Motion Correlation: Selecting Objects by Matching their Movement

EDUARDO VELLOSO, The University of Melbourne
MARCUS CARTER, The University of Melbourne
JOSHUA NEWN, The University of Melbourne
AUGUSTO ESTEVES, Edinburgh Napier University and Lancaster University
CHRISTOPHER CLARKE, Lancaster University
HANS GELLERSEN, Lancaster University

Selection is a canonical task in user interfaces, commonly supported by presenting objects for acquisition by pointing. In this article, we consider *motion correlation* as an alternative for selection. The principle is to represent available objects by motion in the interface, have users identify a target by mimicking its specific motion, and use the correlation between the system's output with the user's input to determine the selection. The resulting interaction has compelling properties, as users are guided by motion feedback, and only need to copy a presented motion. Motion correlation has been explored in earlier work but only recently begun to feature in holistic interface designs. We provide a first comprehensive review of the principle, and present an analysis of five previously published works, in which motion correlation underpinned the design of novel gaze and gesture interfaces for diverse application contexts. We derive guidelines for motion correlation algorithms, motion feedback, choice of modalities, overall design of motion correlation interfaces, and identify opportunities and challenges identified for future research and design.

Additional Key Words and Phrases: Motion correlation, Interaction techniques, Gaze interaction, Gesture interfaces, Eye tracking, Motion tracking, Natural user interfaces

ACM Reference format:
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), 34 pages.
https://doi.org/10.1145/3064937

1 INTRODUCTION

Object Selection is a canonical task in user interfaces [16]. Commonly, a set of objects is presented to the user in a static arrangement, such as icons on a screen. Selection then requires a targeting motion by the user to indicate the object and to confirm its selection, for instance pointing and clicking with the mouse. In this article, we consider the design of interfaces that employ motion very differently for selection: presenting available options each with a distinct motion that users can simply copy, for example by producing a corresponding hand, eye, or body movement. We call

This work is supported by the Victorian State Government and Microsoft through their contributions to the Microsoft Research Centre for Social Natural User Interfaces (SocialNUI), and by Google through a Faculty Research Award.
Author's addresses: E. Velloso, M. Carter, J. Newn, School of Computing and Information Systems, The University of Melbourne, emails: {eduardo.velloso, marcus.carter, joshua.newn}@unimelb.edu.au; A. Esteves, Institute for Informatics and Digital Innovation, Edinburgh Napier University, email: a.esteves@napier.ac.uk; C. Clarke, H. Gellersen, School of Computing and Communications, Lancaster University, emails: {c.clarke1, h.gellersen}@lancaster.ac.uk.

Manuscript submitted to ACM

the principle behind such interfaces *motion correlation*—if each available object displays a different motion, following an object's trajectory disambiguates the selection.

Motion correlation draws on natural human abilities. When a bird flying past draws our attention, our eyes lock onto it and perform a movement that corresponds to the bird's, and if we pointed the bird in flight out to another observer, our hands would describe a similar motion too. When a personal trainer demonstrates an exercise, we are able to copy and reproduce their movements. These abilities have become reflected in user interfaces that display motion as a prompt for users to copy and match for a variety of purposes. Examples include mimicry as a game mechanic [17], prompting of gesture sequences for authentication [48], and interactive movement guidance in sports and therapy [40, 59].

Interactive objects such as icons and buttons are usually at the same position on the screen, and when objects do move, the movement is often user-initiated (e.g. dragging a window) or not selectable (e.g. video playback). However, objects can be put in motion to distinguish them by their movement [68], or to augment them with moving parts [14]. Motion and animation of objects can also be inherent elements of an interface, for example in video games, molecular simulations, air traffic control systems, interactive video, surveillance systems, and video annotation systems [21, 24, 25, 30].

Williamson and Murray-Smith were the first to describe excitation of displayed objects as a strategy for selection by making corresponding motions with an input device [68]. Their work inspired Fekete et al. to explore motion correlation as selection technique in conventional graphical user interfaces [14]. These works represent milestones in establishing the motion correlation principle. However, it is only in recent work that this concept has been applied as part of holistic interface designs. In our combined work, we have used motion correlation with gaze and gesture as modalities, in interfaces designed for public displays, smart watches, interactive TVs, and smart homes [6, 8, 13, 61, 63].

Our aim in this article is to promote motion correlation as a fundamental principle for interaction. We advance theoretical and practical understanding of the principle through the following contributions:

- Analysis of the background and context of motion correlation: how the principle is defined among other selection mechanisms, how human abilities are leveraged for motion correlation, insights from pioneering works on the topic, and the wider context of related work on motion matching.
- Review of five projects in which we have designed, built, and studied input techniques based on motion correlation: *Pursuits*, *Orbits*, *AmbiGaze*, *PathSync*, and *Tracematch*; in each case, analysing the rationale for using motion correlation, the specific ways in which the principle was applied, and the insights gained.
- Synthesis of design knowledge built across our prior work into general guidelines for motion correlation interface design, including algorithm design, the design of motion feedback, the choice of input and output modalities, and the broader choices of putting everything together in cohesive interfaces.
- Identification and discussion of opportunities and challenges toward future work on motion correlation interfaces.

2 MOTION CORRELATION: DEFINITION, BACKGROUND AND RELATED WORK

The interaction principle we discuss in this article is defined by three properties:

(1) Objects available for interaction are represented by a motion displayed to the user;
(2) Users express their intent to interact with an object through movement at the interface that corresponds with the object's motion;
(3) Correlation of the system's output and the user's input determines the selection of objects for interaction.

Earlier work has described this interaction principle in diverse ways, for instance as *feedback-based active selection* [67], *kinematic coincidence* [14], and *rhythmic path mimicry* [6]. In this article we use the term *motion correlation* as shorthand

for the principle, proposed with a view of consolidating terminology. In doing so, we interpret both "motion" and "correlation" in the widest sense. Motion may be of any form and shape, with no limitation on the media, modalities or devices used for input and output. Correlation denotes that there is a relationship and correspondence between the system- and user-generated motion but does not presume any specific definition or measure of similarity.

2.1 Motion Correlation as a Selection Strategy

Selection is a fundamental task when interacting with user interfaces. Fekete et al. present an interesting way of looking at the selection problem, by analysing the basic mechanism through which the system can tell which target is being selected [14]. In all cases, this boils down to how inputs are matched to outputs. Whenever we select, it involves a one-to-one match between the current output presented and the input provided. In conventional pointing, this match is *spatial*—when the coordinates provided by the input device coincide with the location of an object, the system understands that the user wants to select that object.

Movements of the pointing device are echoed on the screen by movements of the pointer (or cursor) and other visual changes. Pointing techniques map the input coordinate space to the coordinate space of the screen. When these two coordinate spaces coincide, the technique is referred to as *direct* [28]. Examples include touch screens, stylus-sensing surfaces, and isomorphic virtual reality environments. When these spaces are separate, the techniques are *indirect* (e.g. mice, trackballs, joysticks) [28].

The second type of matching identified by Fekete et al. is *semantic*. In this case, what connects input actions to actions on the interface is the meaning behind them. Examples include typing a written command on a console, pressing a key on the keyboard, or clapping to turn the lights off.

In contrast to these strategies, motion correlation determines the target that users want to select by matching the target's motion with the user's input. Applied in a conventional setup of screen and pointing device, the *relative* movement of the device would be matched to the *relative* movement of the targets on the screen. In this case, the matching is neither spatial (it does not require any absolute mapping between the input and output coordinate systems) nor semantic (the same movement shape can encode different commands depending on the target it is synchronised with).

As we will detail below, we have used motion correlation with gaze and gestures as input modalities. The gaze interface literature has previously largely focused on spatial matching of gaze direction for selection of what users look at when they dwell on a target [31] but also considered gaze gestures where eye movement patterns are semantically matched [11]. The two matching principles are also prevalent across gesture interfaces, where gestures are either matched for pointer control or interpreted as commands.

2.2 Temporal, Spatial and Spatiotemporal Motion Correlation

Motion is a change in position of an object over time. When considering how well one motion is matched by another, one can consider the spatial aspects (path, shape), temporal aspects (dynamics, rhythm) or a combination of the two. This lets us define a taxonomy for motion correlation interfaces with three principal categories – spatial, temporal, and spatiotemporal. The work we review in this article falls into the third category as the interfaces we have designed involve motion in space that is followed or mimicked by users in real-time. The advantage of considering both time and space in the matching is that the same motion shape can be "reused" for multiple objects with temporal variation, and vice versa. However the other two categories are of interest too, and we briefly review them here.

In the spatial category, the correspondence of two motions is determined solely by spatial aspects, for example the similarity of two shapes while ignoring how the shapes evolved as time series. One can imagine interfaces that guide users in producing symbolic gestures where shape compliance is important while users might purposely be given flexibility in producing the shape. However, we consider this a less compelling variant of motion correlation, as we are interested in the closer coupling between input and output, where input becomes an act of synchronising with a displayed motion.

The temporal case, in contrast, is of fundamental interest as a user's input and a system's output have an inherent temporal dimension. Even a simple discrete input such as pressing a button involves motion, where the timing may be meaningful in relation to the system's changing output. Switch scanning is a standard accessibility technique that illustrates this form of motion correlation: the interface displays the available inputs on a grid with an indicator moving from item to item, and the user aligns their activation of a switch with the indicator's movement [55]. Rhythmic menus are a similar technique [43]: when a user opens a menu, the system highlights each item in succession at a constant rate, presenting a steady rhythm to which the user aligns their input action (a mouse click at the right time, as opposed to the right position). The motions involved in temporal matching can be of more complex structure, for example tapping the rhythm of a song [4] or reproducing rhythmic patterns displayed to the user [18].

2.3 Human Motion Perception and Motor Behaviour

Nature is abundant with all kinds of movement, and the evolution of the ability to track moving targets was crucial for the survival of our species. From detecting an approaching predator in the peripheral vision to tracking a running prey when hunting, movement perception is a fundamental ability that we develop from early childhood.

Research has shown that even one-week old infants are able to follow a moving target with the eyes [5]. This type of eye movement is known as *smooth pursuit*. These movements are characteristically smooth and closely match the movement of the target. This makes them substantially different from the other common types of eye movements, namely *fixations* and *saccades*. An important characteristic of this movement is the ability to select and track a single object, even in the presence of multiple stimuli [1]. In practice, when tracking a moving object, ocular pursuit movements involve a combination of smooth pursuits and catch-up saccades that realign the target in the fovea (the area of the retina with highest visual acuity). When following periodic movements—such as the ones explored in the remainder of the paper—the frequency of movement will dictate the distribution of movement types. Whereas at low frequencies ($< 0.4Hz$), the movement is almost entirely smooth, increasingly more saccades happen at higher frequencies [1].

Though our eyes are naturally drawn by the appearance of a moving target, maintaining the pursuit movement that tracks it requires attention. If the person is paying attention to a specific target, the presence of distractors has little influence on the pursuit, unless they are also being attended to and their retinal motion conflicts with that of the target being pursued [33]. Therefore, selective attention is crucial to be able to accurately follow a target whilst ignoring other stimuli [1].

Motion perception is related to the motor system beyond the control of our eye movements. There is evidence of mirror neurons and motor resonance behaviours, where the same neurons are activated when observing and when executing action [37, 50]. When infants develop the ability to reach for moving objects, they aim ahead of the object rather than at its current position, suggesting a fundamental capacity to coordinate behaviours with external events [66]. While it is more difficult for humans to intercept a target when it is moving [24], it has been observed that we naturally align our hand movement to the phase and frequency of moving stimuli [2, 23].

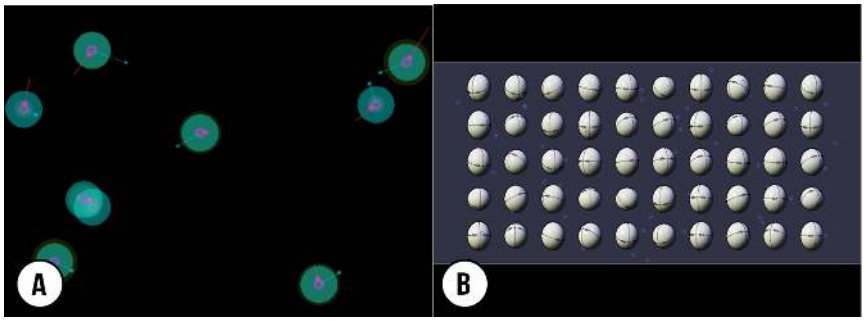

Fig. 1. Williamson's concept demonstrators for Active Selection: in the *Browninan motion* interface, objects are disturbed in their motion (A), and in the *Eggheads* interface in their orientation (B). The user's input applies to all objects simulatenously, and an object is selected by continuous action cancelling the effects of its disturbance (image source: [67, 68])

.

Humans are particularly good at rhythm perception, and corresponding periodic movement. Rhythmic processes are pervasive in nature, for instance our breathing and walking [19]. From early childhood we are used to tapping our feet and clapping our hands in synchrony with music. Human movements are naturally cyclic and harmonic [20], and our ability to move in synchrony with external motion has an analogy in coupled oscillators widely observed in biological systems [56]. Interestingly, coupling with external movement is also observed in social environments, where people subconsciously mimic the behaviour of others (known as the chameleon effect [7]), with evidence reported for a hand and face imitation circuit in the brain [38].

2.4 Active Selection: Motion Correlation inspired by Perceptual Control Theory

Motion correlation for selection in the user interface was first described by Williamson and Murray-Smith [68]. Their original publication presented a twofold motivation–a theoretical argument for treating selection as a continuous interaction with a dynamic system, and a practical argument for a method that avoided the need for a pointer (in light of new types of devices for which pointing can be impractical, e.g. wearables).

Inspired by perceptual control theory [36], they demonstrated a new form of interaction in which pseudo-random disturbances were introduced into the display of objects. The user's movement of an input device applied the same control to all objects, and selection was achieved by stabilising the intended objects. For this, the user varied their action so to cancel the effects of the object's disturbance, resulting in a correlation between the input and the disturbance. In this design, each object can be viewed as an agent that makes changes to its own state and looks for correlations in the users action. Williamson later described this concept as *active selection*, where selection is performed by testing behaviour in a closed loop [67].

Figure 1 shows two concept demonstrators for active selection. The first one is a Brownian motion interface where each object moved on pseudo-random trajectories and speeds (see Figure 1-A). Objects contain circles where the radius represents the selection probability as feedback to the user. The second demonstrator, *Eggheads*, shows multiple faces looking at pseudo-random directions. Moving the mouse controls the orientation of all faces, and to select a given head the user must move the mouse so that it faces forward (see Figure 1-B). Active selection was further illustrated with a Brownian motion interface for text entry [67], shown in Figure 2.

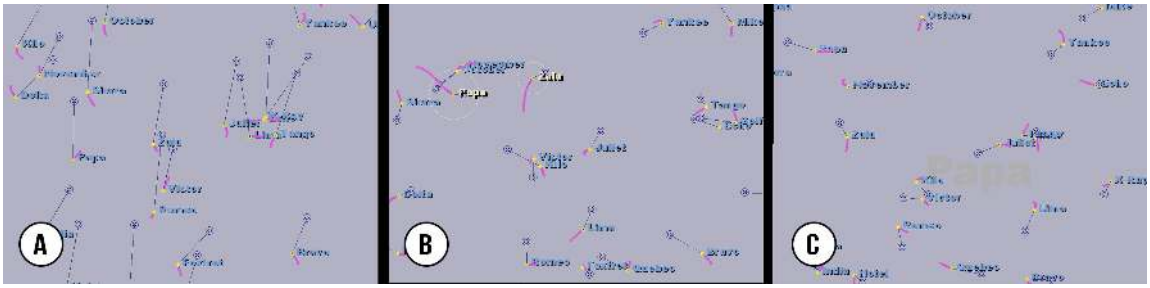

Fig. 2. Active selection for text entry: (A) Objects representing different letters move pseudo-randomly, with feedback on current position (yellow) predicted path (pink), and correlation with the user's input (blue line and dot); (B) As the user's input begins to stabilise the motion of any of the objects, the system provides feedback on its confidence in the selection (white circles); (C) Such ambiguities are resolved through continuous input until a single object is selected (image source: [67]).

This early work demonstrates the essence of motion correlation: that motion is used to couple input and output, in a continuous interaction between user and system. The specific approach in this work is to inject random motion to upset objects out of a state in which they are indistinguishable, and to define selection as a negative feedback loop toward regaining equilibrium. This contrasts our own use of motion correlation, where the principal idea is to present motion to which users couple in synchrony. However, there are key lessons that hold across the different perspectives: the mixed initiative of system and user in the interaction, the information conveyed with even very small movements, and the gradual resolution of ambiguity through continuous motion.

2.5 Motion Pointing: Motion Correlation for Selection in Graphical User Interfaces

Williamson and Murray-Smith's work inspired Fekete et al. to explore motion correlation as a selection principle in the context of conventional graphical user interfaces [14]. Rather than using pseudo-random movement, their idea was to associate objects with oscillatory movement, to draw on user's natural ability for harmonic motion with their hands. In their design of the *motion-pointing* technique, the graphical objects of interest retain their static presence in the interface but are augmented with a moving dot describing a small elliptical movement.

Figure 3 illustrates motion-pointing. In this example, buttons presented to the user are extended with a small red dot moving next to it, each describing a different movement. To select a button, the user matches the movement of its target with the mouse. As they found that the correlation between the elliptical motions and user input was not sufficiently accurate to reliably distinguish between multiple different shaped and phased ellipses, the system highlights four candidate buttons with a triangular shape marking the top, bottom, left or right edges. The user confirms the selection by dragging the mouse towards the corresponding direction. In this combination, the technique is called *Move-and-Stroke*.

The authors evaluated how well users could match the movement of the targets with the mouse. They observed three distinct phases in the movement matching. First, users spend 1-1.5 seconds trying to synchronise the mouse with the object. Second, they manage to synchronise and remain coupled for 1-2 seconds. After that, due to fatigue or lack of feedback, the coupling starts to deteriorate and the user must try to re-synchronise the movement.

The work on motion-pointing, although studied in a more abstract case of an interface, offers key insights of relevance to design of interfaces using movement. It demonstrates the utility of harmonic motion for input, and the use and

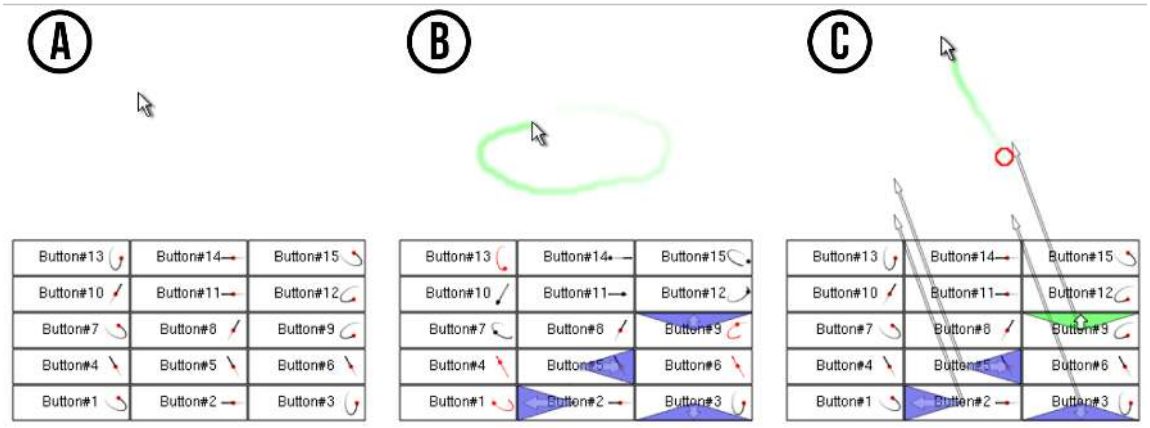

Fig. 3. Fekete et al.'s move-and-stroke technique: (A) Buttons are shown with motion icons indicating the movement required for selection; (B) When the user performs a movement, the top four matches are highlighted with a blue are facing up, down, left or right; (C) The user completes selection with a stroke in one of the shown directions. (image source: [14])

systematic variation of periodic motion as stimulus. However, it also highlights the challenge of designing motion to be sufficiently distinct for detection by correlation of input and output.

2.6 Related Work

Active selection and motion-pointing introduced motion correlation for selecting objects by their movememt. In related work, motion matching has been used for interaction across devices. *Smart-Its Friends* is a technique for pairing devices by shaking them together [29]. Because both devices were subjected to the same motion, they can use motion correlation to determine their connection. Hinckley discussed *synchronous gestures* more generally as "patterns of activity spanning a distributed system that take on a new meaning when they occur together in time" [26], for instance pen strokes performed across displays to connect them [27]. A range of techniques have followed where motion and other inputs are correlated for cross-device interaction, for example with mobile phones on interactive surfaces [52].

The correlation of movement has been used to determine user and device relationships, for example whether two devices are being carried by the same person [39]. An interesting case is matching of user movement (tracked with a depth camera) with the motion of their devices (tracked with built-in accelerometers) for disambiguation of multi-user input on shared displays, as it demonstrates motion correlation across different sensing modalities [15, 51]. Motion can also serve as a secret that devices share for pairing. *Shake Well Before Use* is an extension of the Smart-Its Friends concept with an authentication protocol that has devices agree on a key based on correlation of the motion they independently measure [44, 45]. These works have in common that they demonstrate motion correlation of different input signals, contrasting our focus on matching of input and output.

Some prior work, also in the context of device authentication, is related to presenting motion to the user for reproduction. 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 [48]. Other authentication schemes have been based on rhythm matching, for example using rhythmic input as password [69]. The *Harmony* protocol is an interesting

variant, proposing that a device targeted for pairing responds with multimedia feedback that the user can verify to be in harmony with their own device [34].

Also worthy of note is related work on the coupling of input and output by periodic motion. In *Resonant Bits*, the coupling is explored in terms of resonance and how a system's continuous feedback can guide the user's rhythmic input [3]. In *CycloStar*, continuous closed loop motion is used to support panning and zooming in touch interfaces in a clutch-free manner [41]. We have employed cyclic motion in similar ways in the design of gaze and mid-air gesture interfaces [8, 12].

3 FIVE APPLICATIONS OF MOTION CORRELATION AS INPUT

In this section, we review five of the present authors' recent works in which motion correlation was applied in the design of gaze and gesture interfaces: *Pursuits* [49, 63, 65], *Orbits* [12, 13], *PathSync* [6], *AmbiGaze* [61] and *TraceMatch* [8]. None of these works were designed to study motion correlation in the first place, but demonstrate the principle at work in giving rise to novel, practical and compelling interaction designs. We revisit the five works in chronological order, showing how our initial focus on gaze later expanded to gestures and human movement in general, and reviewing how motion correlation supports interaction with devices in public display, wearable, multi-user, and multi-device contexts. In each case, we analyse what specifically motivated our use of motion correlation, how the principle was applied, and what new insights we gain in retrospect.

3.1 Pursuits

Pursuits is a technique that leverages the smooth pursuit movements that our eyes make when following a moving target. To select a target, all the user has to do is to follow it with their eyes. As shown in Figure 4, the movement of the eyes is matched against the motion of potential targets, in order to determine the selection.

Our original motivation for *Pursuits* was to enable gaze input in a spontaneous manner, such that users could just walk up to a display and have the system immediately respond to their gaze [63, 65]. Prior to our work, gaze-based selection techniques had been based on *fixations*, the small hovering movements our eyes make when we look at a static target [31]. However, fixations are only effective for selection if the user's gaze is first calibrated to the display's coordinates. Our novel insight was that we can sidestep calibration, by using moving targets instead of static ones, and motion correlation instead of conventional gaze pointing.

3.1.1 Implementation & Evaluation. The *Pursuits* technique was implemented with an algorithm that computes the Pearson's correlation coefficient between the coordinates of the uncalibrated gaze point and a target object over a given time window, and selects the object if its coefficient is above a given threshold. The algorithm was evaluated with real user data to provide insight into its discriminatory power: indicating 'how different' the motion of displayed objects has to be (in direction, phase or speed) in order to warrant robust selection. The evaluation also examined the design trade-offs of longer windows for increased certainty of a match versus faster response, and higher thresholds for avoidance of unintended activation versus sensitivity of the technique. See Vidal et al. for the details [65].

We explored the use of *Pursuits* in a number of applications. In a first application, we designed a public information displays with content slowly drifting across the screen (see Figures 4-C,D). Each object moves in a different direction, providing an intuitive example where we can easily understand how users' eye movements will instantly reveal which object they are following. However, the user can completely abstract from the different movements; as a moving object

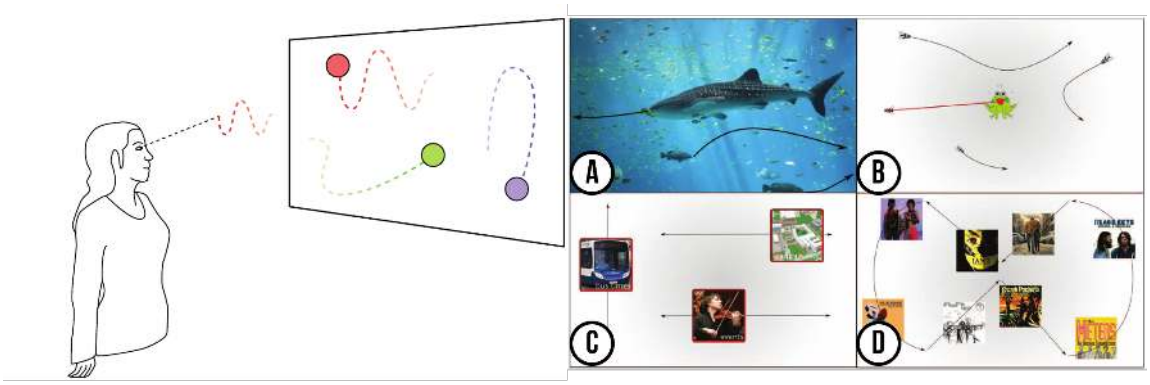

Fig. 4. The *Pursuits* technique for gaze interaction: objects of interest move on the display; to select an object the user simply follows its motion with her gaze: (A) an authentication display that unlocks when users follow different fish in the right order; (B) a game in which users follow the fast movement of flies to have them caught by a frog; (C) a gaze-aware campus information display; (D) a music player where clips are triggered by following one of the album covers drifting over the screen.

draws their interest, it automatically elicits a smooth pursuit response from their eyes. The system infers to which object they attend and provides additional information on the detected topic of interest.

Our second example, shown in Figure 4-B, is a game where the user helps a frog catch flies that are buzzing around the screen—when they follow one of the flies with their eyes, the frog will snap it up with its tongue. The game mechanic is simple and engaging, and players can easily take turns in front of the display, as the system does not require any adaptation to different users [64]. The game shows that *Pursuits* is effective with higher speeds and randomised trajectories of moving targets, and it highlights that the selection is target size independent, as targets are matched by their motion. This is significant for gaze input, as eye tracking has inherent accuracy limitations which makes it difficult to acquire smaller targets by gaze pointing.

The final example is an interface design for authentication (see Figure 4-A). The display shows different species of fish swimming across the display, and to log-in users must follow a certain sequence of fish in the right order. What makes motion correlation work robustly for such a task is that the pursuit movement of following an identified object (i.e., a specific fish) is fundamentally different from the saccadic movement of the eyes when we scan the screen for information (e.g., looking from fish to fish to identify the next in the sequence).

3.1.2 Pursuit calibration. With the pursuit technique, the absolute position of the gaze coordinate provided by an eye tracker does not matter. Because of the scaling performed in the correlation calculation, an object can be selected even when the eye-tracker is 'off target'. However, once the algorithm determines that the user has been following an object, these data can be used as input for calibration of the eye tracker's output. Based on this insight, we took the pursuits work a step further, and developed the *Pursuit Calibration* technique [49].

Figure 5 shows one of the application examples we developed to illustrate pursuit calibration. The application consisted of a constellation explorer. When users approach the screen, they see asteroids flying in space (see Figure 5-A). The movement of the asteroids naturally draws their attention and leads to a 'pursuit reflex'. This enables the system to collect data points for computation of a homography, mapping the eye tracker's output to the screen's coordinates. After calibration, the system is then able to respond to where users fixate, by displaying more detail (see Figure 5-B).

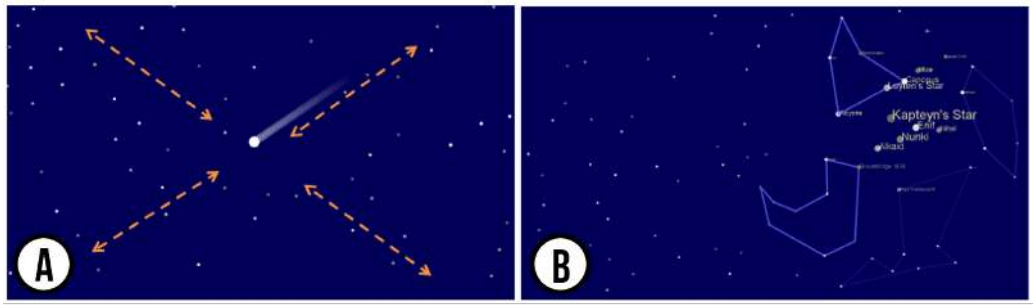

Fig. 5. Application of the *Pursuit Calibration* technique for exploration of stellar constellations: (A) Initially, a few asteroids are shown to attract gaze-following for calibration the system; (B) After calibration, the system seamlessly enables exploration of fixed constellations by gaze.

The fundamental insight underlying pursuit calibration is that users are only to able produce smooth pursuit eye movement when there is a moving target they can follow. When we present a target with unique movement and observe the corresponding eye movement, we can robustly infer that the target is being attended. We previously highlighted and evaluated the advantages of pursuit calibration over conventional gaze calibration—it can be seamlessly embedded within applications, it is tolerant to user disruption, and it can be achieved without the user being aware of the calibration process. However, it is also interesting what the work demonstrates as input technique, specifically how movement is used to overcome the uncertainty of pointing.

3.1.3 Key insights for the design of motion correlation interfaces.

- **Leveraging properties of eye movements**: The eyes produce a distinctive smooth movement when they follow a moving target. This allows us to detect attention to a moving object simply by correlating natural gaze behaviour with presented motion, for object selection as well as eye-tracker calibration.
- **Implicit Interaction**: In earlier work, users had to actively reproduce presented motion, whereas with gaze we can leverage the reflexive nature of human visual attention to movement. The resulting experience is one in which the interaction dissolves with users' natural behaviour.
- **Unaware Input**: Humans have limited self-awareness of the agency of their eyes. Beyond implicit interaction, interfaces can be designed to engage users without active awareness of providing input, as demonstrated for calibration.

3.2 Orbits

Orbits is a technique that brings smooth pursuit-based selection to smart watches. As these devices offer increasingly more functionality, finding suitable ways to interact with them becomes a more difficult problem due to their small size. The first wrist watches replaced pocket watches thanks to how they made it easy to read the time, even when the wearer's hands were busy [22]. Analogously, the motivation for this work was to enable hands-free input on smart watches.

Traditional mechanical watches already contain multiple moving objects in the form of dials and gears. The design of *Orbits* is inspired by these aesthetics (see Figure 6-A). Interactive objects are presented by circular trajectories, which have multiple targets 'in orbit'. Note how this design supports display of different input options in the context of very

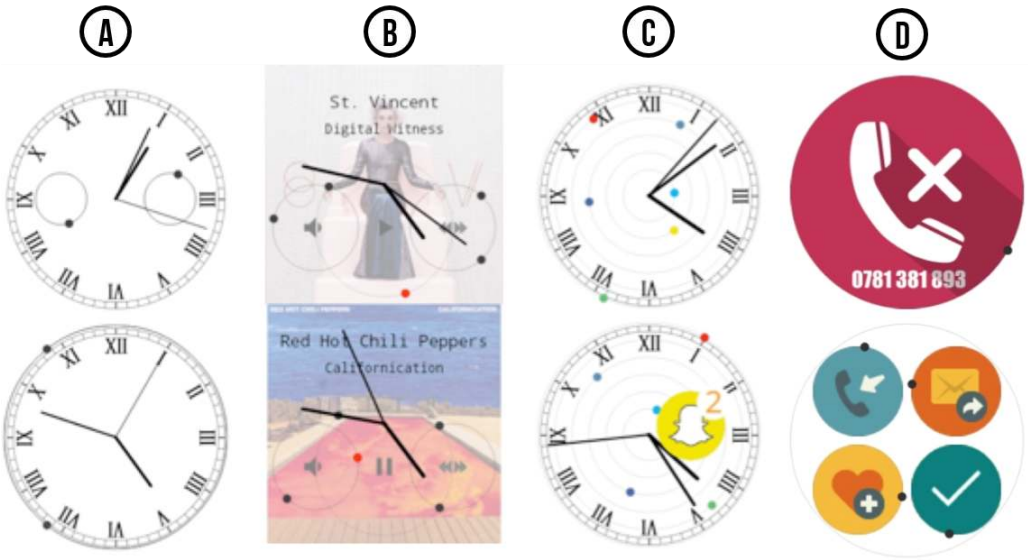

Fig. 6. The design of *Orbits* for gaze input on smart watches. (A) Two watch face designs: smaller orbits that each have a single target (top), and a larger orbit with two targets moving in different phase (bottom). (B)-(D): Application-specific designs for a music player (A), notifications (C), and missed call handling (D).

limited screen real estate. Each target performs a distinct function and is activated when the system detects a high correlation between the motion of the target and the movement of the user's eyes. Therefore, all the user has to do to issue a command, is to follow the corresponding target with their eyes.

3.2.1 Design and evaluation of orbiting controls. Orbits are conceived as explicit controls that are triggered by gaze pursuit of the moving target orbiting the widget. The selection mechanism was based on the original Pursuits algorithm but the smart watch context raised new questions. Given the limited screen real estate, we needed to understand the spatial extent of motion required to invoke a pursuit response, attention to competing motion on a small display, and how other information could still be accessed without unintended pursuit input. In an initial study we recorded eye movement in interaction with varying number, sizes and speeds of orbits, as well as gaze data on non-pursuit tasks, such as checking the time, reading text, and watching video. This showed that the pursuit algorithm can be optimised for detection of orbits as small as 0.6° visual angle, while avoiding false positives. We then evaluated user performance with orbits, and observed that even with an increasing number of orbits, users were able to select targets with high accuracy (e.g., 84% when 8 orbits simultaneously displayed motion). This was an important result for us as it showed that users are able to attend a displayed motion selectively, even in the presence of multiple distractors within the same small display space.

3.2.2 Smart watch applications. We built three prototypes to capture qualitative insights about the technique in likely real world applications—a Music Player, a Notifications Face Plate and a Missed Call Menu. The Music Player displays an orbit each for volume control, play/pause and skipping to previous/next song (Fig. 6-B). Play and pause are toggled by following the same target, while the other controls have two targets each orbiting in the opposite direction. Volume, for instance, is increased when the user follows the clockwise-moving target with their gaze, and

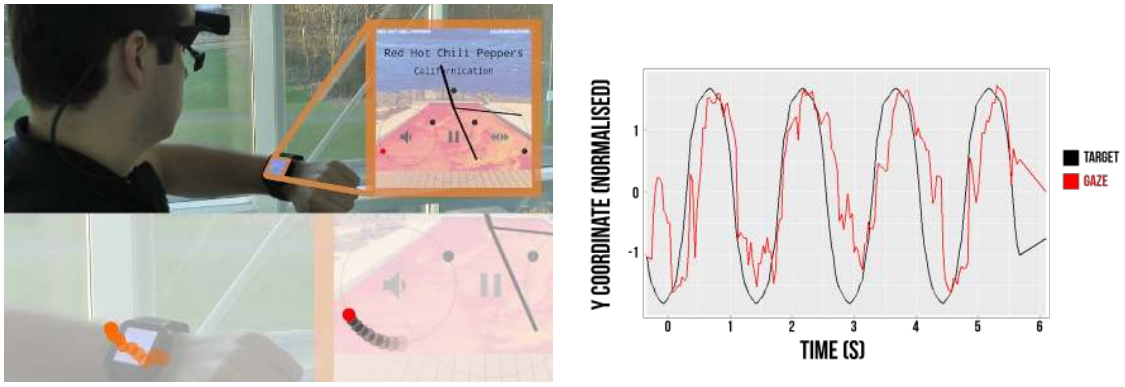

Fig. 7. Left: A user controls the music playback with *Orbits*, turning up the volume by gaze pursuit of the associated target on the watch interface. Right: Example time series of target motion and gaze pursuit.

decreased when they follow the other one. Figure 7 shows the interface in use, and a sample of an orbit's motion with corresponding gaze. The user's gaze is tracked with a portable eye tracker, but gaze coordinates are streamed directly to the smart watch application, without any mapping to the tracker's scene view because this mapping is not required. As a result, Shimizu et al. later demonstrated how the technique also works with EOG glasses, which have severe limitations in estimating absolute gaze points because of data drift, but are great at measuring relative eye movements at high frequencies [54].

The Notifications Face Plate consists of a notification panel in which six targets move on nested orbits behind the hands of the watch (Fig. 6-C). Each target corresponds to a different application and has a matching colour scheme (e.g. blue for Facebook, yellow for Snapchat). The radius of the trajectory represents the number of unaddressed notifications, increasing as more arrive. The application with the most recent notification moves in the opposite direction to the rest to make it more noticeable. When users follow a target, the little coloured dot becomes a small icon with the application logo. If the user keeps following the logo, the app is invoked. This prototype highlights how orbits of different radii can be selected robustly even though they circle each other at different speeds. It also illustrates that extra quantitative (i.e. number of notifications) and temporal (e.g. most recent) information can be encoded into the movement.

The Missed Call Menu supports discreet call handling (Fig.6-D). When the user misses a call, an icon with the contact number appears with an orbiting target. When the user follows this target, four options appear with their corresponding targets: call back, text back, save contact, and dismiss the notification. This example shows how motion-based techniques can support conventional styles of menu-based interaction, where orbits appear like buttons that can be 'pressed' by following the orbiting target.

3.2.3 *Key insights for the design of motion correlation interfaces.*

- **Small Displays**: In a pointing paradigm, the larger the target, the easier it is to select it. With motion correlation, the size of the target itself does not matter, as long as its movement covers enough of a distance to produce a meaningful eye movement in terms of visual angle.
- **Target Density**: Our prototypes show that users can still make accurate selections when many targets move in a dense arrangements, even with overlapping and crossing trajectories. However, it is important to keep in mind

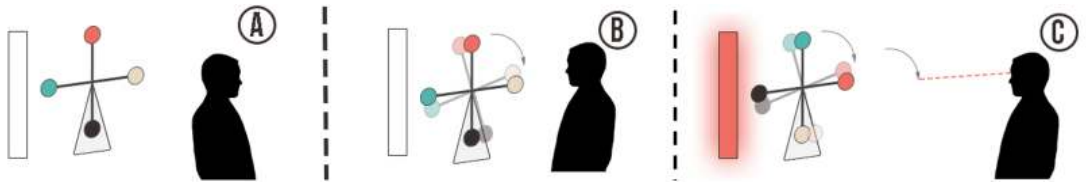

Fig. 8. *AmbiGaze* is designed for gaze only control of objects in a smart environment: (A) Objects remain motionless when they are not attended by the user. (B) When the user faces an object, it responds by presenting available controls as moving targets. (C) The user selects a target by gaze pursuit. In this example, a light's color changes according to the windmill paddle the user is looking at.

that even though it is possible to make such selections, an excessive number of targets can be distracting and may make it difficult for users to find the desired target before locking onto it.

- **Decoupling of Input and Output**. The use of a head-worn camera to track the eye was a pragmatic choice in this project—for a smartwatch interface it might be preferable to have tracking built into the watch. However, it highlights an interesting separation between input and output. Eye tracker and smart watch operate with different coordinate systems without any need for registration and translation, as the relationship of input and output is established solely by motion correlation.
- **Encoding Extra Information in the Movement**: The movement itself can be used to encode additional details about the command it triggers. The target speed, the radius of the trajectory, and the direction of movement can all encode meaning that can be instantly understood by users.

3.3 AmbiGaze

The *AmbiGaze* project investigated how gaze input based on motion correlation can scale to control of different devices in a smart environment [61]. We considered two principal challenges: how motion can be used for dynamic association of gaze with different devices while avoiding continuous motion in the environment; and how different devices (including ones without screens) can present motion to the user for input based on the pursuit technique. Figure 8 illustrate the interaction design. To initiate the control of a device, users direct their attention to it. The device responds by displaying control options as moving targets, which the user selects via the pursuit selection technique.

3.3.1 System Architecture. AmbiGaze relies on an architecture in which each device sends the normalized coordinate of each of their targets to a central server. User input is also captured and sent to the server. Due to the limited tracking range of current remote eye trackers we used a wearable eye tracker. To mitigate the issue of having controls with similar movement patterns in a distributed context, we incorporated an attention-awareness component inspired by Vertegaal et al.'s work on attentive user interfaces [62]. This component estimates which device the user is facing, reducing the number of targets it attempts to correlate with the user's gaze. We implemented attention-awareness using infra-red beacons attached to each of our ambient devices. Each beacon emits a unique light pattern that is picked up by the wearable eye tracker's scene camera (modified with an IR filter). This not only minimizes selection errors (as there are fewer controls to select from), but also manages when a device should display moving stimulus.

3.3.2 Design Space Exploration. We devised a taxonomy for the different ways in which motion can be embedded into a smart environment. This is defined by the way in which motion is generated (virtual x mechanical), and where it is displayed (internal or external to the device). The taxonomy is illustrated with four devices we integrated in the *AmbiGaze* environment, as shown in Figure 9.

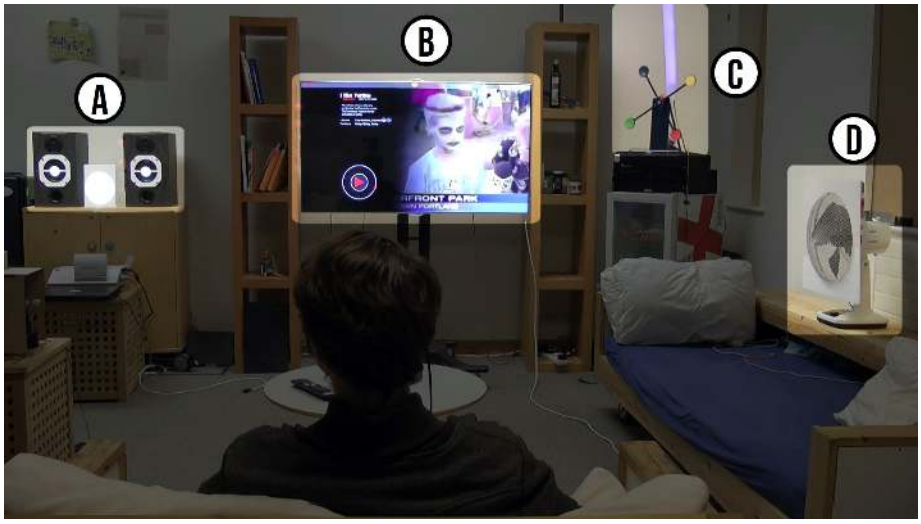

Fig. 9. In *AmbiGaze*, devices display motion by virtual or mechanical means, either generated by the device itself (internally) or by other devices (externally). This is illustrated with four devices integrated in a smart environment: (A) Speaker system; (B) TV interface; (C) Windmill lamp; (D) Fan.

Virtual + Internal: The device renders its own animated graphics. Our example prototype is an interface for a video-on-demand (VoD) service on a large TV screen. Users select which video to watch next, or obtain additional information about a video, by following the motion of targets orbiting different graphical controls such as a play button (Figure 9-B).

Virtual + External: Inspired by *Illumiroom* [32], an external device projects a graphical overlay with moving controls over the interactive device—this is particularly useful when the device does not have a display. Our example is a Music Player in which graphic controls are projected onto a pair of speakers (Figure 9-A).

Mechanical + Internal: The device generates physical movement. Our example is a multi-color lamp that generates moving controls using an electric windmill (see Figure 9-C). Each paddle of the windmill is represented by a different color, and following any of the paddles triggers the corresponding color in the lamp.

Mechanical + External: An external device uses a laser pointer mounted onto a robotic arm to generate motion around the target device—this is particularly useful when the device does not have any actuator. Our example is a digitally controlled fan. We mounted a white acrylic sheet around the fan as surface for projection of a laser's red dot. The robotic arm traces a square around the fan, which the user can follow to switch the fan on or off (see Figure 9-D). The same laser pointer could be used for different ambient devices in the environment, simply by following the user attention to different devices.

3.3.3 Key Insights for the design of motion correlation interfaces.

- **Movement beyond screens:** This project demonstrated that the movement required for enabling correlation interfaces can be generated by means other than screens. As long as the object exhibiting the movement can estimate the relative coordinates of its moving parts, they can be used as targets.
- **Multi-step interaction:** When used in excess, moving targets can be distracting. *AmbiGaze* demonstrates how we can minimize this problem by only displaying movement in devices the user is facing.

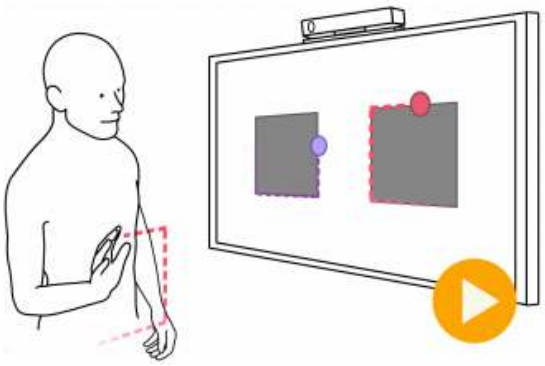

Fig. 10. In *PathSync*, to select an object, users match the movement of the target around it with their hands in mid-air

- **Distributed Interaction Processing:** This project demonstrates distributed management of the interaction process. Each device only broadcasts the relative position of its targets, which are correlated to users' gaze data by a central server. This allows the system to scale to multiple users and devices.

3.4 PathSync

The three projects above demonstrate how moving targets eliciting smooth pursuits can create suitable eyes-only interaction techniques. *PathSync* extends this principle to mid-air gestures (see Figure 10) [6]. Whereas when engaging in a smooth pursuit the eyes naturally lock onto a moving object, we were not sure whether users would be able to do this as easily with their hands in mid-air. Other mid-air gestural interaction techniques work either by mapping the position of the hand to a cursor on the screen or by offering the user a library of gestures of different shapes that they can use to issue commands. In *PathSync*, rather than using the shape of the gesture to disambiguate the command being issued, we use the timing of the gesture.

3.4.1 Implementation & Evaluation. We investigated *PathSync* from three angles. First, we tested the feasibility of the technique in an abstract task in which we asked participants to simply follow the moving target on the screen. We varied the shape (circle, square, diamond, rounded square, and a randomly generated squiggle), speed, and position of the trajectory on the screen. All participants were required to do was to move their hand in synchrony with the movement of the target, and no feedback was given as we recorded them with a Microsoft Kinect sensor. Based on the data collected in this study we made improvements to the original *Pursuits* algorithm, which we discuss later in this paper. The results also led us to choose square trajectories as they yielded the highest average correlation, suggesting that users find it easier to match that shape.

Our second evaluation measured *PathSync*'s performance in comparison to the baseline technique used in the XBox console. In our study, participants completed a tutorial for both techniques and were asked to select a specific option among others in a Windows Metro-style interface. The tutorial was designed to match the procedure and style of the interactive tutorial that ships with the Kinect SDK. We found no significant differences in performance between the two techniques, suggesting that *PathSync* performs similarly to the state-of-the-art, despite using a fundamentally different interaction principle.

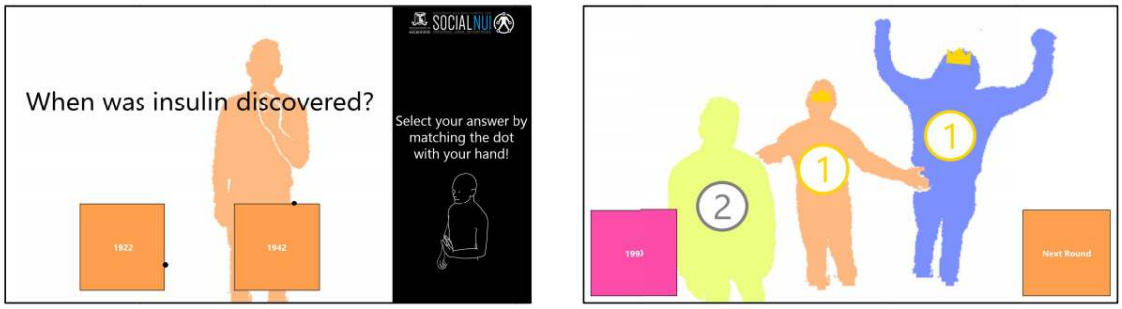

Fig. 11. Social NUIz: A multi-user quiz game in which players select their answer with PathSync. Left: 'Attract' screen. Right: After each round, players are displayed by their silhouette with their score on their chest and a crown if they got the right answer in this round.

Our third and final evaluation concerned how the technique performs in an ecologically valid situation. We deployed a multi-user quiz application called *Social NUIz* on two on-campus public displays, each equipped with Microsoft Kinect sensor (see Figure 11). *Social NUIz* is a multiple-choice quiz game on General Knowledge and University trivia playable by up to six people—the number of bodies that the Kinect can simultaneously track [9]. In each round, players discretely select an answer using *PathSync* from up to four possible answers displayed within a time limit. The remaining time is skipped if all players have made a selection, and those who have selected the right answer is crowned and receives an increment in their score.

To introduce the technique to a new audience, we also developed an 'attract' mode that presented users with a question that had two possible answers. When users were detected, their silhouette was displayed on the background, emphasising that they were being tracked by the Kinect. In subsequent rounds, instead of a silhouette, users were depicted by a coloured vertical bar. We modified the visualisation because a silhouette would make it easier for other players to infer which answer a given player was selecting. From a large sample of over a thousand interactions, we were able to validate *PathSync* as a suitable technique for multi-user interaction with public displays.

3.4.2 Key Insights for the design of motion correlation interfaces.

- **Making Gestures Visible:** A major problem in gestural interfaces is communicating to the user which gestures are available in the interface. By asking users to follow a moving target, *PathSync* makes all of its available commands clearly visible on the interface.
- **Multi-User Interaction:** Even though body-based interfaces made possible by sensors like the Microsoft Kinect generally afford more social user experiences, cursor-based interaction techniques are awkward to be used by many users simultaneously. In *Social NUIz* we demonstrated how users can step in and out of the interaction, selecting options on the interface without the need for a cursor and without revealing to other players which option they chose.
- **Corners work as Sync Points:** In our evaluation we found that users could more easily match the movement along shapes that had corners, such as the square and the diamond. These points acted as salient opportunities for synchronising the movement of the hand with the movement of the target.

Fig. 12. *Tracematch* is designed for casual motion input. A camera tracks any motion in the front off the screen and correlates it with on-screen moving control. This enables users to provide input with any part of their body, and regardless of their posture. In this scene, the user is able to close a pop-up screen on the TV by mimicking the widget's motion with his hand even though he is holding a drink.

Fig. 13. TraceMatch processing pipeline. (A) Features detected from FAST and previous optical flow iteration. (B) Candidate features (green) and non-candidate features (red). (C) Features below the Pearson product-moment correlation coefficient threshold (red), features above the threshold (green), and the first feature to be matched is shown with its trajectory and the fitted circle from RANSAC (blue).

3.5 TraceMatch

The previous projects demonstrated how the same principle of motion correlation can be used with two different modalities, namely gaze and mid-air hand gestures. However, it is easy to imagine how the technique could work with other human movement modalities, such as head [35] or feet movements [58]. With this in mind, we developed *TraceMatch*, a computer vision technique that searches for matching movements in a video feed, effectively enabling interaction based on motion correlation regardless of the modality being used. This was also motivated by the idea of supporting 'lazy' input that requires minimal effort for mundane tasks, such as controlling the lighting or the TV when lounging on a couch. In situations like this, the user might lean on one hand and hold a cup with the other, but they should still be able to provide input—for example, with a head gesture or the hand even though it is holding the cup (Fig. 12). Therefore, we designed the motion tracking and matching without any assumption about users' postures or the body parts they might use for input.

3.5.1 Implementation & Evaluation. The computer vision algorithm in *TraceMatch* works in two stages. First, the algorithm searches for any kind of movement in the scene, without segmenting particular parts of the image, such as hands or heads. More specifically, we use a FAST feature corner detection to find points of interest in the scene,

which are then fed into an optical flow component (see Figure 13-A). We then compare the trajectories of each moving feature point (see Figure 13-B) to the movement of the target on the screen, triggering a selection when the correlation coefficient crosses a given threshold. In this work, we focused only on circular trajectories, so we configured the correlation algorithm to fit the tracking data into a circle using RANSAC, ensuring that no spurious high correlations would be triggered by similar trajectories with different shapes (e.g. elliptical) (see Figure 13-C).

We evaluated the system in regards to its sensitivity and robustness to false positives in a user study. For this purpose, we built an application for smart TVs that shows a widget overlaid on the program that the user can match to see further information about the content on the screen (See Figure 12). The study showed that users differ considerably in performance, preference and spontaneous choice of body movements for input. For example, some found motion matching with head movement difficult, whereas others found it least demanding in terms of effort.

3.5.2 Key Insights for the design of motion correlation interfaces.

- **Abstracting from body modality:** *TraceMatch* searches for any kind of movement in the view of the camera. This means that users can not only interact using whichever body part they wish, but also interact whilst holding other objects.
- **Lazy Input:** In our tests, we found that even small body movements if performed in synchrony with the movement on the screen would trigger selections. This allows for a type of lazy input that is rare in body-based interaction.
- **Posture-agnostic:** As long as users perform a movement that matches that in the screen, they can stand, sit, or lie down to interact with the device. By only searching for movement, our algorithm goes around the challenges of segmenting specific body parts, especially in a cluttered environment.

4 DESIGN GUIDELINES

The above sections have introduced the motion correlation principle, and discussed our experience of designing a novel interfaces in which the principle has been applied in different ways. We now distill the knowledge built across these works into a set of guidelines to support the design of future interfaces that employ selection by motion correlation. We take a bottom-up approach, starting from considerations about algorithm parameters, moving on to the choices about widget design, and concluding with broader choices about putting them together into a cohesive interface. Our aim is to give the reader a sense of the practical decisions that must be taken when designing such systems.

4.1 Algorithm Design

The basic principle in correlation-based interfaces is that it is possible to estimate which target the user wishes to select by how closely the motion of an input modality matches the element of that target. Therefore, quantifying motion similarity is an important aspect of this process. In this subsection, we describe the different decisions in configuring a Pearson's correlation comparison algorithm, evolved through a series of works described previously [6, 12, 13, 61, 63–65]. We also provide the complete pseudo-code of the algorithms used in *Pursuits* [65] and *Orbits* [12] (see Algorithm 1) and its extension used in *PathSync* [6] (see Algorithm 2).

4.1.1 Window Size. Regardless of the choice of algorithm, the comparison of the motion of the target to the motion of the input device must occur over a given time interval, determined by the *window size*. In practice, this determines how much data the algorithm must sample in order to begin the calculation. The recognition performance strongly

ALGORITHM 1: Pursuits Algorithm

Input: A $T \times 2$ matrix with 2D positions of the input device $P(In) = \left\{x_t^{In}, y_t^{In}\right\}$ at time t; a $T \times 2$ matrix with 2D positions $P(Out_i) = \left\{x_t^{Out_i}, y_t^{Out_i}\right\}$ for each of the I moving targets Out_i on the screen at time t; a correlation threshold $\tau \in]0, 1[$; a window size $N < T$.
Output: A Boolean vector $Selected$ of length T, representing which target Out_i, if any, is selected by the input device at time t.

```
for t ∈ 0..T do
    if (t < N) then
        Selected_t ← None;
    else
        for Out_i do
            c_i ← min(cor(x_t^In, x_t^Out_i), cor(y_t^In, y_t^Out_i));
            where cor(a, b) = E[(a−ā)(b−b̄)] / (σ_a σ_b);
        end
        if ∃c_i | c_i > τ then
            Selected_t ← Out_k | c_k = max(c_i)
        else
            Selected_t ← None
        end
    end
end
```

depends on how large this window is. Whereas small windows lead to more responsive triggers, it can also lead to more spurious activations. On the other hand, whereas increasing the window size may decrease the number of false positives, it will also lead to more lag between performing the gesture and triggering a selection. A large window size also tends to yield lower correlation coefficients as it averages a larger amount of data. When choosing a window size it is important to take into account the sampling rate of the input device. For example, whereas a window size of 30 samples might represent a whole second of data in a 30Hz sensor (e.g. an RGB webcam), it can represent only a tenth of a second in a high-speed 300Hz sensor (e.g. Tobii TX300 eye tracker). Typical window sizes evaluated in previous works range from 100ms [65] to 1.3s [12].

4.1.2 Comparison Function. The *comparison function* is the metric that quantifies how closely the two motions match. Fekete et al. compared three metrics: Euclidean distance (see Equation 1), normalised Euclidean distance (see Equation 2), and the sum of the coordinates' correlation coefficients (see Equation 3), calculated over the delta vectors of the input device and the target. These authors eventually chose the normalised Euclidean distance in their prototype implementation.

$$f_{Euclidean}(In, Out) = \sum_{t=0}^{W} \left[\left(\frac{\Delta x_t^{In}}{\Delta t} - \frac{\Delta x_t^{Out}}{\Delta t} \right)^2 + \left(\frac{\Delta y_t^{In}}{\Delta t} - \frac{\Delta y_t^{Out}}{\Delta t} \right)^2 \right]^{\frac{1}{2}} \tag{1}$$

$$f_{NormEuc}(In, Out) = \sum_{t=0}^{W} \left[\left(\frac{\Delta x_t^{In}}{\Delta t||In||} - \frac{\Delta x_t^{Out}}{\Delta t||Out||} \right)^2 + \left(\frac{\Delta y_t^{In}}{\Delta t||In||} - \frac{\Delta y_t^{Out}}{\Delta t||Out||} \right)^2 \right]^{\frac{1}{2}} \tag{2}$$

$$f_{SumCor}(In, Out) = \sum_{t=0}^{W} \left[\left(\frac{\Delta x_t^{In}}{\Delta t} \times \frac{\Delta x_t^{Out}}{\Delta t} \right)^2 + \left(\frac{\Delta y_t^{In}}{\Delta t} \times \frac{\Delta y_t^{Out}}{\Delta t} \right)^2 \right]^{\frac{1}{2}} \tag{3}$$

ALGORITHM 2: PathSync Algorithm

Input: A sequence of T 2D positions of the input device $P(In) = \left\{x_t^{In}, y_t^{In}\right\}$ at time t; a sequence of T 2D positions $P(Out_i) = \left\{x_t^{Out_i}, y_t^{Out_i}\right\}$ for each of the I moving targets Out_i on the screen at time t; a lower correlation threshold τ_{lo} and a higher threshold $\tau_{hi} \in]0, 1[$; a window size $N < T$; a highlighting window size h.

Output: A $T \times I$ matrix $States$, where each cell $States_{t,i} \in \{Deactivated, Activated\}$ represents the state of target Out_i at time t.

$t = 0$; $Selected = \{\}$;
for $t \in 0..T$ **do**
  if $(t < N)$ **then**
    $\forall i | States_{t,i} = Deactivated$;
  else
    for *each target* Out_i **do**
      $RotationMatrix \leftarrow \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix} \times \begin{bmatrix} \vec{v}1_x & \vec{v}2_x \\ \vec{v}1_y & \vec{v}2_y \end{bmatrix}$
      where $\vec{v}1$, $\vec{v}2$ are the eigenvectors of the matrix with $\left\{x_t^{Out_i}, y_t^{Out_i}\right\}$ as columns.
      $\begin{bmatrix} \vdots & \vdots \\ u_t^{Out_i} & v_t^{Out_i} \\ \vdots & \vdots \end{bmatrix} \leftarrow \begin{bmatrix} \vdots & \vdots \\ x_t^{Out_i} & y_t^{Out_i} \\ \vdots & \vdots \end{bmatrix} \times RotationMatrix \quad \begin{bmatrix} \vdots & \vdots \\ u_t^{In} & v_t^{In} \\ \vdots & \vdots \end{bmatrix} \leftarrow \begin{bmatrix} \vdots & \vdots \\ x_t^{In} & y_t^{In} \\ \vdots & \vdots \end{bmatrix} \times RotationMatrix$
      $c_i \leftarrow min(cor(u_t^{In}, u_t^{Out_i}), cor(v_t^{In}, v_t^{Out_i}))$;
      where $cor(a, b) = \frac{E[(a-\bar{a})(b-\bar{b})]}{\sigma_a \sigma_b}$;
      if $c_i < \tau_{lo}$ **then**
        $States_{t,i} \leftarrow Deactivated$
      end
      if $c_> \tau_{hi}$ **then**
        $States_{t,i} \leftarrow Activated$
      else
        $States_{t,i} \leftarrow States_{t-1,i}$
      end
    end
  end
end

Vidal et al. used a formula based on Pearson's correlation, but instead of adding the correlation coefficients along each axis as above, they discarded the largest one (see Equation 4 and Algorithm 1). This was the same algorithm employed in *Orbits*.

$$f_{MinCor} = min(cor(x_t^{In}, x_t^{Out_i}), cor(y_t^{In}, y_t^{Out_i}))$$
$$\textbf{where } cor(a, b) = \frac{E[(a-\bar{a})(b-\bar{b})]}{\sigma_a \sigma_b} \tag{4}$$

The conventional way of computing the correlation coefficient assumes the data on each axis is normally distributed. This causes problems for trajectories in which the data corresponding to the shape does not follow this assumption. For example, in the case of a target moving back and forth along a horizontal line, the value of its position on the vertical axis is constant. When calculating the correlation using Equation 4, the value of the coefficient goes to infinity, as it is divided by a standard deviation of zero. Whereas this problem could be addressed on a case-by-case basis by only considering one axis at a time, even in cases where the value can still be calculated, it can cause problems (e.g. at the

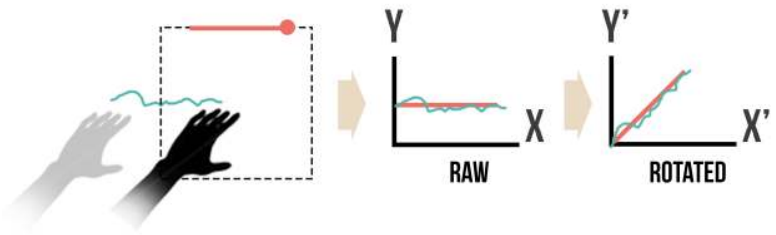

Fig. 14. In *PathSync*, before calculating the correlation, we rotate the shape according to its eigenvectors in order to distribute the variance across both axes.

extremes of a circle or edges of a square). To address this problem, in *PathSync*, we extended this algorithm by rotating the shape before calculating the correlation (see Algorithm 2 and Figure 14) [6].

When choosing a function, it is important to consider the possible values that it can output. An advantage of functions based on Pearson's correlation is that the values always lie between -1 and 1. This also helps derive meaning from these values: if the value in both axes is close to 1, they are moving in tandem; if the value along one of the axes is close to -1, the target is moving in the opposite direction on that axis. Fekete et al.'s Euclidean distance, for example, is not bounded, so further consideration is needed to calibrate the threshold and understand the meaning of the values.

4.1.3 Correlation Threshold. Once the correlation function is applied over the window, it will output a value that represents the degree to which the motion of the input device matches the motion of that particular target. To determine whether the user was indeed following the target in that window, it is necessary to set a *threshold* to separate the two cases. The correlation threshold will determine the sensitivity of the recognition, subject to a trade-off between true- and false-positive rates: a low threshold will be more lenient with poorly matched motion, but more susceptible to false positives, whereas a high threshold will be more robust, but will require more precision from the user in the motion matching.

This value can be empirically determined based on the system's requirements. For example, Figure 15 shows how the true- and false-positive recognition rates behave as we manipulate the correlation threshold, where each curve represents a different window size in the data we collected in the design of *Orbits* [12]. We chose a point in the plot closer to the left size, penalising our true-positive rate in return for more robustness to false-positives.

A further modification that can improve the performance recognition is the use of a bi-level threshold [46], as we did in *PathSync* [6]. As shown in Figure 16, we set two thresholds (0.6 and 0.9). If the target is not activated and the correlation crosses the lower threshold, nothing happens (Point A), but if it crosses the higher threshold, it becomes highlighted (Point B). From then on, as long as the correlation does not go below the lower threshold, the target remains highlighted, even if it goes below the higher threshold (Point C). After a second being highlighted, the target becomes activated (Point D), and will continue to trigger until it goes below the lower threshold (Point E). Once it does, the target will no longer be activated and crossing the lower threshold will not be enough to become highlighted (Point F).

An alternative modification to improve performance recognition is to introduce model fitting. In TraceMatch, we instead added a model fitting algorithm to ensure that motion passing the correlation test also conforms to an expected shape [8]. This ensures that similar motions which may exhibit high correlation values to the expected target (e.g. highly elliptical shapes when matching to a circle) are not registered as a match. This modification is required for modalities such as the hand or head, where smooth movements can be reproduced voluntarily by the user or accidentally by

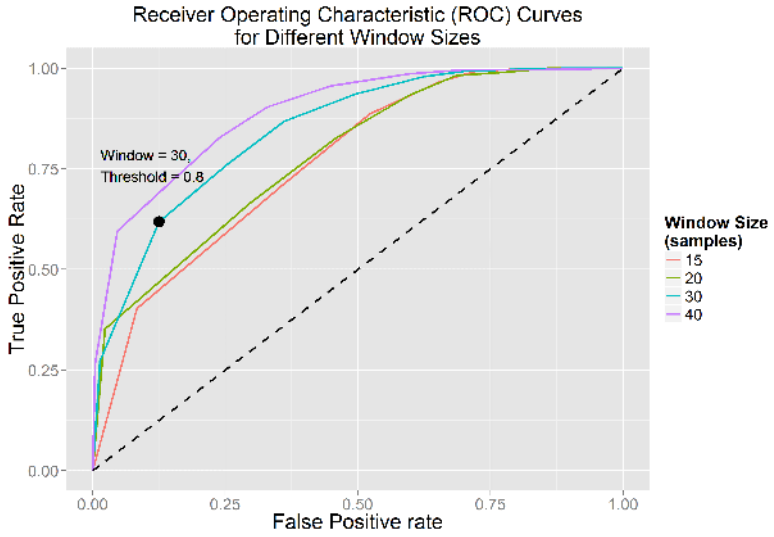

Fig. 15. ROC curve showing the relationship between false- and true-positive rates in *Orbits* for different window sizes as we manipulated the correlation threshold.

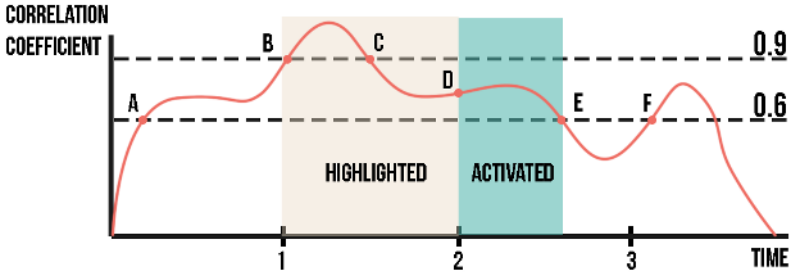

Fig. 16. In *PathSync*, we use two correlation thresholds. After crossing the upper threshold (B), the target is highlighted and remains so until either it is activated after a certain amount of time (D) or it crosses the lower threshold (E).

another stimuli. However, it is not required for the eyes because a smooth pursuit can not be reproduced without the resepective external stimuli.

4.1.4 Disambiguation. The architecture of the system impacts how the algorithm will be able to determine which target is being selected in *ambiguous* situations. In principle, the algorithm should work in a distributed fashion—each moving target could monitor the input device and check when the two motions match, triggering when appropriate. However, by design or by coincidence, it might happen that the motions of two distinct targets are similar within a given window. In a distributed architecture, this would cause both targets to trigger. This can be solved with a central server that checks not only whether the correlation coefficient crossed the threshold, but also which target yields the highest, and only trigger the action associated with that target.

In *AmbiGaze*, we had distinct ambient devices presenting independent moving targets. In such an architecture, we could not guarantee that two devices would not present similar trajectories, and even with a central server, it would be difficult to disambiguate the selection. We addressed this problem with a two-step activation—the user first looks at the desired device, causing it to start displaying its moving targets; from then on, following a moving target with their eyes activated the corresponding target [61]. By matching a user with a device, the system would only compare the motion of the input device to the motion of that device's targets.

4.2 Motion Design

Not even the most robust algorithm would be able to tell apart two targets with the same motion. Therefore, understanding how to design the motion of the targets is crucial in creating a robust recognition system. In this subsection, we discuss the different parameters to take into account when setting the different targets in motion.

4.2.1 Random x Determined. This decision determines whether the motion parameters are generated stochastically through a probability distribution or pre-determined by the interface designer. For example, Williamson et al.'s *Brownian Motion* and *Eggheads* prototypes [67] are randomly generated, whereas our own work uses pre-determined shapes, such as circles in *Orbits* [12] and squares in *PathSync* [6].

Random motion tend to be highly distinct, as the always changing parameters will likely generate substantially different trajectories. Random generation also tends to create interesting and organic motion patterns that might better suit certain interface metaphors (e.g. swimming fish or flocks of birds). However, they can also be somewhat unpredictable, making them more difficult for users to follow. This can be minimised by displaying the subsequent positions and by randomizing parameters of higher-order, which makes the motion smoother. Another challenge for this type of motion is that they lack a point of reference for the user to find the specific target they wish to trigger, as the position of the targets on the screen is unpredictable.

Pre-determined motions have the advantage of being predictable and more easily recognisable. They also allow for closed loops, which lead to periodic gestures. However, they can also lead to similar motions that are difficult to distinguish. Below, we discuss the other parameters that can be manipulated in periodic motions in a closed loop that help the system to tell different targets apart.

4.2.2 Trajectory Shape and Size. The literature presents examples of prototypes and studies with motion correlation trajectories of a wide range of shapes: straight lines [49, 65], ellipses [14], circles [8, 12, 61], squares [6], triangles [6], etc. The trajectory parameters often influence the difficulty of the movement and the robustness of the recognition. For example, in *PathSync*, we compared the correlation yielded when participants tried to follow trajectories of 5 different shapes with their hands (see Figure 17): a circle, a square, a diamond, a rounded square, and a randomly generated closed shape (squiggle). As expected, the squiggle was the hardest shape to follow, followed by the circle and the rounded square. The easiest shapes to follow were the square and diamond. We believe that the reason behind it is that the corners of these shapes give users a "checkpoint" to synchronise with the target motion. This principle is also used in juggling, where novices are often instructed to "look at the highest point" and to "throw the next ball when the previous one reaches the top" [2]. However, the particular difficulties created by the trajectory shapes will strongly depend on the input device.

The size of the trajectory of the screen may or may not impact the recognition performance, depending on the input device. For example, in the case of mid-air gestures, as in *PathSync*, as long as the gesture performed by users was large enough to be captured by the Kinect sensor, the size of the on-screen target should not matter. Having said that,

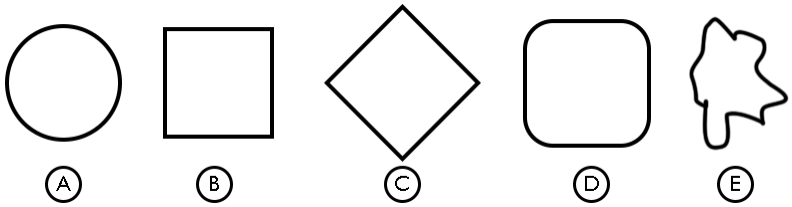

Fig. 17. Trajectory shapes compared in *PathSync* [6]: (A) Circle, (B) Square, (C) Diamond, (D) Rounded Square, (E) Squiggle.

however, we did find that how we displayed the trajectory on the screen influenced how users performed the gestures. For example, we found some correlation between the position where the target was displayed on the screen and the position of the gesture in relation to the user.

In comparison to hand-based techniques, an eye-based technique will more strongly suffer from the effects of the trajectory size. Together with the distance to the display, the trajectory size will determine the visual angle of the eye movement, i.e. how much the eye will move when following the target. If the trajectory is too small, the visual angle range may get too close to the accuracy limitations of the sensor, and the signal recorded by the input device will be more susceptible to noise.

It is important to note that the trajectory size is not a good parameter to disambiguate between two targets. This happens because the correlation calculation normalises the ranges of motion. As a consequence, if the only parameter distinguishing two motions is how large the motion is, further modifications of the algorithms will be necessary. However, the shape can help disambiguate the selection, though if there is no phase difference between two targets (see below), both will still yield high correlations, even if their shape is different.

4.2.3 Phase. From our experience, the difference in *phase* is the best way to distinguish different targets. Varying targets' phase lets different actions available in the interface to be represented by targets moving along a path of the same shape, but slightly offset from one another. This opens the opportunity for the designs of consistent interfaces (e.g. the Window Metro menu we implemented in *PathSync* and the circular menus in *Orbits*). Other characteristics of the interface and of the capabilities of the users in matching the motion will impact how many offset targets can be placed along trajectories of the same shape, but simulations can help inform this decision before testing it with users. For example, we simulated a 60Hz input device following a target on a circular trajectory with different noise levels (representing different abilities of following the target smoothly) with the result shown in Figure 18. We computed the correlation using Algorithm 1 in windows of 30 frames, adding an angular offset as shown in the horizontal axis. The plot lets us discover the boundaries of the potential number of targets for a given threshold and algorithm. In the example, assuming users can match the motion perfectly, if we choose a threshold of 0.8, we see that offsets of 10 degrees on either side would still yield a correlation coefficient above the threshold. This would limit the number of targets to $360/(10 + 10) = 18$. It is interesting to note that when the correlation is smaller than -90 or larger than 90 degrees, the computed coefficient is -1. This is due to the fact that, in these cases, the motion in one of the axes will be in the opposite direction, yielding a correlation coefficient of -1, and because we discard the largest one, the final computation always results in -1.

4.2.4 Direction. The direction of the motion can also increase the number of possible targets along a trajectory of the same shape, but it is necessary to ensure that the motion is distinct in all segments of the trajectory. The problem is

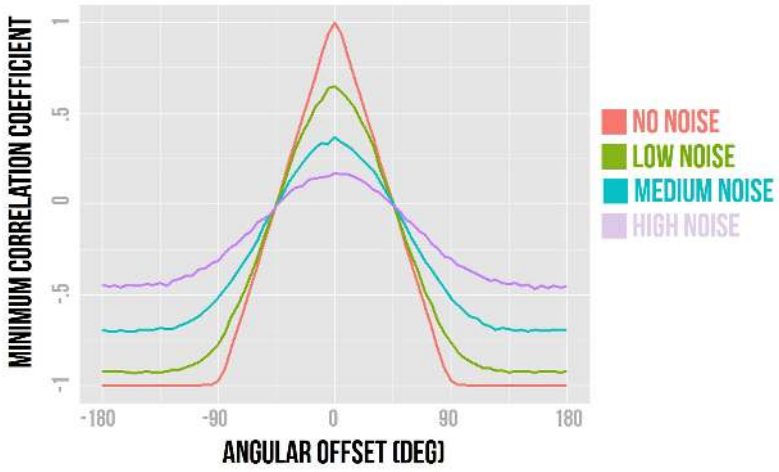

Fig. 18. Simulated correlations between a target moving in a circular trajectory varying the angular offset between the target and the input device for different input noise levels.

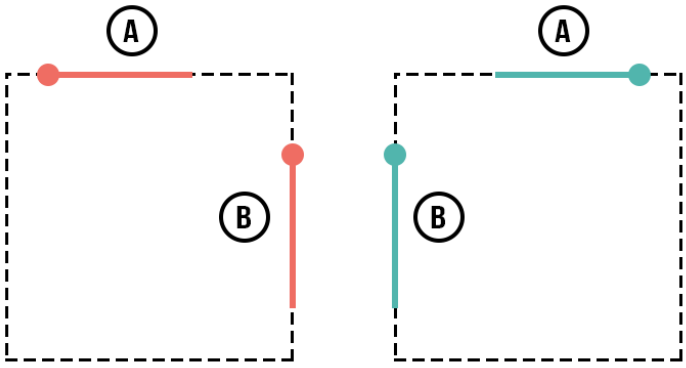

Fig. 19. Alternating high (B) and low (A) correlations with targets in opposite directions.

illustrated in Figure 19, where two targets move around a square in opposite directions. The trail behind the target represents the data in a window over which the correlation is being computed at two instants in time A and B. At time A, the targets move in opposite directions, yielding a correlation of -1 that easily allows us to distinguish them. However, at time B, their motion is indistinguishable until they reach the corner of the square. In this case, increasing the window size to ensure that the corners are being captured or only computing the correlation along the horizontal edges are possible solutions.

4.2.5 Speed. Speed is another motion property that can assist the algorithm in distinguishing targets. However, there are cases where the speed alone does not provide sufficient information. For example, if the frequencies of two given targets are different, they will synchronise at some point—after all, even a broken clock is right twice a day. In

these cases, the size of the window must be large enough the capture sufficient data to allow for them to fall out of synchrony.

4.3 Interface Design

Once the algorithm and the motion of the targets are defined, it is time to integrate them into a cohesive system interface. In this subsection, we focus on aesthetic aspects of motion correlation interfaces and how they affect the interaction.

4.3.1 Input Devices and Modalities. Motion correlation is a generic selection approach that can be used with a variety of input devices and modalities. In this context, Hinckley and Wigdor offer a practical lens to inform this choice [28], which classifies input devices in terms of property sensed, states sensed, number of dimensions, and acquisition time.

Property sensed: Because motion correlation does not require an absolute mapping between the input and output coordinate systens, it offers flexibility for the property sensed by the input device. Whereas the technique works with an established mapping (e.g., Fekete et al. implemented it with a conventional mouse), it also works with only relative motion sensing (e.g., an uncalibrated eye tracker, or an optical flow sensor as used for TraceMatch).

States sensed: Conventional pointing devices often require some means of confirming the selection, such as a mouse click or a touch event. In motion correlation, confirmation is commonly achieved simply by following the motion for a given amount of time. However, being able to sense additional states can be helpful in making the technique more robust to false activation, as the system can be configured to look for matching motion only when certain conditions are met. For example, in *PathSync*, we only correlated the position of users' hands when they were raised.

Number of Dimensions: Because the trajectories we explored in our projects were two-dimensional, suitable input devices for them should sense at least two degrees-of-freedom. Devices that sense more than two can still be used with the appropriate conversion (e.g. the Kinect senses 3D positional data, which can be projected onto the 2D coordinate system of the screen). However, the same principle could be used with different degrees of freedom. For example, an LED strip could offer a 1D animation that could be matched with devices that sense only one dimension, be it positional (e.g. a slider or knob) or not (e.g. pressure, audio, etc.)

Acquisition Time: This generally represents the time it takes to move the user's hand to the input device [28]. Because motion correlation can be used with different modalities this will strongly depend on the choice of body part to be used—for example, in eye-based interaction, the acquisition time is essentially zero as the eyes are always being tracked. Therefore, this strict definition might take other meanings depending on the modality. For example, in *PathSync*, it could refer to the time it takes to raise the hand before engaging in the motion matching per se. In addition to this time, it is important to consider that especially in interfaces that use periodical movements performed with the hands (e.g. targets orbiting around a circular trajectory), users will spend some time waiting until the moving target is at a salient point where they can synchronise the motion.

4.3.2 Path Preview. A lot of our capacity to capture moving objects relies on predicting their future positions. Therefore, displaying where the targets are going can assist users in selecting them. In the prototypes we examined, we found instances of targets moving with full, partial, and no path preview. Whereas in the physical world, the motion of objects is constrained by laws of Mechanics, targets in a graphical user interface can perform any motion the interface designer wishes, which may confuse our natural ability to foresee where they are going and to capture them. To minimise this problem, the interface can show a preview of the path. In *Orbits* and *PathSync*, we displayed the full path of each target [6, 12]. In our deployement of *PathSync*, however, we observed an interesting consequence of how we

displayed the path: because we chose to show it as a closed shape with a colour fill, the trajectories looked like buttons. This caused users who were unfamiliar with the technique to try to touch them to select them. In later deployments of the system, we replaced it for empty squares with a dashed outline [9].

When there are a large number of targets on the screen, displaying their full path can create unwanted visual clutter. In these cases, one solution is to only display part of it, as Williamson did in his *Brownian Motion* prototype (see Figure 2). Finally, as long as the objects follow a predictable motion, it might be more suitable not to present any path preview. In Vidal et al.'s *Frog Game*, part of the challenge was in predicting where the flies would be, so no feedback on their path was provided [65].

4.3.3 Feedback Locus. An important principle of interaction design is to provide users with feedback about their actions. In motion correlation interfaces, targets are constantly moving around the screen and already drawing attention, raising the question of where and how to display feedback for best effect. In *Orbits*, we explored two loci for the feedback: on the target and at the centre of the orbit. Our *Notification Face Plate* prototype is an example of feedback on the target—when users follow one of the dots, it turns into the icon of the corresponding app. The advantage of this design is that users do not have to shift their attention away from the target, which is crucial for eye-based interfaces, but also important for other modalities. There are cases, however, where it might be more suitable for the feedback locus to be away from the target. In our *Music Player* prototype, multiple targets controlled the same parameter. For example, two targets orbiting a speaker icon increased and decreased the volume. In this case, the functionality of each target was communicated by the direction of the motion (clockwise to increase and anti-clockwise to decrease) and the feedback was provided both aurally (through the changes in volume) and visually (through the number of lines around the speaker icon).

4.3.4 Discrete x Continuous Control. While motion correlation interfaces were originally conceived for discrete interaction (i.e. selection), they can also be used for the control of continuous parameters. One way of implementing it is to update the value of such parameters as long as the algorithm determines that the user is following it, as demonstrated in several of our prototypes for volume control. However, it is important to consider the possibility of overshooting the target because of the windowed correlation calculation—after the user stops following the target, it will take some time until the new data fills up the window buffer enough for the correlation to drop under the threshold.

4.3.5 Interface Metaphor. A final aspect of the interface design that we would like to discuss regards the overall aesthetic structure of the interface. The pointing selection principle lends itself well to static interfaces structured around traditional layout principles, such as grids. These layouts can still work with motion correlation interfaces, as we see in how *PathSync* was implemented for the Windows metro interface [6]. However, the dynamic nature of motion correlation interfaces, inspires a different way of thinking about the layout design and its corresponding metaphors. Nature-inspired metaphors implemented in existing prototypes include swimming fish [65], shooting stars [49], and flying insects [65].

5 DISCUSSION

Above, we have shared our understanding and experience of how motion correlation interfaces work, and offered practical guidelines. Here, we take a critical look at the principle by identifying opportunities and challenges for future design and research.

5.1 Opportunities

- **Multi-user interaction with shared displays:** As they are cursorless, motion correlation based techniques are well suited for interaction with shared displays involving multiple users. In particular, this type of interaction is great for interfaces where the informative feedback is useless to an observer outside the interaction loop [68]. We took advantage of this fact in the design of the quiz game *SocialNUIz*: Because all players were making the same shape with their hands (only offset) and there were no cursors on the screen, it was difficult for them to infer other players' answers [6, 9].
- **Non-Pointing Input Devices:** Several modern input devices are not necessarily well suited for pointing, such as accelerometers and gyroscopes, but are great at recognising motion and rhythm. The principle of motion correlation interfaces then offers great potential for cursorless interaction through these devices [68]. For example, though in our *Orbits* and *AmbiGaze* prototypes we employed an infra-red-based eye tracker, our technique also works with one based on electro-oculography (EOG) [54]. This type of device tracks relative movements of the eyes through the electrical signals generated by the ocular muscles, so the gaze point estimation tends to drift with time. Therefore, whereas an EOG tracker would be a poor choice for a pointing interface, it still works well with a motion correlation technique.
- **Feedback Modalities not Suited for Pointing**: Analogously, many output devices are not well suited for pointing either, such as LED strips, audio displays, and mechanical devices [68]. As we showed in *AmbiGaze*, the motion feedback necessary for enabling the interaction can be generated in several ways beyond conventional screens [61]. Further, in the works discussed in this paper, the motion on the interface was always represented by one or more moving particles. An exciting direction for future work is to consider other types of motion, such as flow fields.
- **Incorporating User Performance Models:** As HCI matures as a research field, the more we understand about human capabilities through models of user behaviour. These models can be directly incorporated into the selection algorithm by taking into account delays, lags, transfer functions and any other model of how a user would behave when matching a moving target [67].
- **Cursor-Independence:** In cursor-based selection, users not only need to focus on the target they wish to acquire, but also manage the indirect control of the cursor. Because motion correlation interfaces do not require a cursor, this is no longer a concern. However, though a cursor is no longer necessary, it can still be incorporated in the interaction technique if so desired. For example, Fekete et al.'s *Motion Pointing* was designed for a mouse, so even though the selection was not performed by the cursor, users could still see it on the screen [14]. However, to make a selection, they only had to pay attention to the desired target, not to the cursor.
- **Discoverability and Learnability:** In a critique of gestural interfaces, Norman and Nielsen argue that modern implementations often break several fundamental principles of interaction design, such as discoverability and visibility [47]. When compared to other gestural techniques, correlation-based ones have the advantage that all available commands are shown on the screen [14], making it easy for users to discover their functionality and to learn how to perform the different gestures. In our public deployment of *PathSync*, we found that users that had never used the system before were able to play the quiz game without us having to explain them the technique [6, 9].
- **Interfaces with naturally moving targets**: There are many application domains in which users must select moving targets, such as video games, video recordings of team sports, scientific simulations, air traffic control

systems, interactive TV, surveillance systems, and annotation systems [21, 24, 25, 30]. In these cases, whereas the inherent motion of their targets creates a problem for selection by pointing, they create a promising opportunity for selection by motion correlation. However, the applicability of this principle will depend on how different the motions of the targets are. For example, in recordings of team sports, it often happens that all athletes are running in the same direction, making it difficult for the algorithm to disambiguate the selection.

- **Target Size Independence**: Because in this type of interface, only the properties of the motion affect the selection performance, the properties of the targets themselves matter less than in pointing-based interfaces. This means that selecting a small target is no more difficult than a large one [63]. We leveraged this property to fit multiple targets in the small screen of a smart watch [12], but there are many other contexts where small targets are necessary due to the size of the display or the aesthetics of the interface, including head-mounted displays for peripheral (Google Glass) or mixed-reality interaction (Microsoft's Hololens).

5.2 Challenges & Limitations

- **Matching robustness:** The matching algorithm should be able to accurately trigger the correct target, but also to robustly discard false positives [14]. In the previous section, we described several parameters that determine how robustly the algorithm can determine which target is being selected. These decisions are not independent, as changing one parameter tends to affect others, so empirically determining sweet spots is crucial to good usability.
- **Capabilities of the Input Device and User:** Even a perfect algorithm will fail if the incoming data does not accurately reflect the characteristics of the target's motions. This can happen because of the input device or the user. Noise, lag, insufficient tracking range, mapping distortions, and failure to track the user are a few of the common sources of error caused by the input device. These can be predicted to a certain extent by empirically measuring the device's error characteristics and compensating for those in the algorithm. Problems caused by users' abilities to match the motion are somewhat harder to resolve. These can include lagging behind the target, failing to match the trajectory's shape, fatigue, etc. Thorough user testing before deployment can be very helpful in identifying these issues with the particular target user group.
- **Delimiting Gestures:** modalities such as gaze and hand tracking are in a sense 'always-on'—as long as the user is within the tracking range, the device is capturing data. This creates a challenge for the algorithm to segment the data that relates to the interaction from natural user behaviour not aimed at providing input to the system. Therefore, the algorithm should be able to tell when an input motion refers to an attempt at selection or not [14]. In *PathSync*, we segmented the input stream when users raised and lowered their hand [6]. In *AmbiGaze*, we only displayed the moving targets when the user faced the corresponding ambient device, effectively segmenting the data according to where the user's attention is directed [61]. In *Orbits*, though we did not implement the segmentation explicitly, we suggested ways of doing it depending on the type of eye tracker [12]—with a mobile eye tracker, the system could activate when the watch came into the scene camera's view; with a remote eye tracker mounted on the watch, the system could activate when it detects the user's eyes. However, all of the approaches above do not fully segment the input data. For example, even with a raised hand (or the corresponding delimiters in the eye-based systems), the user could still be thinking about which target to choose and scanning his options. More sophisticated approaches to segmentation (i.e. using a machine learning algorithm to detect when the eyes engage in a smooth pursuit and only run the correlation algorithm in those segments) might yield superior results, which makes them an important direction for future work.

- **Visual clutter:** An inherent problem of any interface that is based on motion is that animated targets can be distracting if used in excess. Not only this can draw users away from the task at hand, but also, in eye-based interfaces, lead to unwanted activations. Though the literature provides many examples of motion correlation interfaces with over 10 simultaneous targets, we recommend keeping this number to the minimum necessary. Visual clutter can also be minimised with clever graphic design. For example, in *Orbits*, our Notification Face Plate application initially shows a small coloured dot for each application. Only after following one of these dots is that the corresponding dot turn into the application icon. Another strategy is to weave the target motion into the interface metaphor: a public display for a theme park can use the movement of the attractions (e.g., a roller coaster) to trigger functionality associated with the attraction itself (see Section 4.3.5).
- **Motion characteristics:** Certain properties of the motion (e.g. complexity of the trajectory shape, speed, etc.) determine how well users will be able to match it, so different targets may be easier or harder to select than others. This must be taken into account in the same way that a button's size and distance to the cursor affect selection performance in pointing interfaces. However, this can also be leveraged as an opportunity for design. For example, safety-critical operations might require the user to follow a more intricate motion shape in order to minimise false activation.
- **Different input devices:** Most input devices are designed for pointing, not for motion matching. Therefore, different input devices and modalities will yield different levels of recognition performance. In our work, we showed how this technique can be used with different eye trackers, a depth camera, and a conventional RGB webcam. An interesting direction for future work is to explore how the technique performs with other devices and modalities, such as finger [53], head [42], and feet movements [60], as well as other types of actions beyond movement, such as audio [10] and pressure [57].
- **Cancellation:** Because of the many challenges outlined above, selection mistakes are expected, particularly when users are new to the technique. Interface designers should therefore allow users to cancel selection or undo other actions. The simplest way to solve the problem is to include 'go back' or 'undo' buttons at every step of the navigation, but due to the dynamic nature of the technique, it might be beneficial to also include some form of real-time feedback as the user performs the action to allow them to correct any mistake before the selection is actually triggered. For example, Williamson et al. added a circle to each target that would grow in radius depending on the probability that the user wanted to select that target [67]. In *PathSync*, our bi-level thresholding method allowed us to highlight the target before the selection is invoked, giving the user some extra time to prevent an unintended activation.

6 CONCLUSION

In this article, we have provided a first comprehensive analysis of motion correlation as an interaction principle for object selection by matching of their movement. While motion correlation had been explored in a variety of works, we have sought to lay a foundation for the design of motion correlation interfaces. We started with theoretical considerations: defining properties of motion correlation and how they contrast other input principles; a basic taxonomy for motion correlation interfaces and related work on synchronous motion; background on human perceptual and motor abilities; and insights from earlier publications on motion correlation. Based on this framing, we reviewed five of our own works, retrospectively focusing on how motion correlation supported novel gaze and gesture interactions in different design contexts. We extracted both principal insights for design of motion correlation interfaces, and practical guidelines for algorithm design, motion feedback, modality choices, and interface architecture.

Motion correlation is a fundamental principle for interaction. It defines a universal and versatile mechanism—allowing for any movement humans can produce to be leveraged for interaction. The same conceptual model of matching motion can underpin interfaces that otherwise can be very diverse. We have shown this for gaze and gestures, and interaction with devices ranging from large displays to small wearables and ambient digital objects. There is an almost boundless design space for the means by which motion can be presented (e.g., also aurally and haptically) and by which it can be sensed (e.g., with embedded sensor instead of optical trackers). Interfaces can be flexibly configured, as input and output devices can be dynamically coupled; all they need to be able to interact is an agreed spectrum of motions that one is capable of displaying and the other capable of sensing.

Early work had demonstrated motion correlation in a setup with a display and a pointing device, where the principle is arguably not at its best, as pointing is more effective for the fine-grained and detailed selections common in desktop interfaces. However, in our work we found motion correlation to be useful and compelling when the user's movement is directly coupled with a devices' motion feedback (and not mediated via an input device the user would manipulate). Motion correlation leverages natural human eye movement and motor abilities, and addresses many of the challenges associated with interfaces that are based on gaze and gestures, for example calibration, discoverability, learnability, and physical effort.

Motion correlation is of appeal in particular for pervasive and spontaneous forms of interaction, as motion is highly discoverable and can help bootstrap interaction. The dynamic coupling of input and output provides flexibility for interfacing of many devices through one common input mechanism, of relevance in our increasingly device-rich environments, and for multiple users to have simultaneous access. There is also great potential for tailored interface designs, for example for interaction with "Internet of Things" devices, as the type of motions, the means for their display, and the sensors for their tracking can all be chosen to fit particular device contexts. While this would lead to an increasing variety of interfaces, they would all be based on the same principle and conceptual model of matching motion.

REFERENCES

[1] Graham R. Barnes. 2011. Ocular pursuit movements. *The Oxford Handbook of Eye Movements* (2011), 115–132. DOI: http://dx.doi.org/10.1093/oxfordhb/9780199539789.001.0001

[2] Peter J. Beek and Arthur Lewbel. 1995. The science of juggling. *Scientific American* 273, 5 (1995), 92–97. DOI: http://dx.doi.org/10.1038/scientificamerican1195-92

[3] 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

[4] Daniel Boland and Roderick Murray-Smith. 2013. Finding My Beat: Personalised Rhythmic Filtering for Mobile Music Interaction. In *Proceedings of the 15th International Conference on Human-computer Interaction with Mobile Devices and Services (MobileHCI '13)*. ACM, New York, NY, USA, 21–30. DOI: http://dx.doi.org/10.1145/2493190.2493220

[5] Thomas Berry Brazelton, Mary Louise Scholl, and John S. Robey. 1966. Visual Responses in the Newborn. *Pediatrics* 37, 2 (1966), 284–290.

[6] 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 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 3415–3427. DOI: http://dx.doi.org/10.1145/2858036.2858284

[7] Tanya L. Chartrand and John A. Bargh. 1999. The chameleon effect: the perception-behavior link and social interaction. *Journal of Personality and Social Psychology* 76, 6 (1999), 893–910. http://dx.doi.org.arugula.cc.columbia.edu:2048/10.1037/0022-3514.76.6.893

[8] 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: http://dx.doi.org/10.1145/2971648.2971714

[9] Travis Cox, Marcus Carter, and Eduardo Velloso. 2016. Public DisPLAY: Social Games on Interactive Public Screens. In *Proceedings of the 28th Australian Conference on Computer-Human Interaction (OzCHI '16)*. ACM, New York, NY, USA, 371–380. DOI: http://dx.doi.org/10.1145/3010915.3010917

[10] Andrew Crossan and Roderick Murray-Smith. 2006. Rhythmic Interaction for Song Filtering on a Mobile Device. In *Haptic and Audio Interaction Design: First International Workshop, HAID 2006, Glasgow, UK, August 31 - September 1, 2006. Proceedings*, David McGookin and Stephen Brewster (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 45–55. DOI: http://dx.doi.org/10.1007/11821731_5

[11] Heiko Drewes and Albrecht Schmidt. 2007. Interacting with the Computer Using Gaze Gestures. In *Human-Computer Interaction – INTERACT 2007: 11th IFIP TC 13 International Conference*. Springer Berlin Heidelberg, 475–488. DOI: http://dx.doi.org/10.1007/978-3-540-74800-7_43

[12] Augusto Esteves, Eduardo Velloso, Andreas Bulling, and Hans Gellersen. 2015. Orbits: Enabling Gaze Interaction in Smart Watches Using Moving Targets. In *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers (UbiComp/ISWC'15 Adjunct)*. ACM, New York, NY, USA, 419–422. DOI: http://dx.doi.org/10.1145/2800835.2800942

[13] 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

[14] 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

[15] Shimin Feng and Roderick Murray-Smith. 2016. Transformations of Gaussian Process Priors for User Matching. *International Journal of Human-Computer Studies* 86, C (Feb. 2016), 32–47. DOI: http://dx.doi.org/10.1016/j.ijhcs.2015.09.001

[16] James D. Foley, Victor L. Wallace, and Peggy Chan. 1984. The human factors of computer graphics interaction techniques. *IEEE Computer Graphics and Applications* 4, 11 (Nov 1984), 13–48. DOI: http://dx.doi.org/10.1109/MCG.1984.6429355

[17] Jayden Garner, Gavin Wood, Sebastiaan Pijnappel, Martin Murer, and Florian Mueller. 2014. I-dentity: Innominate Movement Representation As Engaging Game Element. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 2181–2190. DOI: http://dx.doi.org/10.1145/2556288.2557257

[18] 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 (CHI '12)*. ACM, New York, NY, USA, 1253–1262. DOI: http://dx.doi.org/10.1145/2207676.2208579

[19] Leon Glass. 2001. Synchronization and rhythmic processes in physiology. *Nature* 410, 6825 (08 03 2001), 277–284. http://dx.doi.org/10.1038/35065745

[20] Yves Guiard. 1993. On Fitts' and Hooke's laws: simple harmonic movement in upper-limb cyclical aiming. *Acta Psychologica* 82, 1-3 (1993), 139–159.

[21] Tyler J. Gunn, Pourang Irani, and John Anderson. 2009. An Evaluation of Techniques for Selecting Moving Targets. In *CHI '09 Extended Abstracts on Human Factors in Computing Systems (CHI EA '09)*. ACM, New York, NY, USA, 3329–3334. DOI: http://dx.doi.org/10.1145/1520340.1520481

[22] Reyne Haines. 2010. *Vintage Watches* (1 ed.). Krause Publications.

[23] Hermann Haken, J. A. Scott Kelso, and Herbert Bunz. 1985. A theoretical model of phase transitions in human hand movements. *Biological Cybernetics* 51, 5 (1985), 347–356. DOI: http://dx.doi.org/10.1007/BF00336922

[24] Khalad Hasan, Tovi Grossman, and Pourang Irani. 2011. Comet and Target Ghost: Techniques for Selecting Moving Targets. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 839–848. DOI: http://dx.doi.org/10.1145/1978942.1979065

[25] Jutta Hild, Dennis Gill, and Jürgen Beyerer. 2014. Comparing Mouse and MAGIC Pointing for Moving Target Acquisition. In *Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '14)*. ACM, New York, NY, USA, 131–134. DOI: http://dx.doi.org/10.1145/2578153.2578172

[26] 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

[27] Ken Hinckley, Gonzalo Ramos, Francois Guimbretiere, Patrick Baudisch, and Marc Smith. 2004. Stitching: Pen Gestures That Span Multiple Displays. In *Proceedings of the Working Conference on Advanced Visual Interfaces (AVI '04)*. ACM, New York, NY, USA, 23–31. DOI: http://dx.doi.org/10.1145/989863.989866

[28] Ken Hinckley and Daniel Wigdor. 2012. Input Technologies and Techniques. In *The Human-Computer Interaction Handbook ÃćâĆňàĂIJ Fundamentals, Evolving Technologies and Emerging Applications, Third Edition*, J. Jacko (Ed.). Taylor & Francis. DOI: http://dx.doi.org/10.1201/b11963-9

[29] Lars Erik Holmquist, Friedemann Mattern, Bernt Schiele, Petteri Alahuhta, Michael Beigl5, and Hans-W. 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*. Springer Berlin Heidelberg, 116–122. DOI: http://dx.doi.org/10.1007/3-540-45427-6_10

[30] Michael Victor Ilich. 2009. *Moving target selection in interactive video.* Ph.D. Dissertation. The University of British Columbia, Vancouver.

[31] Robert J. K. 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: Empowering people (CHI '90)*. ACM, 11–18. DOI: http://dx.doi.org/10.1145/97243.97246

[32] Brett R. Jones, Hrvoje Benko, Eyal Ofek, and Andrew D. Wilson. 2013. IllumiRoom: Peripheral Projected Illusions for Interactive Experiences. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 869–878. DOI: http://dx.doi.org/10.1145/2470654.2466112

[33] Dirk Kerzel, David Souto, and Nathalie E. Ziegler. 2008. Effects of attention shifts to stationary objects during steady-state smooth pursuit eye movements. *Vision Research* 48, 7 (2008), 958 – 969. DOI: http://dx.doi.org/10.1016/j.visres.2008.01.015

[34] Tim Kindberg, Kan Zhang, and Seung Hyun Im. 2005. *Evidently secure device associations.* Technical Report HPL-2005-40. HP Laboratories Bristol.

[35] Rick Kjeldsen. 2001. Head gestures for computer control. In *Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems, 2001. Proceedings. IEEE ICCV Workshop on*. 61–67. DOI: http://dx.doi.org/10.1109/RATFG.2001.938911

[36] Alfred Kuhn and William T. Powers. 1975. *Behavior: The Control of Perception*. Vol. 4. SAGE Publications. 306 pages. DOI: http://dx.doi.org/10.2307/2063243

[37] 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 (11 2011), 1–8. DOI: http://dx.doi.org/10.1371/journal.pone.0026859

[38] Kenneth R. Leslie, Scott H. Johnson-Frey, and Scott T. Grafton. 2004. Functional imaging of face and hand imitation: towards a motor theory of empathy. *NeuroImage* 21, 2 (2004), 601 – 607. DOI: http://dx.doi.org/10.1016/j.neuroimage.2003.09.038

[39] Jonathan Lester, Blake Hannaford, and Gaetano Borriello. 2004. "Are You with Me?" – Using Accelerometers to Determine If Two Devices Are Carried by the Same Person. In *Proc. of the Second International Conference on Pervasive Computing (Pervasive'04)*. Springer Berlin Heidelberg, 33–50. DOI: http://dx.doi.org/10.1007/978-3-540-24646-6_3

[40] Baihua Li, Mark Maxwell, Daniel Leightley, Angela Lindsay, Wendy Johnson, and Andrew Ruck. 2014. *Development of Exergame-based Virtual Trainer for Physical Therapy using Kinect*. Springer Fachmedien Wiesbaden, 79–88. DOI: http://dx.doi.org/10.1007/978-3-658-07141-7_11

[41] 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 (CHI '10)*. ACM, New York, NY, USA, 2615–2624. DOI: http://dx.doi.org/10.1145/1753326.1753724

[42] 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

[43] Sébastien Maury, Sylvie Athénes, and Stéphane Chatty. 1999. Rhythmic Menus: Toward Interaction Based on Rhythm. In *CHI '99 Extended Abstracts on Human Factors in Computing Systems (CHI EA '99)*. ACM, New York, NY, USA, 254–255. DOI: http://dx.doi.org/10.1145/632716.632873

[44] Rene Mayrhofer and Hans Gellersen. 2007. Shake Well Before Use: Authentication Based on Accelerometer Data. In *Proceedings of the 5th International Conference on Pervasive Computing (PERVASIVE'07)*. Springer-Verlag, Berlin, Heidelberg, 144–161. DOI: http://dx.doi.org/10.1007/978-3-540-72037-9_9

[45] Rene Mayrhofer and Hans Gellersen. 2009. Shake Well Before Use: Intuitive and Secure Pairing of Mobile Devices. *IEEE Transactions on Mobile Computing* 8, 6 (June 2009), 792–806. DOI: http://dx.doi.org/10.1109/TMC.2009.51

[46] 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 (MobileHCI '12)*. ACM, New York, NY, USA, 147–150. DOI: http://dx.doi.org/10.1145/2371574.2371598

[47] 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

[48] Shwetak N. Patel, Jeffrey S. Pierce, and Gregory D. Abowd. 2004. A Gesture-based Authentication Scheme for Untrusted Public Terminals. In *Proc. of ACM Symposium on User Interface Software & Technology (UIST '04)*. 157–160. DOI: http://dx.doi.org/10.1145/1029632.1029658

[49] Ken Pfeuffer, Mélodie 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 (UIST '13)*. ACM, New York, NY, USA, 261–270. DOI: http://dx.doi.org/10.1145/2501988.2501998

[50] Giacomo Rizzolatti, Luciano Fadiga, Léonardo Fogassi, and Vittorio Gallese. 1999. Resonance behaviors and mirror neurons. *Archives italiennes de biologie* 137, 2 (1999), 85–100.

[51] Mahsan Rofouei, Andrew Wilson, A.J. 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 (CHI '12)*. ACM, New York, NY, USA, 1915–1918. DOI: http://dx.doi.org/10.1145/2207676.2208332

[52] Dominik Schmidt, Julian Seifert, Enrico Rukzio, and Hans Gellersen. 2012. 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

[53] Toby Sharp, Cem Keskin, Duncan Robertson, Jonathan Taylor, Jamie Shotton, David Kim, Christoph Rhemann, Ido Leichter, Alon Vinnikov, Yichen Wei, Daniel Freedman, Pushmeet Kohli, Eyal Krupka, Andrew Fitzgibbon, and Shahram Izadi. 2015. Accurate, Robust, and Flexible Real-time Hand Tracking. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 3633–3642. DOI: http://dx.doi.org/10.1145/2702123.2702179

[54] Junichi Shimizu, Juyoung Lee, Murtaza Dhuliawala, Andreas Bulling, Thad Starner, Woontack Woo, and Kai Kunze. 2016. Solar System: Smooth Pursuit Interactions Using EOG Glasses. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct (UbiComp '16)*. ACM, New York, NY, USA, 369–372. DOI: http://dx.doi.org/10.1145/2968219.2971376

[55] Richard C. Simpson and Heidi H. Koester. 1999. Adaptive one-switch row-column scanning. *IEEE Transactions on Rehabilitation Engineering* 7, 4 (Dec 1999), 464–473. DOI: http://dx.doi.org/10.1109/86.808950

[56] Steven H Strogatz and Ian Stewart. 1993. Coupled oscillators and biological synchronization. *Scientific American* 269, 6 (1993), 102–109. DOI: http://dx.doi.org/10.1038/scientificamerican1293-102

[57] Faisal Taher, Jason Alexander, John Hardy, and Eduardo Velloso. 2014. An Empirical Characterization of Touch-Gesture Input-Force on Mobile Devices. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces (ITS '14)*. ACM, New York, NY, USA, 195–204. DOI: http://dx.doi.org/10.1145/2669485.2669515

[58] 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. In *Proc. of the IFIP TC13 International Confernece on Human-Computer Interaction (Interact'13)*. Springer International Publishing, 384–401. DOI: http://dx.doi.org/10.1007/978-3-319-22701-6_29

[59] Eduardo Velloso, Andreas Bulling, and Hans Gellersen. 2013. MotionMA: Motion Modelling and Analysis by Demonstration. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 1309–1318. DOI: http://dx.doi.org/10.1145/2470654.2466171

[60] Eduardo Velloso, Dominik Schmidt, Jason Alexander, Hans Gellersen, and Andreas Bulling. 2015. The Feet in Human–Computer Interaction: A Survey of Foot-Based Interaction. *Comput. Surveys* 48, 2, Article 21 (Sept. 2015), 35 pages. DOI: http://dx.doi.org/10.1145/2816455

[61] Eduardo Velloso, Markus Wirth, Christian Weichel, Augusto Esteves, and Hans Gellersen. 2016. AmbiGaze: Direct Control of Ambient Devices by Gaze. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems (DIS '16)*. ACM, New York, NY, USA, 812–817. DOI: http://dx.doi.org/10.1145/2901790.2901867

[62] Roel Vertegaal, Aadil Mamuji, Changuk Sohn, and Daniel Cheng. 2005. Media Eyepliances: Using Eye Tracking for Remote Control Focus Selection of Appliances. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems (CHI EA '05)*. ACM, New York, NY, USA, 1861–1864. DOI: http://dx.doi.org/10.1145/1056808.1057041

[63] 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

[64] Mélodie Vidal, Andreas Bulling, and Hans Gellersen. 2015. Pursuits: Spontaneous Eye-Based Interaction for Dynamic Interfaces. *GetMobile: Mobile Comp. and Comm.* 18, 4 (Jan. 2015), 8–10. DOI: http://dx.doi.org/10.1145/2721914.2721917

[65] Mélodie Vidal, Ken Pfeuffer, Andreas Bulling, and Hans Gellersen. 2013. Pursuits: Eye-based Interaction with Moving Targets. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems (CHI EA '13)*. ACM, New York, NY, USA, 3147–3150. DOI: http://dx.doi.org/10.1145/2468356.2479632

[66] Claes von Hofsten. 1980. Predictive reaching for moving objects by human infants. *Journal of experimental child psychology* 30, 3 (Dec 1980), 369–82. DOI: http://dx.doi.org/10.1016/0022-0965(80)90043-0

[67] John Williamson. 2006. *Continuous uncertain interaction.* Ph.D. Dissertation. University of Glasgow.

[68] 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

[69] Jacob Otto Wobbrock. 2009. TapSongs: Tapping Rhythm-based Passwords on a Single Binary Sensor. In *Proceedings of the 22Nd Annual ACM Symposium on User Interface Software and Technology (UIST '09)*. ACM, New York, NY, USA, 93–96. DOI: http://dx.doi.org/10.1145/1622176.1622194