	Dm: cityblock; δ: 10.0; Pa: 10; No: z-score;	15
S(h)		DTW	100.0	0.0	3.85	60	10.655	Dm: sqeuclidean; δ: 10.0; Pa: 0; No: z-score;	30
H	4	DTW	100.0	14.6	3.62	60	8.757	Dm: chebyshev; δ: 10.0; Pa: 0; No: minmax;	30
H(h)		DTW	100.0	12.5	3.72	60	10.592	Dm: cityblock; δ: 10.0; Pa: 10; No: minmax;	30
V		DTW	100.0	5.2	3.31	30	3.436	Dm: sqeuclidean; δ: inf; Pa: 0; No: z-score;	30
V(h)		DTW	100.0	6.2	5.45	120	3.533	Dm: sqeuclidean; δ: 10.0; Pa: 0; No: minmax;	30

Table A.8: Top performing algorithms for the longest common subsequence algorithm that achieve a 100% true positive rate, whilst minimising both false positives and average activation time for the circle (C), square (S), horizontal (H), vertical (V), and their harmonic (h) variants.

Shape	Speed	Algorithm	TP Rate	FP Rate	Activation time (s)	Buffer size	Threshold	Parameters	Posthoc
C		LCS	100.0	0.0	2.30	30	1.000	Dm: sqeuclidean; ε: 0.3; Pa: 10; δ: 20.0; No: procrustes;	15
S		LCS	100.0	0.0	2.67	30	1.000	Dm: euclidean; Pa: 10; δ: 10.0; No: procrustes;	30
S(h)		LCS	100.0	0.0	2.57	30	1.000	Dm: chebyshev; Pa: 10; δ: 10.0; No: procrustes;	30
H	2	LCS	100.0	2.1	3.02	45	0.733	Dm: cityblock; ε: 0.2; Pa: 10; δ: inf; No: minmax;	30
H(h)		LCS	100.0	1.0	3.40	45	0.978	Dm: cityblock; ε: 0.3; Pa: 10; δ: 10.0; No: minmax;	30
V		LCS	100.0	0.0	3.43	90	0.733	Dm: euclidean; ε: 0.3; Pa: 0; δ: inf; No: z-score;	1
V(h)		LCS	100.0	4.2	4.15	90	0.533	Dm: cityblock; ε: 0.3; Pa: 0; δ: 20.0; No: z-score;	30
C		LCS	100.0	0.0	2.86	45	0.733	Dm: chebyshev; ε: 0.2; Pa: 0; δ: inf; No: procrustes;	15
S		LCS	100.0	0.0	3.20	60	1.000	Dm: sqeuclidean; ε: 0.3; Pa: 20; δ: inf; No: z-score;	1
S(h)		LCS	100.0	0.0	3.17	60	1.000	Dm: sqeuclidean; ε: 0.3; Pa: 20; δ: inf; No: z-score;	1
H	4	LCS	100.0	2.1	4.34	90	0.789	Dm: euclidean; ε: 0.2; Pa: 10; δ: inf; No: procrustes;	15
H(h)		LCS	100.0	10.4	3.39	60	0.767	Dm: euclidean; ε: 0.2; Pa: 10; δ: 20.0; No: z-score;	1
V		LCS	100.0	2.1	4.41	90	0.756	Dm: cityblock; ε: 0.2; Pa: 10; δ: inf; No: procrustes;	15
V(h)		LCS	100.0	2.1	3.21	45	0.978	Dm: cityblock; ε: 0.3; Pa: 0; δ: 10.0; No: minmax;	15

Bibliography

[1] 2017. Table RAS50001, Reported Road Casualties Great Britain: 2016. (2017). [Cited on page 27]

[2] Johnny Accot and Shumin Zhai. 2002. More Than Dotting the I's — Foundations for Crossing-based Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '02)*. ACM, New York, NY, USA, 73–80. DOI:http://dx.doi.org/10.1145/503376.503390 [Cited on page 59]

[3] Christopher Ackad, Andrew Clayphan, Martin Tomitsch, and Judy Kay. 2015. An In-the-wild Study of Learning Mid-air Gestures to Browse Hierarchical Information at a Large Interactive Public Display. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. ACM, New York, NY, USA, 1227–1238. DOI: http://dx.doi.org/10.1145/2750858.2807532 [Cited on page 11]

[4] David Ahlström, Khalad Hasan, and Pourang Irani. 2014. Are You Comfortable Doing That?: Acceptance Studies of Around-device Gestures in and for Public Settings. In *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services (MobileHCI '14)*. ACM, New York, NY, USA, 193–202. DOI:http://dx.doi.org/10.1145/2628363.2628381 [Cited on page 2]

[5] Roland Aigner, Daniel Wigdor, Hrvoje Benko, Michael Haller, David Lindbauer, Alexandra Ion, Shengdong Zhao, and JTKV Koh. 2012. Understanding mid-air hand gestures: A study of human preferences in usage of gesture types for hci. *Microsoft Research TechReport MSR-TR-2012-111* 2 (2012). [Cited on pages 8, 9, and 10]

[6] Ahmad Akl, Chen Feng, and Shahrokh Valaee. 2011. A novel accelerometer-based gesture recognition system. *IEEE Transactions on Signal Processing* 59, 12 (2011), 6197–6205. [Cited on page 23]

[7] Fraser Anderson and Walter F Bischof. 2013. Learning and performance with gesture guides. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1109–1118. [Cited on page 13]

[8] Rowel Atienza, Ryan Blonna, Maria Isabel Saludares, Joel Casimiro, and Vivencio Fuentes. 2016. Interaction techniques using head gaze for virtual reality. In *2016 IEEE Region 10 Symposium (TENSYMP)*. IEEE, 110–114. [Cited on page 27]

[9] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. 2009. Visual tracking with online multiple instance learning. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 983–990. [Cited on page 62]

[10] Rafael Ballagas, Michael Rohs, and Jennifer G. Sheridan. 2005. Sweep and Point and Shoot: Phonecam-based Interactions for Large Public Displays. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems (CHI EA '05)*. ACM, New York, NY, USA, 1200–1203. DOI: http://dx.doi.org/10.1145/1056808.1056876 [Cited on page 58]

[11] Amartya Banerjee, Jesse Burstyn, Audrey Girouard, and Roel Vertegaal. 2011. Pointable: an in-air pointing technique to manipulate out-of-reach targets on tabletops. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. ACM, 11–20. [Cited on page 15]

[12] Amartya Banerjee, Jesse Burstyn, Audrey Girouard, and Roel Vertegaal. 2012. MultiPoint: Comparing Laser and Manual Pointing As Remote Input in Large Display Interactions. *Int. J. Hum.-Comput. Stud.* 70, 10 (Oct. 2012), 690–702. DOI:http://dx.doi.org/10.1016/j.ijhcs.2012.05.009 [Cited on pages 14 and 15]

[13] Graham R Barnes. 2011. Ocular pursuit movements. *The Oxford Handbook of Eye Movements* (2011), 115–132. [Cited on page 17]

[14] Olivier Bau and Wendy E Mackay. 2008. OctoPocus: a dynamic guide for learning gesture-based command sets. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*. ACM, 37–46. [Cited on pages 12 and 13]

[15] Thomas Baudel and Michel Beaudouin-Lafon. 1993. Charade: remote control of objects using free-hand gestures. *Commun. ACM* 36, 7 (1993), 28–35. [Cited on pages 10, 11, and 23]

[16] AMV BBDO. 2011. Choose a Different Ending. Video. (23 November 2011). Retrieved September 10, 2016 from https://www.youtube.com/watch?v=Ig5OUQcqLTg. [Cited on page 49]

[17] Peter Jan Beek. 1989. *Juggling dynamics*. Free University Press. [Cited on page 17]

[18] Michael Beigl. 1999. Point & Click—Interaction in Smart Environments. In *Handheld and Ubiquitous Computing (Lecture Notes in Computer Science)*, Hans. Gellersen (Ed.), Vol. 1707. Springer Berlin Heidelberg, 311–313. DOI:http://dx.doi.org/10.1007/3-540-48157-5_31 [Cited on page 26]

[19] Victoria Bellotti, Maribeth Back, W. Keith Edwards, Rebecca E. Grinter, Austin Henderson, and Cristina Lopes. 2002. Making Sense of Sensing Systems: Five Questions for Designers and Researchers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '02)*. ACM, New York, NY, USA, 415–422. DOI:http://dx.doi.org/10.1145/503376.503450 [Cited on pages 2 and 117]

[20] Serge Belongie, Jitendra Malik, and Jan Puzicha. 2002. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 4 (2002), 509–522. [Cited on page 24]

[21] Peter Bennett, Stuart Nolan, Ved Uttamchandani, Michael Pages, Kirsten Cater, and Mike Fraser. 2015. Resonant Bits: Harmonic Interaction with Virtual Pendulums. In *Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction (TEI '15)*. ACM, New York, NY, USA, 49–52. DOI:http://dx.doi.org/10.1145/2677199.2680569 [Cited on page 16]

[22] François Bérard. 1999. The Perceptual Window: Head Motion as a New Input Stream.. In *INTERACT*. Citeseer, 238–237. [Cited on page 14]

[23] Yannick Bernaerts, Matthias Druwé, Sebastiaan Steensels, Jo Vermeulen, and Johannes Schöning. 2014. The office smartwatch: development and design of a smartwatch app to digitally augment interactions in an office environment. In *Proceedings of the 2014 companion publication on Designing interactive systems*. ACM, 41–44. [Cited on page 23]

[24] Ana M Bernardos, David Gómez, and José R Casar. 2016. A Comparison of Head Pose and Deictic Pointing Interaction Methods for Smart Environments. *International Journal of Human-Computer Interaction* 32, 4 (2016), 325–351. [Cited on pages 14, 15, and 75]

[25] Donald J Berndt and James Clifford. 1994. Using dynamic time warping to find patterns in time series.. In *KDD workshop*, Vol. 10. Seattle, WA, 359–370. [Cited on page 96]

[26] Anastasia Bezerianos and Ravin Balakrishnan. 2005. View and Space Management on Large Displays. *IEEE Comput. Graph. Appl.* 25, 4 (July 2005), 34–43. DOI:http://dx.doi.org/10.1109/MCG.2005.92 [Cited on page 27]

[27] Renaud Blanch, Yves Guiard, and Michel Beaudouin-Lafon. 2004. Semantic Pointing: Improving Target Acquisition with Control-display Ratio Adaptation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '04)*. ACM, New York, NY, USA, 519–526. DOI:http://dx.doi.org/10.1145/985692.985758 [Cited on pages 14 and 58]

[28] Florian Block, Michael Haller, Hans Gellersen, Carl Gutwin, and Mark Billinghurst. 2008. VoodooSketch: Extending Interactive Surfaces with Adaptable Interface Palettes. In *Proceedings of the 2Nd International Conference on Tangible and Embedded Interaction (TEI '08)*. ACM, New

York, NY, USA, 55–58. DOI:http://dx.doi.org/10.1145/1347390.1347404 [Cited on page 14]

[29] Richard A. Bolt. 1980. "Put-that-there": Voice and Gesture at the Graphics Interface. *SIGGRAPH Comput. Graph.* 14, 3 (July 1980), 262–270. DOI:http://dx.doi.org/10.1145/965105.807503 [Cited on pages 9, 11, and 14]

[30] Sebastian Boring, Dominikus Baur, Andreas Butz, Sean Gustafson, and Patrick Baudisch. 2010. Touch Projector: Mobile Interaction Through Video. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. ACM, New York, NY, USA, 2287–2296. DOI:http://dx.doi.org/10.1145/1753326.1753671 [Cited on page 58]

[31] Jean-Yves Bouguet. 2000. Pyramidal implementation of the Lucas Kanade feature tracker. *Intel Corporation, Microprocessor Research Labs* (2000). [Cited on pages 33 and 61]

[32] Andrew Bragdon, Robert Zeleznik, Brian Williamson, Timothy Miller, and Joseph J LaViola Jr. 2009. GestureBar: improving the approachability of gesture-based interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2269–2278. [Cited on page 12]

[33] Stephen A Brewster. 2002. Non-speech auditory output. *The human-computer interaction handbook* (2002), 220–239. [Cited on page 9]

[34] Karl Bringmann. 2014. Why walking the dog takes time: Frechet distance has no strongly subquadratic algorithms unless SETH fails. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*. IEEE, 661–670. [Cited on page 98]

[35] Michelle A Brown, Wolfgang Stuerzlinger, and others. 2014. The performance of un-instrumented in-air pointing. In *Proceedings of Graphics Interface 2014*. Canadian Information Processing Society, 59–66. [Cited on page 75]

[36] Martinus J Buekers, Hedwig P Bogaerts, Stephan P Swinnen, and Werner F Helsen. 2000. The synchronization of human arm movements to external events. *Neuroscience Letters* 290, 3 (2000), 181–184. [Cited on pages 17 and 78]

[37] Jack Callahan, Don Hopkins, Mark Weiser, and Ben Shneiderman. 1988. An empirical comparison of pie vs. linear menus. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 95–100. [Cited on page 13]

[38] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2018. OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields. In *arXiv preprint arXiv:1812.08008*. [Cited on pages 24, 80, and 81]

[39] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. 2017. Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. In *CVPR*. [Cited on pages 24, 80, and 81]

[40] Marcus Carter, Eduardo Velloso, John Downs, Abigail Sellen, Kenton O'Hara, and Frank Vetere. 2016. PathSync: Multi-User Gestural Interaction with Touchless Rhythmic Path Mimicry. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 13. DOI:http://dx.doi.org/10.1145/2858036.2858284 [Cited on pages 3, 10, 14, 16, 20, 26, 28, 29, 34, 36, 78, 81, 93, 94, 108, 116, and 118]

[41] Fernando Cassola, Leonel Morgado, Fausto de Carvalho, Hugo Paredes, Benjamim Fonseca, and Paulo Martins. 2014. Online-Gym: a 3D virtual gymnasium using Kinect interaction. *Procedia Technology* 13 (2014), 130–138. [Cited on page 27]

[42] Tanja Ceux, Gilles Montagne, and Martinus J Buekers. 2010. The integration of temporally shifted visual feedback in a synchronization task: The role of perceptual stability in a visuo-proprioceptive conflict situation. *Human movement science* 29, 6 (2010), 893–909. [Cited on pages 17 and 78]

[43] Charlene Chamberlain, Jill P Morford, and Rachel I Mayberry. 1999. *Language acquisition by eye*. Psychology Press. [Cited on page 9]

[44] Lei Chen, M Tamer Özsu, and Vincent Oria. 2005. Robust and fast similarity search for moving object trajectories. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*. ACM, 491–502. [Cited on page 106]

[45] Ming-yu Chen, Lily Mummert, Padmanabhan Pillai, Alexander Hauptmann, and Rahul Sukthankar. 2010. Controlling Your TV with Gestures. In *Proceedings of the International Conference on Multimedia Information Retrieval (MIR '10)*. ACM, New York, NY, USA, 405–408. DOI: http://dx.doi.org/10.1145/1743384.1743453 [Cited on pages 26 and 64]

[46] Yizong Cheng. 1995. Mean shift, mode seeking, and clustering. *IEEE transactions on pattern analysis and machine intelligence* 17, 8 (1995), 790–799. [Cited on page 25]

[47] YL Chi, SK Ong, ML Yuan, and AYC Nee. 2007. Wearable interface for the physical disabled. In *Proceedings of the 1st international convention on Rehabilitation engineering & assistive technology: in conjunction with 1st Tan Tock Seng Hospital Neurorehabilitation Meeting*. ACM, 28–32. [Cited on page 23]

[48] Ngip-Khean Chuan and Ashok Sivaji. 2012. Combining eye gaze and hand tracking for pointer control in HCI: Developing a more robust and accurate interaction system for pointer positioning and clicking. In *Humanities, Science and Engineering (CHUSER), 2012 IEEE Colloquium on*. IEEE, 172–176. [Cited on page 75]

[49] Philip R Cohen, Mary Dalrymple, Douglas B Moran, FC Pereira, and Joseph W Sullivan. 1989. Synergistic use of direct manipulation and natural language. In *ACM SIGCHI Bulletin*, Vol. 20. ACM, 227–233. [Cited on page 118]

[50] Philip R Cohen, Michael Johnston, David McGee, Sharon L Oviatt, Jay Pittman, Ira A Smith, Liang Chen, and Josh Clow. 1997. QuickSet: Multimodal Interaction for Distributed Applications.. In *ACM Multimedia*, Vol. 97. 31–40. [Cited on page 8]

[51] Thomas Cover and Peter Hart. 1967. Nearest neighbor pattern classification. *IEEE transactions on information theory* 13, 1 (1967), 21–27. [Cited on page 25]

[52] Martin Danelljan, Fahad Shahbaz Khan, Michael Felsberg, and Joost Van de Weijer. 2014. Adaptive color attributes for real-time visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1090–1097. [Cited on page 62]

[53] Nasser H Dardas and Nicolas D Georganas. 2011. Real-time hand gesture detection and recognition using bag-of-features and support vector machine techniques. *IEEE Transactions on Instrumentation and measurement* 60, 11 (2011), 3592–3607. [Cited on page 25]

[54] Hoang Anh Dau, Eamonn Keogh, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, Yanping, Bing Hu, Nurjahan Begum, Anthony Bagnall, Abdullah Mueen, Gustavo Batista, and Hexagon-ML. 2018. The UCR Time Series Classification Archive. (October 2018). `https://www.cs.ucr.edu/~eamonn/time_series_data_2018/`. [Cited on page 93]

[55] William Delamare, Céline Coutrix, and Laurence Nigay. 2015. Designing guiding systems for gesture-based interaction. In *Proceedings of the 7th ACM SIGCHI Symposium on Engineering Interactive Computing Systems*. ACM, 44–53. [Cited on page 13]

[56] William Delamare, Thomas Janssoone, Céline Coutrix, and Laurence Nigay. 2016. Designing 3D gesture guidance: visual feedback and feedforward design options. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*. ACM, 152–159. [Cited on page 13]

[57] Artem Dementyev and Joseph A Paradiso. 2014. WristFlex: low-power gesture input with wrist-worn pressure sensors. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*. ACM, 161–166. [Cited on page 23]

[58] Katie Derthick, James Scott, Nicolas Villar, and Christian Winkler. 2013. Exploring Smartphone-based Web User Interfaces for Appliances. In *Proceedings of the 15th International Conference on Human-computer Interaction with Mobile Devices and Services (MobileHCI '13)*. ACM, New York, NY, USA, 227–236. `DOI:http://dx.doi.org/10.1145/2493190.2493239` [Cited on page 26]

[59] Murtaza Dhuliawala, Juyoung Lee, Junichi Shimizu, Andreas Bulling, Kai Kunze, Thad Starner, and Woontack Woo. 2016. Smooth eye movement interaction using EOG glasses. In *Proceedings of the 18th ACM International Conference on Multimodal Interaction*. ACM, 307–311. [Cited on page 19]

[60] Gavin Doherty and Mieke Massink. 1999. Continuous interaction and human control. In *Proceedings of the XVIII European Annual Conference on Human Decision Making and Manual Control*. 80–96. [Cited on page 2]

[61] Tanja Döring, Dagmar Kern, Paul Marshall, Max Pfeiffer, Johannes Schöning, Volker Gruhn, and Albrecht Schmidt. 2011. Gestural interaction on the steering wheel: reducing the visual demand. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 483–492. [Cited on page 11]

[62] Ashley Dover, G. Michael Poor, Darren Guinness, and Alvin Jude. 2016. Improving Gestural Interaction With Augmented Cursors. In *Proceedings of the 2016 Symposium on Spatial User Interaction (SUI '16)*. ACM, New York, NY, USA, 135–138. DOI:http://dx.doi.org/10.1145/2983310.2985765 [Cited on page 74]

[63] Heiko Drewes, Mohamed Khamis, and Florian Alt. 2018. Smooth Pursuit Target Speeds and Trajectories. In *Proceedings of the 17th International Conference on Mobile and Ubiquitous Multimedia*. ACM, 139–146. [Cited on pages 82 and 106]

[64] Heiko Drewes and Albrecht Schmidt. 2007. Interacting with the computer using gaze gestures. In *IFIP Conference on Human-Computer Interaction*. Springer, 475–488. [Cited on page 19]

[65] Lars C Ebert, G Hatch, MJ Thali, and Steffen Ross. 2013. Invisible touch?Control of a DICOM viewer with finger gestures using the Kinect depth camera. *Journal of Forensic Radiology and Imaging* 1, 1 (2013), 10–14. [Cited on page 27]

[66] Thomas Eiter and Heikki Mannila. 1994. *Computing discrete Fréchet distance*. Technical Report. Citeseer. [Cited on pages 97 and 98]

[67] Scott Elrod, Richard Bruce, Rich Gold, David Goldberg, Frank Halasz, William Janssen, David Lee, Kim McCall, Elin Pedersen, Ken Pier, John Tang, and Brent Welch. 1992. Liveboard: A Large Interactive Display Supporting Group Meetings, Presentations, and Remote Collaboration. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '92)*. ACM, New York, NY, USA, 599–607. DOI:http://dx.doi.org/10.1145/142750.143052 [Cited on page 14]

[68] Douglas C. Engelbart and William K. English. 1968. A Research Center for Augmenting Human Intellect. In *Proceedings of the December 9-11, 1968, Fall Joint Computer Conference, Part I (AFIPS '68 (Fall, part I))*. ACM, New York, NY, USA, 395–410. DOI:http://dx.doi.org/10.1145/1476589.1476645 [Cited on page 8]

[69] Augusto Esteves, Eduardo Velloso, Andreas Bulling, and Hans Gellersen. 2015. Orbits: Gaze Interaction for Smart Watches Using Smooth Pursuit Eye Movements. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. ACM, New

York, NY, USA, 457–466. DOI:http://dx.doi.org/10.1145/2807442.2807499 [Cited on pages 3, 20, 26, 28, 29, 30, 34, 59, and 81]

[70] Augusto Esteves, David Verweij, Liza Suraiya, Rasel Islam, Youryang Lee, and Ian Oakley. 2017. SmoothMoves: Smooth Pursuits Head Movements for Augmented Reality. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*. ACM, 167–178. [Cited on pages 21, 27, and 94]

[71] D Gareth Evans, Roger Drew, and Paul Blenkhorn. 2000. Controlling mouse pointer position using an infrared head-operated joystick. *IEEE Transactions on rehabilitation engineering* 8, 1 (2000), 107–117. [Cited on page 23]

[72] Gunnar Farnebäck. 2003. Two-frame Motion Estimation Based on Polynomial Expansion. In *Proceedings of the 13th Scandinavian Conference on Image Analysis (SCIA'03)*. Springer-Verlag, Berlin, Heidelberg, 363–370. http://dl.acm.org/citation.cfm?id=1763974.1764031 [Cited on pages 33 and 62]

[73] Jean-Daniel Fekete, Niklas Elmqvist, and Yves Guiard. 2009. Motion-pointing: Target Selection Using Elliptical Motions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 289–298. DOI:http://dx.doi.org/10.1145/1518701.1518748 [Cited on pages 1, 2, 3, 10, 16, 18, 26, 28, 29, 94, and 108]

[74] Martin A. Fischler and Robert C. Bolles. 1981. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM* 24, 6 (June 1981), 381–395. DOI:http://dx.doi.org/10.1145/358669.358692 [Cited on page 34]

[75] Paul M Fitts and Michael I Posner. 1967. Human performance. (1967). [Cited on page 9]

[76] Andrew Fitzgibbon, Maurizio Pilu, and Robert B Fisher. 1999. Direct least square fitting of ellipses. *IEEE Transactions on pattern analysis and machine intelligence* 21, 5 (1999), 476–480. [Cited on page 83]

[77] George W. Fitzmaurice, Hiroshi Ishii, and William A. S. Buxton. 1995. Bricks: Laying the Foundations for Graspable User Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '95)*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 442–449. DOI:http://dx.doi.org/10.1145/223904.223964 [Cited on page 14]

[78] David Fono and Roel Vertegaal. 2005. EyeWindows: evaluation of eye-controlled zooming windows for focus selection. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 151–160. [Cited on page 19]

[79] Alexandre RJ François and Gérard G Medioni. 1999. Adaptive color background modeling for real-time segmentation of video streams. In *Proceedings of the International Conference on Imaging Science, Systems, and Technology*, Vol. 1. 227–232. [Cited on page 25]

[80] Dustin Freeman, Hrvoje Benko, Meredith Ringel Morris, and Daniel Wigdor. 2009. ShadowGuides: visualizations for in-situ learning of multi-touch and whole-hand gestures. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*. ACM, 165–172. [Cited on pages 12, 13, and 78]

[81] Euan Freeman, Stephen Brewster, and Vuokko Lantz. 2016. Do that, there: an interaction technique for addressing in-air gesture systems. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 2319–2331. [Cited on pages 28, 36, 78, 81, 111, and 112]

[82] Euan Freeman, Gareth Griffiths, and Stephen A Brewster. 2017. Rhythmic micro-gestures: discreet interaction on-the-go. In *Proceedings of the 19th ACM International Conference on Multimodal Interaction*. ACM, 115–119. [Cited on pages 21 and 113]

[83] William T. Freeman and Craig D. Weissman. 1994. Television Control by Hand Gestures. (1994). [Cited on pages 24 and 26]

[84] Franca Garzotto and Matteo Valoriani. 2012. Don't touch the oven: motion-based touchless interaction with household appliances. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*. ACM, 721–724. [Cited on page 11]

[85] William W Gaver. 1997. Auditory interfaces. In *Handbook of human-computer interaction*. Elsevier, 1003–1041. [Cited on page 9]

[86] Dariu M Gavrila. 1999. The visual analysis of human movement: A survey. *Computer vision and image understanding* 73, 1 (1999), 82–98. [Cited on page 24]

[87] Emilien Ghomi, Guillaume Faure, Stéphane Huot, Olivier Chapuis, and Michel Beaudouin-Lafon. 2012. Using rhythmic patterns as an input method. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1253–1262. [Cited on page 16]

[88] Emilien Ghomi, Stéphane Huot, Olivier Bau, Michel Beaudouin-Lafon, and Wendy E Mackay. 2013. Arpège: learning multitouch chord gestures vocabularies. In *Proceedings of the 2013 ACM international conference on Interactive tabletops and surfaces*. ACM, 209–218. [Cited on page 13]

[89] Ivan Golod, Felix Heidrich, Christian Möllering, and Martina Ziefle. 2013. Design principles of hand gesture interfaces for microinteractions. In *Proceedings of the 6th International Conference on Designing Pleasurable Products and Interfaces*. ACM, 11–20. [Cited on page 11]

[90] Helmut Grabner, Michael Grabner, and Horst Bischof. 2006. Real-time tracking via on-line boosting.. In *Bmvc*, Vol. 1. 6. [Cited on page 62]

[91] Chauncey Graetzel, Terry Fong, Sebastien Grange, and Charles Baur. 2004. A non-contact mouse for surgeon-computer interaction. *Technology and Health Care* 12, 3 (2004), 245–257. [Cited on page 27]

[92] Saul Greenberg and Michael Boyle. 2002. Customizable Physical Interfaces for Interacting with Conventional Applications. In *Proceedings of the 15th Annual ACM Symposium on User Interface Software and Technology (UIST '02)*. ACM, New York, NY, USA, 31–40. DOI:http://dx.doi.org/10.1145/571985.571991 [Cited on page 14]

[93] Yves Guiard. 1987. Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model. *Journal of motor behavior* 19, 4 (1987), 486–517. [Cited on page 22]

[94] Yves Guiard. 1993. On Fitts's and Hooke's laws: Simple harmonic movement in upper-limb cyclical aiming. *Acta psychologica* 82, 1-3 (1993), 139–159. [Cited on pages 6, 18, 78, and 114]

[95] Yves Guiard. 2009. The Problem of Consistency in the Design of Fitts' Law Experiments: Consider Either Target Distance and Width or Movement Form and Scale. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 1809–1818. DOI:http://dx.doi.org/10.1145/1518701.1518980 [Cited on page 71]

[96] Yves Guiard, Julien Gori, Quentin Roy, and Olivier Rioul. 2017. Not Just Pointing: Shannon's Information Theory as a General Tool for Performance Evaluation of Input Techniques. (2017). [Cited on page 44]

[97] Darren Guinness, Alvin Jude, G. Michael Poor, and Ashley Dover. 2015. Models for Rested Touchless Gestural Interaction. In *Proceedings of the 3rd ACM Symposium on Spatial User Interaction (SUI '15)*. ACM, New York, NY, USA, 34–43. DOI:http://dx.doi.org/10.1145/2788940.2788948 [Cited on pages 15 and 26]

[98] Aakar Gupta, Antony Irudayaraj, Vimal Chandran, Goutham Palaniappan, Khai N. Truong, and Ravin Balakrishnan. 2016. Haptic Learning of Semaphoric Finger Gestures. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. ACM, New York, NY, USA, 219–226. DOI:http://dx.doi.org/10.1145/2984511.2984558 [Cited on page 13]

[99] Jan Hess, Guy Küstermann, and Volkmar Pipek. 2008. Premote: A User Customizable Remote Control. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems (CHI EA '08)*. ACM, New York, NY, USA, 3279–3284. DOI:http://dx.doi.org/10.1145/1358628.1358844 [Cited on page 26]

[100] Jahani F Hessam, Massimo Zancanaro, Manolya Kavakli, and Mark Billinghurst. 2017. Towards optimization of mid-air gestures for in-vehicle interactions. In *Proceedings of the 29th Australian Conference on Computer-Human Interaction*. ACM, 126–134. [Cited on page 27]

[101] Juan David Hincapié-Ramos, Xiang Guo, and Pourang Irani. 2014. The Consumed Endurance Workbench: A Tool to Assess Arm Fatigue During Mid-air Interactions. In *Proceedings of the 2014 Companion Publication on Designing Interactive Systems (DIS Companion '14)*. ACM, New York, NY, USA, 109–112. DOI:http://dx.doi.org/10.1145/2598784.2602795 [Cited on pages 2, 15, 26, and 111]

[102] Ken Hinckley. 2003. Synchronous Gestures for Multiple Persons and Computers. In *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology (UIST '03)*. ACM, New York, NY, USA, 149–158. DOI:http://dx.doi.org/10.1145/964696.964713 [Cited on page 22]

[103] Ken Hinckley, Patrick Baudisch, Gonzalo Ramos, and Francois Guimbretiere. 2005. Design and analysis of delimiters for selection-action pen gesture phrases in scriboli. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 451–460. [Cited on page 11]

[104] Ken Hinckley, Randy Pausch, John C. Goble, and Neal F. Kassell. 1994. Passive Real-world Interface Props for Neurosurgical Visualization. In *Conference Companion on Human Factors in Computing Systems (CHI '94)*. ACM, New York, NY, USA, 232–. DOI:http://dx.doi.org/10.1145/259963.260443 [Cited on page 14]

[105] Ken Hinckley, Randy Pausch, Dennis Proffitt, and Neal F Kassell. 1998. Two-handed virtual manipulation. *ACM Transactions on Computer-Human Interaction (TOCHI)* 5, 3 (1998), 260–302. [Cited on page 14]

[106] Ken Hinckley and Daniel Wigdor. 2002. Input technologies and techniques. *The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications* (2002), 151–168. [Cited on page 8]

[107] Lars Erik Holmquist, Friedemann Mattern, Bernt Schiele, Petteri Alahuhta, Michael Beigl, and Hans-Werner Gellersen. 2001. Smart-Its Friends: A Technique for Users to Easily Establish Connections Between Smart Artefacts. In *Proceedings of the 3rd International Conference on Ubiquitous Computing (UbiComp '01)*. Springer-Verlag, London, UK, UK, 116–122. http://dl.acm.org/citation.cfm?id=647987.741340 [Cited on page 22]

[108] Michael J Hove, Michael J Spivey, and Carol L Krumhansl. 2010. Compatibility of motion facilitates visuomotor synchronization. *Journal of Experimental Psychology: Human Perception and Performance* 36, 6 (2010), 1525. [Cited on page 17]

[109] Yi-Ta Hsieh, Antti Jylhä, Valeria Orso, Luciano Gamberini, and Giulio Jacucci. 2016. Designing a willing-to-use-in-public hand gestural interaction technique for smart glasses. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 4203–4215. [Cited on page 27]

[110] Daniel P Huttenlocher, Gregory A Klanderman, and William J Rucklidge. 1993. Comparing images using the Hausdorff distance. *IEEE Transactions on pattern analysis and machine intelligence* 15, 9 (1993), 850–863. [Cited on page 97]

[111] Inwook Hwang, Hyun-Cheol Kim, Jihun Cha, Chunghyun Ahn, Karam Kim, and Jong-Il Park. 2015. A gesture based tv control interface for visually impaired: Initial design and user study. In *Frontiers of Computer Vision (FCV), 2015 21st Korea-Japan Joint Workshop on*. IEEE, 1–5. [Cited on pages 26 and 64]

[112] Michael Isard and Andrew Blake. 1998. Condensation—conditional density propagation for visual tracking. *International journal of computer vision* 29, 1 (1998), 5–28. [Cited on page 25]

[113] Jana M Iverson, Olga Capirci, and M Cristina Caselli. 1994. From communication to language in two modalities. *Cognitive development* 9, 1 (1994), 23–43. [Cited on page 9]

[114] Robert JK Jacob. 1990. What you look at is what you get: eye movement-based interaction techniques. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 11–18. [Cited on page 19]

[115] Robert JK Jacob, Audrey Girouard, Leanne M Hirshfield, Michael S Horn, Orit Shaer, Erin Treacy Solovey, and Jamie Zigelbaum. 2008. Reality-based interaction: a framework for post-WIMP interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 201–210. [Cited on pages 1 and 113]

[116] Shahram Jalaliniya, Diako Mardanbeigi, Thomas Pederson, and Dan Witzner Hansen. 2014. Head and eye movement as pointing modalities for eyewear computers. In *2014 11th International Conference on Wearable and Implantable Body Sensor Networks Workshops*. IEEE, 50–53. [Cited on page 23]

[117] Rados Javanovic and Ian MacKenzie. 2010. MarkerMouse: mouse cursor control using a head-mounted marker. *Computers Helping People with Special Needs* (2010), 49–56. [Cited on pages 14 and 75]

[118] Soonmook Jeong, Jungdong Jin, Taehoun Song, Keyho Kwon, and Jae Wook Jeon. 2012. Single-camera dedicated television control system using gesture drawing. *IEEE Transactions on Consumer Electronics* 58, 4 (2012), 1129–1137. [Cited on pages 26 and 64]

[119] Rose Johnson, Kenton O'Hara, Abigail Sellen, Claire Cousins, and Antonio Criminisi. 2011. Exploring the potential for touchless interaction in image-guided interventional radiology. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 3323–3332. [Cited on page 27]

[120] Ricardo Jota, Miguel A Nacenta, Joaquim A Jorge, Sheelagh Carpendale, and Saul Greenberg. 2010. A comparison of ray pointing techniques for very large displays. In *Proceedings of graphics interface 2010*. Canadian Information Processing Society, 269–276. [Cited on page 15]

[121] Alvin Jude, G Michael Poor, and Darren Guinness. 2014a. An evaluation of touchless hand gestural interaction for pointing tasks with preferred and non-preferred hands. In *Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational*. ACM, 668–676. [Cited on page 14]

[122] Alvin Jude, G. Michael Poor, and Darren Guinness. 2014b. Personal Space: User Defined Gesture Space for GUI Interaction. In *CHI '14 Extended Abstracts on Human Factors in Computing Systems (CHI EA '14)*. ACM, New York, NY, USA, 1615–1620. DOI:http://dx.doi.org/10.1145/2559206.2581242 [Cited on pages 14 and 15]

[123] Raine Kajastila and Tapio Lokki. 2013. Eyes-free interaction with free-hand gestures and auditory menus. *International Journal of Human-Computer Studies* 71, 5 (2013), 627–640. [Cited on page 27]

[124] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. 2010. Forward-backward error: Automatic detection of tracking failures. In *Pattern recognition (ICPR), 2010 20th international conference on*. IEEE, 2756–2759. [Cited on pages 61 and 62]

[125] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. 2012. Tracking-learning-detection. *IEEE transactions on pattern analysis and machine intelligence* 34, 7 (2012), 1409–1422. [Cited on page 62]

[126] Rudolph Emil Kalman. 1960. A new approach to linear filtering and prediction problems. (1960). [Cited on page 25]

[127] Shaun K Kane, Jacob O Wobbrock, and Richard E Ladner. 2011. Usable gestures for blind people: understanding preference and performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 413–422. [Cited on page 11]

[128] Maria Karam and m. c. schraefel. 2005. A Taxonomy of Gestures in Human Computer Interactions. (2005). https://eprints.soton.ac.uk/261149/ [Cited on page 9]

[129] Joseph Jofish Kaye. 2004. Making Scents: aromatic output for HCI. *interactions* 11, 1 (2004), 48–61. [Cited on page 9]

[130] Adam Kendon. 1986. Current issues in the study of gesture. *The biological foundations of gestures: Motor and semiotic aspects* 1 (1986), 23–47. [Cited on page 9]

[131] Mohamed Khamis, Ozan Saltuk, Alina Hang, Katharina Stolz, Andreas Bulling, and Florian Alt. 2016. TextPursuits: using text for pursuits-based interaction and calibration on public displays. In

Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing. ACM, 274–285. [Cited on page 19]

[132] Rick Kjeldsen. 2006. Improvements in Vision-based Pointer Control. In *Proceedings of the 8th International ACM SIGACCESS Conference on Computers and Accessibility (Assets '06)*. ACM, New York, NY, USA, 189–196. DOI:http://dx.doi.org/10.1145/1168987.1169020 [Cited on page 14]

[133] Rick Kjeldsen and Jacob Hartman. 2001. Design Issues for Vision-based Computer Interaction Systems. In *Proceedings of the 2001 Workshop on Perceptive User Interfaces (PUI '01)*. ACM, New York, NY, USA, 1–8. DOI:http://dx.doi.org/10.1145/971478.971511 [Cited on page 11]

[134] Tiiu Koskela and Kaisa Väänänen-Vainio-Mattila. 2004. Evolution towards smart home environments: empirical evaluation of three user interfaces. *Personal and Ubiquitous Computing* 8, 3-4 (2004), 234–240. DOI:http://dx.doi.org/10.1007/s00779-004-0283-x [Cited on pages 26, 57, and 64]

[135] Georgios Kouroupetroglou, Alexandros Pino, Athanasios Balmpakakis, Dimitrios Chalastanis, Vasileios Golematis, Nikolaos Ioannou, and Ioannis Koutsoumpas. 2011. Using Wiimote for 2D and 3D pointing tasks: gesture performance evaluation. In *International Gesture Workshop*. Springer, 13–23. [Cited on page 75]

[136] Richard J Krauzlis. 2005. The control of voluntary eye movements: new perspectives. *The Neuroscientist* 11, 2 (2005), 124–137. [Cited on page 17]

[137] Per Ola Kristensson, Thomas Nicholson, and Aaron Quigley. 2012. Continuous recognition of one-handed and two-handed gestures using 3D full-body motion tracking sensors. In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces*. ACM, 89–92. [Cited on page 11]

[138] Myron W Krueger, Thomas Gionfriddo, and Katrin Hinrichsen. 1985. VIDEOPLACE?an artificial reality. In *ACM SIGCHI Bulletin*, Vol. 16. ACM, 35–40. [Cited on pages 1 and 24]

[139] Gordon Kurtenbach, Thomas P. Moran, and William Buxton. 1994. Contextual animation of gestural commands. In *Computer Graphics Forum*, Vol. 13. Wiley Online Library, 305–314. [Cited on pages 12 and 13]

[140] Mikko Kytö, Barrett Ens, Thammathip Piumsomboon, Gun A Lee, and Mark Billinghurst. 2018. Pinpointing: Precise Head-and Eye-Based Target Selection for Augmented Reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 81. [Cited on page 27]

[141] Claire Landmann, Sofia M Landi, Scott T Grafton, and Valeria Della-Maggiore. 2011. fMRI supports the sensorimotor theory of motor resonance. *PLoS One* 6, 11 (2011), e26859. [Cited on page 18]

[142] Ivan Laptev, Barbara Caputo, and others. 2004. Recognizing human actions: a local SVM approach. In *null*. IEEE, 32–36. [Cited on page 25]

[143] Susan J Lederman and Roberta L Klatzky. 2009. Haptic perception: A tutorial. *Attention, Perception, & Psychophysics* 71, 7 (2009), 1439–1459. [Cited on page 9]

[144] Taehee Lee and Tobias Hollerer. 2007. Handy AR: Markerless inspection of augmented reality objects using fingertip tracking. In *2007 11th IEEE International Symposium on Wearable Computers*. IEEE, 83–90. [Cited on page 27]

[145] Stephen R Levine and Susan F Ehrlich. 1995. The Freestyle system: a design perspective. In *Readings in Human–Computer Interaction*. Elsevier, 871–880. [Cited on page 8]

[146] Christophe Leys, Christophe Ley, Olivier Klein, Philippe Bernard, and Laurent Licata. 2013. Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of Experimental Social Psychology* 49, 4 (2013), 764–766. [Cited on page 83]

[147] Jaime Lien, Nicholas Gillian, M Emre Karagozler, Patrick Amihood, Carsten Schwesig, Erik Olson, Hakim Raja, and Ivan Poupyrev. 2016. Soli: Ubiquitous gesture sensing with millimeter wave radar. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 142. [Cited on page 25]

[148] Chiuhsiang Joe Lin, Sui-Hua Ho, and Yan-Jyun Chen. 2015. An investigation of pointing postures in a 3D stereoscopic environment. *Applied ergonomics* 48 (2015), 154–163. [Cited on pages 14 and 75]

[149] Jiayang Liu, Lin Zhong, Jehan Wickramasuriya, and Venu Vasudevan. 2009. uWave: Accelerometer-based personalized gesture recognition and its applications. *Pervasive and Mobile Computing* 5, 6 (2009), 657–675. [Cited on page 23]

[150] Logitech. 2010. Logitech Study Shows Multiple Remote Controls Hindering Entertainment Experiences Around the Globe. Press Release. (Nov 2010). `http://www.logitech.com/en-us/press/press-releases/7748` [Cited on page 26]

[151] Benjamin Long, Sue Ann Seah, Tom Carter, and Sriram Subramanian. 2014. Rendering volumetric haptic shapes in mid-air using ultrasound. *ACM Transactions on Graphics (TOG)* 33, 6 (2014), 181. [Cited on page 11]

[152] Bruce D. Lucas and Takeo Kanade. 1981. An iterative image registration technique with an application to stereo vision. In *In IJCAI81*. 674–679. [Cited on pages 33 and 61]

[153] Ewa Luger and Abigail Sellen. 2016. Like having a really bad PA: the gulf between user expectation and experience of conversational agents. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 5286–5297. [Cited on page 8]

[154] Alan Lukezic, Tomas Vojir, Luka ˇCehovin Zajc, Jiri Matas, and Matej Kristan. 2017. Discriminative correlation filter with channel and spatial reliability. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 6309–6318. [Cited on pages 25 and 81]

[155] Qingshan Luo, Xiaodong Kong, Guihua Zeng, and Jianping Fan. 2010. Human action detection via boosted local motion histograms. *Machine Vision and Applications* 21, 3 (2010), 377–389. [Cited on page 25]

[156] I. Scott MacKenzie, Tatu Kauppinen, and Miika Silfverberg. 2001. Accuracy Measures for Evaluating Computer Pointing Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '01)*. ACM, New York, NY, USA, 9–16. DOI:http://dx.doi.org/10.1145/365024.365028 [Cited on page 71]

[157] Yuichi Maki, Shingo Kagami, and Koichi Hashimoto. 2010. Accelerometer detection in a camera view based on feature point tracking. In *2010 IEEE/SICE International Symposium on System Integration*. IEEE, 448–453. [Cited on page 106]

[158] Sylvain Malacria, Eric Lecolinet, and Yves Guiard. 2010. Clutch-free panning and integrated pan-zoom control on touch-sensitive surfaces: the cyclostar approach. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2615–2624. [Cited on page 16]

[159] Rainer Malkewitz. 1998. Head Pointing and Speech Control As a Hands-free Interface to Desktop Computing. In *Proceedings of the Third International ACM Conference on Assistive Technologies (Assets '98)*. ACM, New York, NY, USA, 182–188. DOI:http://dx.doi.org/10.1145/274497.274531 [Cited on page 14]

[160] Paul Marshall, Richard Morris, Yvonne Rogers, Stefan Kreitmayer, and Matt Davies. 2011. Rethinking 'Multi-user': An In-the-wild Study of How Groups Approach a Walk-up-and-use Tabletop Interface. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 3033–3042. DOI:http://dx.doi.org/10.1145/1978942.1979392 [Cited on page 27]

[161] Pierre-François Marteau. 2008. Time warp edit distance with stiffness adjustment for time series matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31, 2 (2008), 306–318. [Cited on page 106]

[162] Fabrice Matulic and Daniel Vogel. 2018. Multiray: Multi-Finger Raycasting for Large Displays. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 245. [Cited on page 14]

[163] Dan Mauney, Jonathan Howarth, Andrew Wirtanen, and Miranda Capra. 2010. Cultural similarities and differences in user-defined gestures for touchscreen user interfaces. In *CHI'10 Extended Abstracts on Human Factors in Computing Systems*. ACM, 4015–4020. [Cited on page 11]

[164] Sébastien Maury, Sylvie Athénes, and Stéphane Chatty. 1999. Rhythmic menus: toward interaction based on rhythm. In *Conference on Human Factors in Computing Systems: CHI'99 extended abstracts on Human factors in computing systems*, Vol. 15. 254–255. [Cited on page 16]

[165] Keenan R May, Thomas M Gable, and Bruce N Walker. 2014. A multimodal air gesture interface for in vehicle menu navigation. In *Adjunct Proceedings of the 6th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*. ACM, 1–6. [Cited on page 27]

[166] Rachel I Mayberry and Elena Nicoladis. 2000. Gesture reflects language development: Evidence from bilingual children. *Current Directions in Psychological Science* 9, 6 (2000), 192–196. [Cited on page 9]

[167] Sven Mayer, Katrin Wolf, Stefan Schneegass, and Niels Henze. 2015. Modeling distant pointing for compensating systematic displacements. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 4165–4168. [Cited on page 15]

[168] Rene Mayrhofer and Hans Gellersen. 2009. Shake well before use: Intuitive and secure pairing of mobile devices. *Mobile Computing, IEEE Transactions on* 8, 6 (2009), 792–806. DOI:http://dx.doi.org/10.1109/TMC.2009.51 [Cited on pages 3 and 22]

[169] Suzanne P McKee and Ken Nakayama. 1984. The detection of motion in the peripheral visual field. *Vision research* 24, 1 (1984), 25–32. [Cited on page 112]

[170] David McNeill. 1992. *Hand and mind: What gestures reveal about thought*. University of Chicago press. [Cited on page 9]

[171] David McNeill. 2008. *Gesture and thought*. University of Chicago press. [Cited on page 9]

[172] Helena M Mentis, Kenton O'Hara, Abigail Sellen, and Rikin Trivedi. 2012. Interaction proxemics and image use in neurosurgery. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 927–936. [Cited on page 27]

[173] Courtney Messenbaugh. 2016. BMW's 7-Series 'gesture controls' work pretty well. (May 2016). https://eu.usatoday.com/story/money/cars/2016/05/16/bmws-7-series-gesture-controls-work-pretty-well/32613369/(accessed˜15/04/2019) [Cited on page 27]

[174] Andre Mewes, Bennet Hensen, Frank Wacker, and Christian Hansen. 2017. Touchless interaction with software in interventional radiology and surgery: a systematic literature review. *International journal of computer assisted radiology and surgery* 12, 2 (2017), 291–305. [Cited on page 27]

[175] Microsoft. 2014. Kinect for Windows | Human Interface Guidelines v2.0. (2014). [Cited on pages 15 and 36]

[176] Mark R Mine. 1995. Virtual environment interaction techniques. *UNC Chapel Hill CS Dept* (1995). [Cited on page 23]

[177] Pranav Mistry and Pattie Maes. 2009. SixthSense: a wearable gestural interface. In *ACM SIGGRAPH ASIA 2009 Sketches*. ACM, 11. [Cited on page 27]

[178] Meredith Ringel Morris, Jacob O Wobbrock, and Andrew D Wilson. 2010. Understanding users' preferences for surface gestures. In *Proceedings of graphics interface 2010*. Canadian Information Processing Society, 261–268. [Cited on page 12]

[179] Leigh A Mrotek and John F Soechting. 2007. Target interception: hand–eye coordination and strategies. *Journal of Neuroscience* 27, 27 (2007), 7297–7309. [Cited on page 93]

[180] Meinard Müller. 2007. *Information retrieval for music and motion*. Vol. 2. Springer. [Cited on page 96]

[181] Kouichi Murakami and Hitomi Taguchi. 1991. Gesture recognition using recurrent neural networks. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 237–242. [Cited on pages 25 and 106]

[182] Brad A Myers, Rishi Bhatnagar, Jeffrey Nichols, Choon Hong Peck, Dave Kong, Robert Miller, and A Chris Long. 2002. Interacting at a distance: measuring the performance of laser pointers and other devices. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 33–40. [Cited on page 14]

[183] Miguel A Nacenta, Yemliha Kamber, Yizhou Qiang, and Per Ola Kristensson. 2013. Memorability of pre-designed and user-defined gesture sets. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1099–1108. [Cited on page 12]

[184] Sebastiaan FW Neggers and Harold Bekkering. 2001. Gaze anchoring to a pointing target is present during the entire pointing movement and is driven by a non-visual signal. *Journal of Neurophysiology* 86, 2 (2001), 961–970. [Cited on page 17]

[185] Matei Negulescu, Jaime Ruiz, and Edward Lank. 2012. A recognition safety net: bi-level threshold recognition for mobile motion gestures. In *Proceedings of the 14th international conference on Human-computer interaction with mobile devices and services*. ACM, 147–150. [Cited on page 99]

[186] Jakob Nielsen. 1995. 10 usability heuristics for user interface design. *Nielsen Norman Group* 1, 1 (1995). [Cited on page 117]

[187] Jakob Nielsen. 2005. Ten usability heuristics. (2005). [Cited on page 12]

[188] Donald A. Norman. 2010. Natural User Interfaces Are Not Natural. *Interactions* 17, 3 (May 2010), 6–10. DOI:http://dx.doi.org/10.1145/1744161.1744163 [Cited on page 2]

[189] Donald A. Norman and Jakob Nielsen. 2010. Gestural Interfaces: A Step Backward in Usability. *interactions* 17, 5 (Sept. 2010), 46–49. DOI:http://dx.doi.org/10.1145/1836216.1836228 [Cited on page 11]

[190] RG O'Hagan, Alexander Zelinsky, and Sebastien Rougeaux. 2002. Visual gesture interfaces for virtual environments. *Interacting with Computers* 14, 3 (2002), 231–250. [Cited on page 14]

[191] Kenton O'Hara, Gerardo Gonzalez, Abigail Sellen, Graeme Penney, Andreas Varnavas, Helena Mentis, Antonio Criminisi, Robert Corish, Mark Rouncefield, Neville Dastur, and Tom Carrell. 2014. Touchless Interaction in Surgery. *Commun. ACM* 57, 1 (Jan. 2014), 70–77. DOI:http://dx.doi.org/10.1145/2541883.2541899 [Cited on pages 11, 14, 27, and 66]

[192] Alex Olwal and Andrew D Wilson. 2008. SurfaceFusion: unobtrusive tracking of everyday objects in tangible user interfaces. In *Proceedings of graphics interface 2008*. Canadian Information Processing Society, 235–242. [Cited on page 24]

[193] Michael Ortega and Laurence Nigay. 2009. AirMouse: Finger gesture for 2D and 3D interaction. In *IFIP Conference on Human-Computer Interaction*. Springer, 214–227. [Cited on page 14]

[194] Sharon Oviatt, Antonella DeAngeli, and Karen Kuhn. 1997. Integration and synchronization of input modes during multimodal human-computer interaction. In *Referring Phenomena in a Multimedia Context and their Computational Treatment*. Association for Computational Linguistics, 1–13. [Cited on page 118]

[195] Oyebade K Oyedotun and Adnan Khashman. 2017. Deep learning in vision-based static hand gesture recognition. *Neural Computing and Applications* 28, 12 (2017), 3941–3951. [Cited on pages 25 and 106]

[196] Shwetak N. Patel, Jeffrey S. Pierce, and Gregory D. Abowd. 2004. A Gesture-based Authentication Scheme for Untrusted Public Terminals. In *Proc. of ACM Symp. on User Interf. Softw. & Techn. (UIST '04)*. 157–160. DOI:http://dx.doi.org/10.1145/1029632.1029658 [Cited on page 22]

[197] C Lieke E Peper and Peter J Beek. 1998. Are frequency-induced transitions in rhythmic coordination mediated by a drop in amplitude? *Biological Cybernetics* 79, 4 (1998), 291–300. [Cited on page 17]

[198] Laura Ann Petitto. 1992. Modularity and constraints in early lexical acquisition: Evidence from children's early language and gesture. (1992). [Cited on page 9]

[199] Ken Pfeuffer, Melodie Vidal, Jayson Turner, Andreas Bulling, and Hans Gellersen. 2013. Pursuit calibration: Making gaze calibration less tedious and more flexible. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*. ACM, 261–270. [Cited on pages 3, 21, and 113]

[200] Alexandros Pino, Evangelos Tzemis, Nikolaos Ioannou, and Georgios Kouroupetroglou. 2013. Using kinect for 2D and 3D pointing tasks: performance evaluation. In *International Conference on Human-Computer Interaction*. Springer, 358–367. [Cited on pages 58 and 75]

[201] Thammathip Piumsomboon, Adrian Clark, Mark Billinghurst, and Andy Cockburn. 2013. User-defined gestures for augmented reality. In *IFIP Conference on Human-Computer Interaction*. Springer, 282–299. [Cited on page 27]

[202] Katrin Plaumann, Jan Ehlers, Florian Geiselhart, Gabriel Yuras, Anke Huckauf, and Enrico Rukzio. 2015. Better Than You Think: Head Gestures for Mid Air Input. In *IFIP Conference on Human-Computer Interaction*. Springer, 526–533. [Cited on page 15]

[203] Lorenzo Porzi, Stefano Messelodi, Carla Mara Modena, and Elisa Ricci. 2013. A smart watch-based gesture recognition system for assisting people with visual impairments. In *Proceedings of the 3rd ACM international workshop on Interactive multimedia on mobile & portable devices*. ACM, 19–24. [Cited on page 23]

[204] William T Powers and William T Powers. 1973. *Behavior: The control of perception*. Aldine Chicago. [Cited on page 18]

[205] C Prablanc, JF Echallier, E Komilis, and M Jeannerod. 1979. Optimal response of eye and hand motor systems in pointing at a visual target. *Biological cybernetics* 35, 2 (1979), 113–124. [Cited on page 17]

[206] Qifan Pu, Sidhant Gupta, Shyamnath Gollakota, and Shwetak Patel. 2013. Whole-home gesture recognition using wireless signals. In *Proceedings of the 19th annual international conference on Mobile computing & networking*. ACM, 27–38. [Cited on page 25]

[207] Francis Quek, David McNeill, Robert Bryll, Susan Duncan, Xin-Feng Ma, Cemil Kirbas, Karl E McCullough, and Rashid Ansari. 2002. Multimodal human discourse: gesture and speech. *ACM Transactions on Computer-Human Interaction (TOCHI)* 9, 3 (2002), 171–193. [Cited on pages 9 and 14]

[208] Argenis Ramirez Gomez and Hans-Werner Georg Gellersen. 2017. Behavioral Analysis of Smooth Pursuit Eye Movements for Interaction. In *COGAIN Symposium*. [Cited on page 82]

[209] Chotirat Ann Ratanamahatana and Eamonn Keogh. 2005. Three myths about dynamic time warping data mining. In *Proceedings of the 2005 SIAM International Conference on Data Mining*. SIAM, 506–510. [Cited on page 96]

[210] Siddharth S Rautaray and Anupam Agrawal. 2015. Vision based hand gesture recognition for human computer interaction: a survey. *Artificial intelligence review* 43, 1 (2015), 1–54. [Cited on page 25]

[211] Keith Rayner. 1998. Eye movements in reading and information processing: 20 years of research. *Psychological bulletin* 124, 3 (1998), 372. [Cited on page 19]

[212] Guinness World Records. 2011. Fastest selling gaming peripheral. Website. (3 January 2011). Retrieved February 15, 2019 from `https://www.guinnessworldrecords.com/world-records/fastest-selling-gaming-peripheral/`. [Cited on page 27]

[213] Bruno H Repp. 2005. Sensorimotor synchronization: a review of the tapping literature. *Psychonomic bulletin & review* 12, 6 (2005), 969–992. [Cited on page 17]

[214] Bruno H Repp. 2006. Rate limits of sensorimotor synchronization. *Advances in cognitive psychology* 2, 2-3 (2006), 163–181. [Cited on page 20]

[215] Bruno H Repp and Amandine Penel. 2004. Rhythmic movement is attracted more strongly to auditory than to visual rhythms. *Psychological research* 68, 4 (2004), 252–270. [Cited on pages 17 and 113]

[216] Gabriel Reyes, Jason Wu, Nikita Juneja, Maxim Goldshtein, W Keith Edwards, Gregory D Abowd, and Thad Starner. 2018. SynchroWatch: One-handed synchronous smartwatch gestures using correlation and magnetic sensing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 4 (2018), 158. [Cited on page 21]

[217] Miguel Reyes, Gabriel Dominguez, and Sergio Escalera. 2011. Featureweighting in dynamic timewarping for gesture recognition in depth data. In *2011 IEEE international conference on computer vision workshops (ICCV Workshops)*. IEEE, 1182–1188. [Cited on page 95]

[218] Andreas Riener. 2012. Gestural interaction in vehicular applications. *Computer* 45, 4 (2012), 42–47. [Cited on page 11]

[219] Giacomo Rizzolatti, Luciano Fadiga, Léonardo Fogassi, and Vittorio Gallese. 1999. Resonance behaviors and mirror neurons. *Archives italiennes de biologie* 137, 2 (1999), 85–100. [Cited on page 18]

[220] Christof Roduner, Marc Langheinrich, Christian Floerkemeier, and Beat Schwarzentrub. 2007. Operating Appliances with Mobile Phones—Strengths and Limits of a Universal Interaction Device. In *Pervasive Computing (Lecture Notes in Computer Science)*, Anthony LaMarca, Marc Langheinrich, and Khai N. Truong (Eds.), Vol. 4480. Springer Berlin Heidelberg, 198–215. `DOI: http://dx.doi.org/10.1007/978-3-540-72037-9_12` [Cited on page 26]

[221] Melvyn Roerdink, Ellen D Ophoff, C Lieke E Peper, and Peter J Beek. 2008. Visual and musculoskeletal underpinnings of anchoring in rhythmic visuo-motor tracking. *Experimental Brain Research* 184, 2 (2008), 143–156. [Cited on page 17]

[222] M Roerdink, CE Peper, and PJ Beek. 2005. Effects of correct and transformed visual feedback on rhythmic visuo-motor tracking: Tracking performance and visual search behavior. *Human movement science* 24, 3 (2005), 379–402. [Cited on pages 17 and 78]

[223] Mahsan Rofouei, Andrew Wilson, AJ Brush, and Stewart Tansley. 2012. Your phone or mine?: fusing body, touch and device sensing for multi-user device-display interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1915–1918. [Cited on page 22]

[224] Edward Rosten and Tom Drummond. 2006. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*, Vol. 1. 430–443. DOI:http://dx.doi.org/10.1007/11744023_34 [Cited on pages 32 and 61]

[225] Gustavo Rovelo, Donald Degraen, Davy Vanacken, Kris Luyten, and Karin Coninx. 2015. Gestu-Wan-an intelligible mid-air gesture guidance system for walk-up-and-use displays. In *Human-Computer Interaction*. Springer, 368–386. [Cited on page 13]

[226] Hiroaki Sakoe and Seibi Chiba. 1978. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing* 26, 1 (1978), 43–49. [Cited on page 99]

[227] Lawrence Sambrooks and Brett Wilkinson. 2013. Comparison of Gestural, Touch, and Mouse Interaction with Fitts' Law. In *Proceedings of the 25th Australian Computer-Human Interaction Conference: Augmentation, Application, Innovation, Collaboration (OzCHI '13)*. ACM, New York, NY, USA, 119–122. DOI:http://dx.doi.org/10.1145/2541016.2541066 [Cited on pages 71 and 75]

[228] T Scott Saponas, Desney S Tan, Dan Morris, and Ravin Balakrishnan. 2008. Demonstrating the feasibility of using forearm electromyography for muscle-computer interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 515–524. [Cited on page 23]

[229] Thomas Schlömer, Benjamin Poppinga, Niels Henze, and Susanne Boll. 2008. Gesture recognition with a Wii controller. In *Proceedings of the 2nd international conference on Tangible and embedded interaction*. ACM, 11–14. [Cited on page 23]

[230] Dominik Schmidt, Fadi Chehimi, Enrico Rukzio, and Hans Gellersen. 2010. PhoneTouch: A Technique for Direct Phone Interaction on Surfaces. In *Proceedings of the 23Nd Annual ACM Sym-*

posium on User Interface Software and Technology (UIST '10). ACM, New York, NY, USA, 13–16. DOI:http://dx.doi.org/10.1145/1866029.1866034 [Cited on page 22]

[231] Dominik Schmidt, David Molyneaux, and Xiang Cao. 2012a. PICOntrol: Using a Handheld Projector for Direct Control of Physical Devices Through Visible Light. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology (UIST '12)*. ACM, New York, NY, USA, 379–388. DOI:http://dx.doi.org/10.1145/2380116.2380166 [Cited on page 26]

[232] Dominik Schmidt, Julian Seifert, Enrico Rukzio, and Hans Gellersen. 2012b. A Cross-device Interaction Style for Mobiles and Surfaces. In *Proceedings of the Designing Interactive Systems Conference (DIS '12)*. ACM, New York, NY, USA, 318–327. DOI:http://dx.doi.org/10.1145/2317956.2318005 [Cited on page 22]

[233] Richard A Schmidt. 1991. Frequent augmented feedback can degrade learning: Evidence and interpretations. In *Tutorials in motor neuroscience*. Springer, 59–75. [Cited on page 13]

[234] Maureen Schultz, Janet Gill, Sabiha Zubairi, Ruth Huber, and Fred Gordin. 2003. Bacterial contamination of computer keyboards in a teaching hospital. *Infection Control & Hospital Epidemiology* 24, 4 (2003), 302–303. [Cited on page 27]

[235] Matthias Schwaller and Denis Lalanne. 2013. Pointing in the air: measuring the effect of hand selection strategies on performance and effort. In *Human Factors in Computing and Informatics*. Springer, 732–747. [Cited on page 75]

[236] Julia Schwarz, Charles Claudius Marais, Tommer Leyvand, Scott E Hudson, and Jennifer Mankoff. 2014. Combining body pose, gaze, and gesture to determine intention to interact in vision-based interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 3443–3452. [Cited on page 11]

[237] Jakub Segen and Senthil Kumar. 1998. Gesture VR: vision-based 3 D hand interace for spatial interaction. In *ACM Multimedia*. 455–464. [Cited on page 14]

[238] Joan Serra and Josep Ll Arcos. 2014. An empirical evaluation of similarity measures for time series classification. *Knowledge-Based Systems* 67 (2014), 305–314. [Cited on page 77]

[239] Gözel Shakeri, John H Williamson, and Stephen Brewster. 2017. Novel Multimodal Feedback Techniques for In-Car Mid-Air Gesture Interaction. In *Proceedings of the 9th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*. ACM, 84–93. [Cited on page 27]

[240] Osamu Shigeta, Shingo Kagami, and Koichi Hashimoto. 2008. Identifying a moving object with an accelerometer in a camera view. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 3872–3877. [Cited on page 106]

[241] B Shneiderman. 1987. Direct manipulation: A step beyond programming languages (excerpt). *Readings in Human-Computer Interaction: A Multidisciplinary Approach* (1987), 461–467. [Cited on page 12]

[242] Garth Shoemaker, Anthony Tang, and Kellogg S. Booth. 2007. Shadow Reaching: A New Perspective on Interaction for Large Displays. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology (UIST '07)*. ACM, New York, NY, USA, 53–56. DOI: http://dx.doi.org/10.1145/1294211.1294221 [Cited on page 14]

[243] Jamie Shotton, Andrew W Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, and Andrew Blake. 2011. Real-time human pose recognition in parts from single depth images.. In *Cvpr*, Vol. 2. 3. [Cited on page 24]

[244] Linda E Sibert and Robert JK Jacob. 2000. Evaluation of eye gaze interaction. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. ACM, 281–288. [Cited on page 8]

[245] Tomas Simon, Hanbyul Joo, Iain Matthews, and Yaser Sheikh. 2017. Hand Keypoint Detection in Single Images using Multiview Bootstrapping. In *CVPR*. [Cited on pages 24, 80, and 81]

[246] Richard C Simpson and Heidi H Koester. 1999. Adaptive one-switch row-column scanning. *IEEE Transactions on Rehabilitation Engineering* 7, 4 (1999), 464–473. [Cited on page 16]

[247] Rajinder Sodhi, Hrvoje Benko, and Andrew Wilson. 2012. LightGuide: projected visualizations for hand movement guidance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 179–188. [Cited on page 13]

[248] Peng Song, Wooi Boon Goh, William Hutama, Chi-Wing Fu, and Xiaopei Liu. 2012. A handle bar metaphor for virtual object manipulation with mid-air interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1297–1306. [Cited on page 14]

[249] R. William Soukoreff and I. Scott MacKenzie. 2004. Towards a Standard for Pointing Device Evaluation, Perspectives on 27 Years of Fitts' Law Research in HCI. *Int. J. Hum.-Comput. Stud.* 61, 6 (Dec. 2004), 751–789. DOI:http://dx.doi.org/10.1016/j.ijhcs.2004.09.001 [Cited on pages 71, 72, and 75]

[250] Nigel Stepp and Michael T Turvey. 2017. Anticipation in manual tracking with multiple delays. *Journal of Experimental Psychology: Human Perception and Performance* 43, 5 (2017), 914. [Cited on page 78]

[251] Helman I Stern, Juan P Wachs, and Yael Edan. 2008. Optimal consensus intuitive hand gesture vocabulary design. In *2008 IEEE International Conference on Semantic Computing*. IEEE, 96–103. [Cited on page 12]

[252] Rainer Stiefelhagen and Jie Zhu. 2002. Head orientation and gaze direction in meetings. In *CHI'02 Extended Abstracts on Human Factors in Computing Systems*. ACM, 858–859. [Cited on page 58]

[253] Matt Strickland, Jamie Tremaine, Greg Brigley, and Calvin Law. 2013. Using a depth-sensing infrared camera system to access and manipulate medical imaging from within the sterile operating field. *Canadian Journal of Surgery* 56, 3 (2013), E1. [Cited on page 27]

[254] Breanna E Studenka and Howard N Zelaznik. 2011. Synchronization in repetitive smooth movement requires perceptible events. *Acta Psychologica* 136, 3 (2011), 432–441. [Cited on pages 92 and 114]

[255] Breanna E Studenka, Howard N Zelaznik, and Ramesh Balasubramaniam. 2012. The distinction between tapping and circle drawing with and without tactile feedback: an examination of the sources of timing variance. *The Quarterly Journal of Experimental Psychology* 65, 6 (2012), 1086–1100. [Cited on page 78]

[256] David J Sturman and David Zeltzer. 1994. A survey of glove-based input. *IEEE Computer graphics and Applications* 14, 1 (1994), 30–39. [Cited on page 23]

[257] Mark JM Sullman, Francesc Prat, and Duygu Kuzu Tasci. 2015. A roadside study of observable driver distractions. *Traffic injury prevention* 16, 6 (2015), 552–557. [Cited on page 27]

[258] Ke Sun, Yuntao Wang, Chun Yu, Yukang Yan, Hongyi Wen, and Yuanchun Shi. 2017. Float: one-handed and touch-free target selection on smartwatches. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 692–704. [Cited on page 23]

[259] Ivan E Sutherland. 1964. Sketchpad a man-machine graphical communication system. *Simulation* 2, 5 (1964), R–3. [Cited on page 8]

[260] Justin H Tan, Cherng Chao, Mazen Zawaideh, Anne C Roberts, and Thomas B Kinney. 2013. Informatics in radiology: Developing a touchless user interface for intraoperative image control during interventional radiology procedures. *Radiographics* 33, 2 (2013), E61–E70. [Cited on page 27]

[261] J-C Terrillon, Mahdad N Shirazi, Hideo Fukamachi, and Shigeru Akamatsu. 2000. Comparative performance of different skin chrominance models and chrominance spaces for the automatic detection of human faces in color images. In *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)*. IEEE, 54–61. [Cited on page 24]

[262] Chuck Thacker, Ed McCreight, Butler Lampson, Robert Sproull, and David Boggs. 1979. Alto: A personal computer. `https://www.microsoft.com/en-us/research/publication/alto-personal-computer/` [Cited on page 8]

[263] Edward Tse, Saul Greenberg, Chia Shen, Clifton Forlines, and Ryo Kodama. 2008. Exploring true multi-user multimodal interaction over a digital table. In *Proceedings of the 7th ACM conference on Designing interactive systems*. ACM, 109–118. [Cited on page 8]

[264] Robrecht PRD van der Wel, Dagmar Sternad, and David A Rosenbaum. 2009. Moving the arm at different rates: slow movements are avoided. *Journal of motor behavior* 42, 1 (2009), 29–36. [Cited on pages 20 and 92]

[265] Radu-Daniel Vatavu. 2012a. Point & Click Mediated Interactions for Large Home Entertainment Displays. *Multimedia Tools Appl.* 59, 1 (July 2012), 113–128. DOI:http://dx.doi.org/10.1007/s11042-010-0698-5 [Cited on page 14]

[266] Radu-Daniel Vatavu. 2012b. User-defined Gestures for Free-hand TV Control. In *Proceedings of the 10th European Conference on Interactive Tv and Video (EuroiTV '12)*. ACM, New York, NY, USA, 45–48. DOI:http://dx.doi.org/10.1145/2325616.2325626 [Cited on pages 12 and 26]

[267] Eduardo Velloso, Jason Alexander, Andreas Bulling, and Hans Gellersen. 2015. *Interactions Under the Desk: A Characterisation of Foot Movements for Input in a Seated Position*. Springer International Publishing, Cham, 384–401. DOI:http://dx.doi.org/10.1007/978-3-319-22701-6_29 [Cited on page 58]

[268] Eduardo Velloso, Marcus Carter, Joshua Newn, Augusto Esteves, Christopher Clarke, and Hans Gellersen. 2017. Motion Correlation: Selecting Objects by Matching Their Movement. *ACM Trans. Comput.-Hum. Interact.* 24, 3, Article 22 (April 2017), 35 pages. DOI:http://dx.doi.org/10.1145/3064937 [Cited on pages 2 and 15]

[269] Eduardo Velloso, Flavio Luiz Coutinho, Andrew Kurauchi, and Carlos H Morimoto. 2018. Circular orbits detection for gaze interaction using 2D correlation and profile matching algorithms. In *Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications*. ACM, 25. [Cited on pages 93, 95, and 99]

[270] Eduardo Velloso, Dominik Schmidt, Jason Alexander, Hans Gellersen, and Andreas Bulling. 2015. The Feet in Human–Computer Interaction: A Survey of Foot-Based Interaction. *ACM Comput. Surv.* 48, 2, Article 21 (Sept. 2015), 35 pages. DOI:http://dx.doi.org/10.1145/2816455 [Cited on page 14]

[271] Eduardo Velloso, Markus Wirth, Christian Weichel, Augusto Esteves, and Hans Gellersen. 2016. AmbiGaze: Direct Control of Ambient Devices by Gaze. In *Proceedings of the Designing Interactive Systems Conference (DIS '16)*. ACM, New York, NY, USA, 4. DOI:http://dx.doi.org/10.1145/2901790.2901867 [Cited on pages 19, 28, 30, and 113]

[272] David Verweij, Augusto Esteves, Vassilis-Javed Khan, and Saskia Bakker. 2017. WaveTrace: motion matching input using wrist-worn motion sensors. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. ACM, 2180–2186. [Cited on page 21]

[273] Mélodie Vidal, Andreas Bulling, and Hans Gellersen. 2013. Pursuits: Spontaneous Interaction with Displays Based on Smooth Pursuit Eye Movement and Moving Targets. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '13)*. ACM, New York, NY, USA, 439–448. DOI:http://dx.doi.org/10.1145/2493432.2493477 [Cited on pages 3, 19, 20, 26, 28, 29, 34, 93, 94, 108, and 113]

[274] Nicolas Villar and Hans Gellersen. 2007. A Malleable Control Structure for Softwired User Interfaces. In *Proceedings of the 1st International Conference on Tangible and Embedded Interaction (TEI '07)*. ACM, New York, NY, USA, 49–56. DOI:http://dx.doi.org/10.1145/1226969.1226980 [Cited on page 14]

[275] Michail Vlachos, George Kollios, and Dimitrios Gunopulos. 2002. Discovering similar multidimensional trajectories. In *Proceedings 18th international conference on data engineering*. IEEE, 673–684. [Cited on pages 96, 99, and 103]

[276] Daniel Vogel and Ravin Balakrishnan. 2005. Distant Freehand Pointing and Clicking on Very Large, High Resolution Displays. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology (UIST '05)*. ACM, New York, NY, USA, 33–42. DOI:http://dx.doi.org/10.1145/1095034.1095041 [Cited on pages 14, 15, and 27]

[277] Juan Wachs, Helman Stern, Yael Edan, Michael Gillam, Craig Feied, Mark Smithd, and Jon Handler. 2008. Real-time hand gesture interface for browsing medical images. *International Journal of Intelligent Computing in Medical Sciences & Image Processing* 2, 1 (2008), 15–25. [Cited on page 27]

[278] Edwin Walsh, Walter Daems, and Jan Steckel. 2015. An optical head-pose tracking sensor for pointing devices using IR-LED based markers and a low-cost camera. In *SENSORS, 2015 IEEE*. IEEE, 1–4. [Cited on page 75]

[279] Robert Walter, Gilles Bailly, and Jörg Müller. 2013. StrikeAPose: revealing mid-air gestures on public displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 841–850. [Cited on pages 11, 13, 27, and 112]

[280] Robert Walter, Gilles Bailly, Nina Valkanova, and Jörg Müller. 2014. Cuenesics: using mid-air gestures to select items on interactive public displays. In *Proceedings of the 16th international conference on Human-computer interaction with mobile devices & services*. ACM, 299–308. [Cited on page 15]

[281] Heng Wang, Alexander Kläser, Cordelia Schmid, and Liu Cheng-Lin. 2011. Action recognition by dense trajectories. In *CVPR 2011-IEEE Conference on Computer Vision & Pattern Recognition*. IEEE, 3169–3176. [Cited on page 25]

[282] Xiaoyue Wang, Abdullah Mueen, Hui Ding, Goce Trajcevski, Peter Scheuermann, and Eamonn Keogh. 2013. Experimental comparison of representation methods and distance measures for time series data. *Data Mining and Knowledge Discovery* 26, 2 (2013), 275–309. [Cited on pages 4, 77, 93, and 96]

[283] Andrew M Webb, Michel Pahud, Ken Hinckley, and Bill Buxton. 2016. Wearables as context for guiard-abiding bimanual touch. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. ACM, 287–300. [Cited on page 22]

[284] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. 2016. Convolutional pose machines. In *CVPR*. [Cited on pages 24, 80, and 81]

[285] Hongyi Wen, Julian Ramos Rojas, and Anind K Dey. 2016. Serendipity: Finger gesture recognition using an off-the-shelf smartwatch. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 3847–3851. [Cited on page 23]

[286] Daniel Wigdor and Dennis Wixon. 2011. *Brave NUI World: Designing Natural User Interfaces for Touch and Gesture* (1st ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. [Cited on pages 2 and 9]

[287] John Williamson. 2006. *Continuous uncertain interaction*. Ph.D. Dissertation. University of Glasgow. [Cited on pages 16 and 28]

[288] John Williamson and Roderick Murray-Smith. 2002. Audio feedback for gesture recognition. (2002). [Cited on page 11]

[289] John Williamson and Roderick Murray-Smith. 2004. Pointing Without a Pointer. In *CHI '04 Extended Abstracts on Human Factors in Computing Systems (CHI EA '04)*. ACM, New York, NY, USA, 1407–1410. DOI:http://dx.doi.org/10.1145/985921.986076 [Cited on pages 2, 18, 26, 28, and 29]

[290] Andrew Wilson and Steven Shafer. 2003. XWand: UI for Intelligent Spaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*. ACM, New York, NY, USA, 545–552. DOI:http://dx.doi.org/10.1145/642611.642706 [Cited on page 26]

[291] Andrew D Wilson and Hrvoje Benko. 2014. Crossmotion: fusing device and image motion for user identification, tracking and device association. In *Proceedings of the 16th International Conference on Multimodal Interaction*. ACM, 216–223. [Cited on page 22]

[292] Raymond H Wimmers, Peter J Beek, and Piet CW van Wieringen. 1992. Phase transitions in rhythmic tracking movements: A case of unilateral coupling. *Human Movement Science* 11, 1-2 (1992), 217–226. [Cited on page 17]

[293] Jacob O Wobbrock, Htet Htet Aung, Brandon Rothrock, and Brad A Myers. 2005. Maximizing the guessability of symbolic input. In *CHI'05 extended abstracts on Human Factors in Computing Systems*. ACM, 1869–1872. [Cited on page 12]

[294] Jacob O Wobbrock, Meredith Ringel Morris, and Andrew D Wilson. 2009. User-defined gestures for surface computing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1083–1092. [Cited on page 12]

[295] Jacob O. Wobbrock, Kristen Shinohara, and Alex Jansen. 2011. The Effects of Task Dimensionality, Endpoint Deviation, Throughput Calculation, and Experiment Design on Pointing Measures and Models. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 1639–1648. DOI:http://dx.doi.org/10.1145/1978942.1979181 [Cited on page 71]

[296] Hans Peter Wyss, Roland Blach, and Matthias Bues. 2006. iSith-Intersection-based spatial interaction for two hands. In *3D User Interfaces (3DUI'06)*. IEEE, 59–61. [Cited on page 15]

[297] Hee-Deok Yang, A-Yeon Park, and Seong-Whan Lee. 2007. Gesture spotting and recognition for human–robot interaction. *IEEE Transactions on robotics* 23, 2 (2007), 256–270. [Cited on page 11]

[298] Alper Yilmaz, Omar Javed, and Mubarak Shah. 2006. Object tracking: A survey. *Acm computing surveys (CSUR)* 38, 4 (2006), 13. [Cited on page 25]

[299] Howard N Zelaznik, Rebecca Spencer, and Richard B Ivry. 2002. Dissociation of explicit and implicit timing in repetitive tapping and drawing movements. *Journal of Experimental Psychology: Human Perception and Performance* 28, 3 (2002), 575. [Cited on page 17]

[300] Robert C Zeleznik, Andrew S Forsberg, and Jürgen P Schulze. 2005. Look-that-there: Exploiting gaze in virtual reality interactions. *Technical report, Technical Report CS-05* (2005). [Cited on page 23]

[301] Cheng Zhang, Anandghan Waghmare, Pranav Kundra, Yiming Pu, Scott Gilliland, Thomas Ploetz, Thad E Starner, Omer T Inan, and Gregory D Abowd. 2017. FingerSound: Recognizing unistroke thumb gestures using a ring. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 3 (2017), 120. [Cited on page 21]

[302] Tengxiang Zhang, Xin Yi, Ruolin Wang, Yuntao Wang, Chun Yu, Yiqin Lu, and Yuanchun Shi. 2018. Tap-to-Pair: Associating Wireless Devices with Synchronous Tapping. *Proceedings of the*

ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 2, 4 (2018), 201. [Cited on page 22]

[303] Jingbo Zhao and Robert S Allison. 2017. Real-time head gesture recognition on head-mounted displays using cascaded hidden Markov models. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2361–2366. [Cited on page 27]

[304] Yu Zheng. 2015. Trajectory data mining: an overview. *ACM Transactions on Intelligent Systems and Technology (TIST)* 6, 3 (2015), 29. [Cited on page 77]

[305] Yunyue Zhu and Dennis Shasha. 2003. Warping indexes with envelope transforms for query by humming. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*. ACM, 181–192. [Cited on page 95]

[306] Thomas G Zimmerman, Jaron Lanier, Chuck Blanchard, Steve Bryson, and Young Harvill. 1987. A hand gesture interface device. In *ACM SIGCHI Bulletin*, Vol. 18. ACM, 189–192. [Cited on page 23]

[307] Gottfried Zimmermann, Gregg Vanderheiden, and Al Gilman. 2002. Prototype Implementations for a Universal Remote Console Specification. In *CHI '02 Extended Abstracts on Human Factors in Computing Systems (CHI EA '02)*. ACM, New York, NY, USA, 510–511. DOI:http://dx.doi.org/10.1145/506443.506454 [Cited on page 26]