Enabling Techniques to support Reliable Smartphone-Based Motion Gesture Interaction

by

Ankit Kamal

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2014

AUTHOR'S DECLARATION

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

When using motion gestures - 3D movements of a mobile phone - as an input modality, one significant challenge is how to teach end users the movement parameters necessary to successfully issue a command. Is a simple video or image depicting movement of a smartphone sufficient? Or do we need three-dimensional depictions of movement on external screens to train users? In this thesis, we explore mechanisms to teach end users motion gestures and analyze the user's perceived reliability of motion gesture recognition. Regarding teaching motion gestures, two factors were examined. The first factor is how to represent motion gestures: as icons that describe movement, video that depicts movement using the smartphone screen, or a Kinect-based teaching mechanism that captures and depicts the gesture on an external display in three-dimensional space. The second factor explored is recognizer feedback, i.e. a simple representation of the proximity of a motion gesture to the desired motion gesture based on a distance metric extracted from the recognizer. Our results show that, by combining video with recognizer feedback, participants master motion gestures equally quickly as end users that learn using a Kinect. These results demonstrate the viability of training end users to perform motion gestures using only the smartphone display. Regarding user's perceived reliability of the gesture recognizer, the effects of bi-level thresholding on the workload and acceptance of end-users were examined. Bi-level thresholding is a motion gesture recognition technique that mediates between false positives, and false negatives by using two threshold levels: a tighter threshold that limits false positives and recognition errors and a looser threshold that prevents repeated errors (false negatives) by analyzing movements in sequence. By holding recognition rates constant but adjusting for fixed versus bi-level thresholding, we show that systems using bi-level thresholding result in significantly lower workload scores on the NASA-TLX. Overall, these results argue for the viability of bi-level thresholding as an effective technique for balancing between different types of recognizer errors.

Acknowledgements

I would like to give a special thanks to the most awesome supervisor, Prof. Edward Lank for always guiding me in the right direction, helping me and being so friendly to me. A thanks to the HCI lab members too for helping make this happen. Thank you Yi, Valerie, Jeff, Krzysztof, Corona, Ming, Adam, Dan, Mike and especially Keiko, my project partner, who helped me so much in part of my thesis work. I would also like to thank the participants in all our studies for helping in our research. And at last, my thanks to my parents for their unceasing encouragement and support.

Dedication

This is dedicated to my parents, my brother and my best friends - Mary and Zeeshan.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Overview

Hand motion—pointing, gesturing, grasping, shaking, tapping—is a rich channel of communication. We point and gesture while we talk; we grasp tools to extend our capabilities; we grasp, rotate, and shake items to explore them. Surface gestures on a smartphone like swipe, tap and pinch to zoom are the common mode of input in our time. However sometimes, surface gestures can be distracting in certain scenarios like walking or climbing stairs. This could in fact prove to be dangerous. We sometimes need a more eyes-free input modality in such scenarios where the touchscreen is not readily available. This is where motion gestures come into the picture. Inspired by these everyday movements to extend conversation, researchers [29, 30, 36, 37, 42] have begun to explore motion gestures, i.e. deliberate movements to issue commands to a device, as an input modality. Motion gestures have been applied to, for example, large-screen displays, desktop computers, and smartphones.

We were particularly interested in motion gestures as an input modality for modern smartphones. The reasons for this are twofold. First, modern smartphones contain an evolving set of sensors for recognizing movement of the phone, including accelerometers, gyroscopes and cameras, so the technology already exists to support motion gesture input. Second, using a motion gesture provides many attendant benefits, including an expanded input space and the ability to issue commands eyes-free without using the touch screen by leveraging proprioception [30]. In this thesis, we – **(1) explore mechanisms to teach end users motion gestures and (2) analyze the user's perceived reliability of motion gesture recognition**, as presented in the following sections.

1.2 Teaching Motion Gestures

One of the most significant barriers to widespread adoption of motion gesture input involves teaching end-users to perform motion gestures. Motion gestures are not self-revealing; end-users need to be taught the set of motion gestures supported by a smartphone device. As well, for each of these gestures, end-users need to understand exactly how to perform the gestures to ensure maximum recognition accuracy. Constraints on movement include the shape of the movement of the motion gesture (its three-dimensional path in space) and the kinematics of the motion gesture (the tolerances for fast or slow

motion gestures). In order to properly convey this information to the user, some kind of scaffolding is needed.

The process of instructing and correcting the actions of a learner is typically called scaffolding. Scaffolding involves both a depiction of the desired activity and assessment to correct inaccuracies. Many questions arise when considering how to depict motion gestures. Can we simply show icons of motion gestures that depict movement (see Ruiz et al. [37]). Do we instead need to show a brief video of movement on the smartphone display? Or do we require an external display to see movement in larger scale? Alongside techniques for depicting the form of a motion gesture, recognizing a motion gesture involves contrasting the gesture performed with some desired template, i.e. providing some form of feedback that guides a user more quickly to the correct action. We wish to also understand how feedback can be used to help a user converge to the ideal motion gesture more quickly.

1.2.1 Our Contribution

We contrast three techniques for teaching motion gestures: icons, smartphone videos, or Kinect plus videos on external displays. For each of these techniques, we also study the effect that simple recognizer feedback – more specifically a visualization of the distance between a desired template and the actual input movement of the user – on the ability of end-users to accurately perform motion gestures.

We show that Kinect-based instruction, where the movement is displayed as a 3D wireframe and participant movement is captured and replicated for direct contrast, teaches motion gestures very quickly for a group of participants. As well, for Kinect-based feedback, the presence of a visualization of recognizer distance had limited additional effect on accuracy, primarily because the Kinect's contrasting of input motion from template was sufficient. However, we also show that, while video on a smartphone screen is worse than Kinect, video on a smartphone screen plus a simple visualization of recognizer feedback causes the smartphone video condition to converge to the performance of Kinect-based instruction.

The significance of these results lies in the training of end-users on motion gestures as input to smartphones. Before embarking on this research, we were unsure whether it was possible to train end users effectively on motion gestures without someone present to demonstrate the motion gestures, or without external hardware (e.g. a Kinect) to allow users to master the kinematics of the motion gesture commands. Given our results, it now seems plausible to construct a teaching aid for motion gestures that uses only the smartphone display.

1.3 User's perceived reliability of a motion gesture recognizer

Research on smartphone motion gestures is primarily motivated by a desire to make motion gesture more desirble and reliable as an input modality. To improve motion gesture input, researchers have characterized what constitutes a "natural" gesture [36], trained users to perform gestures more accurately [19], or improved the precision and recall of recognizers that seek to interpret motion gesture input to smartphones [29]. However, for reliable recognition of these "natural" gestures, some form of safety net is needed in order to distinguish them from everyday noise.

Our work focuses specifically on the goal of improving the perceived reliability of motion gesture recognizers. Designing highly accurate motion gesture recognizers is challenging because of the similarity between a motion gesture and everyday device movement. In their work on eliciting motion gesture sets, Ruiz et al. [37] note that the motion gestures that were elicited from their participants have low kinematic impulse, and tend to represent short duration movement with one or two degrees of freedom. Figure 2.1 depicts these sets. As a result, it becomes difficult to distinguish motion gesture input from everyday device movement and to distinguish one motion gesture from another. The typical approach to address this is to tighten thresholds for recognition (i.e. criterion values) such that end-users must be more precise when performing an input action, reducing the likelihood of false positives and recognition errors. Unfortunately, tighter thresholds make it more difficult for end-users to perform input with sufficient precision to exceed the threshold, increasing the number of errors of omission made by a system, i.e. the false negative rate.

In recent work, bi-level thresholding has been proposed as a recognition technique to mediate between the false positive and false negative rates [29]. In bi-level thresholding, the system uses two thresholds, a tighter threshold that is selected to prevent false positives and recognition errors, and a looser threshold that prevents repeated false negatives. If a gesture is sufficiently precise to meet the tighter threshold, it is recognized. If, instead, it does not meet the tight threshold but meets the looser threshold, the system moves to a primed state. From this primed state, if a second gesture exceeds the looser threshold, the gesture is recognized. In other words, two near-misses are equivalent to one successful gesture. In earlier work, Negulescu et al. analyzed the effect that bi-level thresholding has on overall recognition, and found that bi-level thresholding permitted near perfect recognition within two attempts with no false positives for a majority of users (95.3% success rate overall with one outlier). As well, 74% of gestures in their study were observed to have a first-instance loose threshold, followed by a tight threshold. Typical errors (4.7%) were a result of users (very infrequently) missing the loose threshold on their first attempt and needing three attempts to successfully perform the gesture [36].

While bi-level thresholding seems to hold promise as a means for enhancing the reliability of recognition, we remain unsure of users reaction to bi-level thresholding. In particular, consider two systems with identical recognition rates. Does bi-level thresholding actually provide an advantage over a similarly accurate recognition system with a single threshold? If given a choice, should developers expend more effort improving first-instance recognition, or is engineering against repeated failures sufficient? At what recognition rate – 70%, 80% – does the effect of bi-level thresholding become less significant?

1.3.1 Our contribution

With these questions in mind, we perform a deeper analysis of the cognitive effects of bi-level thresholding. We implement a tunable recognition system that controls error rate and analyze the effects of bi-level thresholding using a set of dependent measures: walking speed while performing motion gestures; NASA TLX workload scores; and physiological measures (GSR/BVP). We show, in particular, a statistically significant TLX workload difference between single-threshold and bi-level threshold systems when recognition rate is identical across both systems. Overall, these results provide additional evidence of the benefits of bi-level thresholding as a strategy for the recognition of gestural input in the presence of everyday movement. This work was done in collaboration with Keiko Katsuragawa, our lab member.

The remainder of this thesis is organized as follows. Chapter 2 highlights the related work in motion gestures - techniques for teaching gestures and recognizing motion gestures. Chapter 3 presents our pilot study and a final study for evaluating techniques to teach motion gestures and finally we discuss our results. In Chapter 4, we have a discussion about teaching motion gestures. Chapter 5 presents a study to analyze the cognitive effects of fixed vs bi-level thresholding in motion gesture recognizers and we discuss our results. Chapter 6 presents a discussion, implications and limitations of the study in chapter 5. We finally conclude our findings in chapter 7.

Chapter 2

Related Work

2.1 Work on Gestural Interaction

Free-space hand gesture interaction (as in the movie *Minority Report*) has been perceived of as a novel, futuristic input technique, despite known problems with fatigue, i.e. gorilla arm. Bolt designed a "put-that-there" system in 1980 that combined pointing with voice commands [7]. Vogel and Balakrishnan [42] explored the design space for freehand gestural interaction for large vertical displays.

Motion gestures are a known, albeit underutilized, technique for controlling smartphones. Hinckley et al. [17] proposed using tilt on mobile devices to allow a user to change screen orientation—a feature now commonly found on many smartphones. In addition to navigation, tilt sensors have also been used for text input [18] and accessing data on virtual shelves around a user [23]. Commercially, the use of a shake motion gesture to shuffle music is one common example of controlling a smartphone or personal music player (e.g. iPod) via a motion gesture. As well, some modern smartphones allow the user to place the smartphone face-down on a desk to mute the ringtone for an incoming phone call. Finally, the Google App for iPhone turns on voice search if the iPhone is brought to your ear.

2.1.1 Previous research on motion gestures

Ruiz et al. [37] created a taxonomy describing the attributes of smartphone motion gestures and their natural mappings onto smartphone commands. They showed that a consensus exists among users on parameters of movement and on mappings of motion gestures onto commands. They also enumerated a user-defined motion-gesture set for smartphone input. Figure 2.1 shows the motion gesture set.

Alongside work on motion gesture input, Ruiz and Li [36] explored how best to discriminate between deliberate motion gestures and everyday movement of a smartphone. They proposed "DoubleFlip", a motion gesture designed as an input delimiter for smartphone motion gestures. The DoubleFlip delimiter is performed by quickly rotating the wrist such that the phone's display is facing away from the user and back to the original position with the display of the phone facing the user. They showed that DoubleFlip is easy to invoke and unlikely to be accidentally invoked by users.

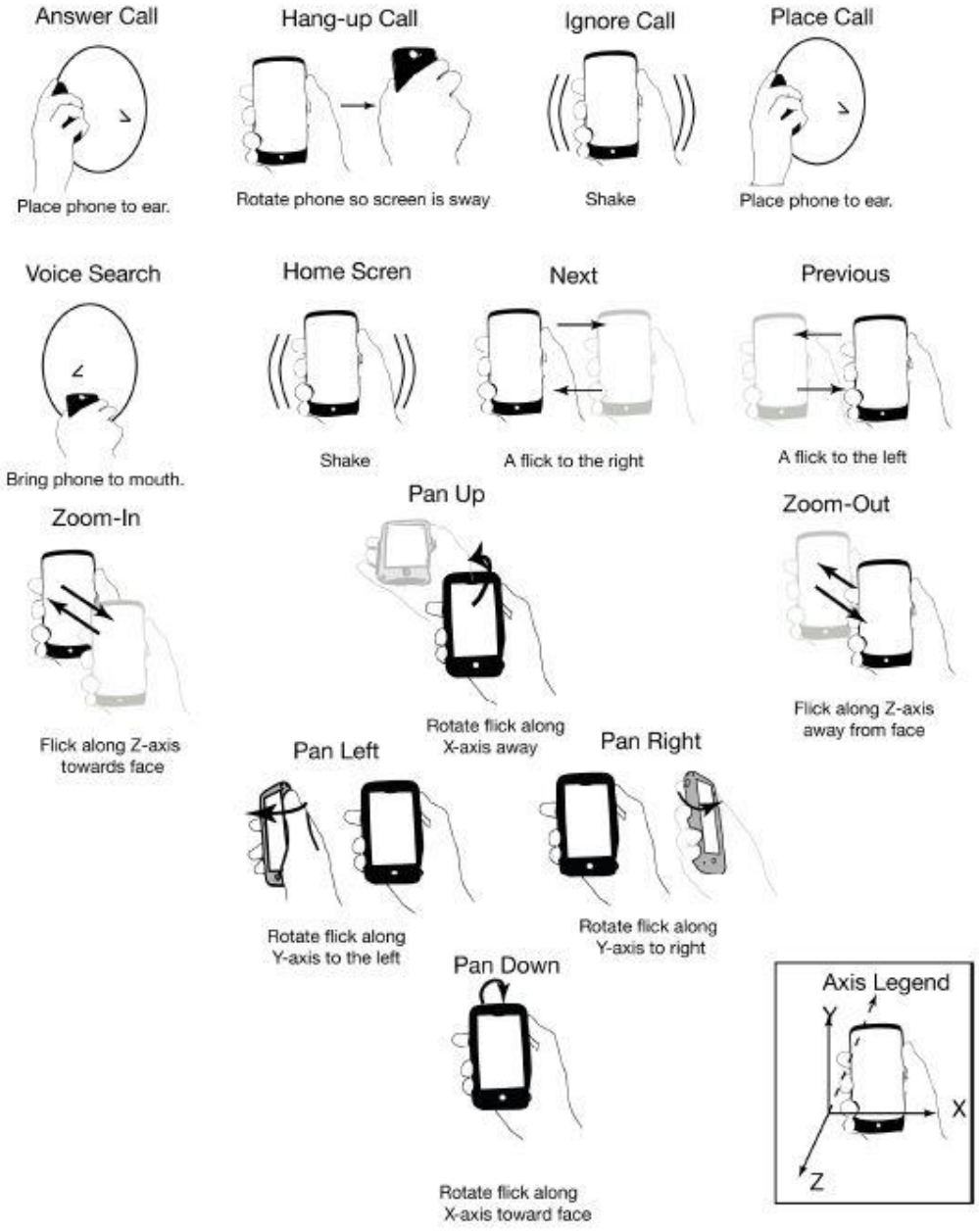


Figure 2. 1 The user-defined motion gesture set. A flick is defined by a quick movement in a particular direction and returning to the starting position.

Negulescu et al. [30] analyzed the relative cognitive cost of taps, surface gestures, and motion gestures for distracted input on smartphone devices. They show that there is no significant difference in reaction

time for motion gestures, taps, or surface gestures on smartphones, and that the use of motion gestures results in participants in a study spending significantly less time looking at the smartphone during walking than taps, even with eyes-free optimized input interfaces.

Negulescu et al. [29] also explored techniques for limiting false positives and false negatives for motion gesture input. They devised a "bi-level threshold" recognizer which helped lower the rate of recognition failures by accepting either a tightly thresholded gesture or two consecutive gestures recognized by a looser-threshold model.

2.1.2 Previous research on teaching surface gestures

Our research in this paper focuses specifically on teaching motion gestures to smartphone users. Significant past work exists in teaching users gestural input languages. Kurtenbach's [22] Marking menus, an extension of pie menus [11], combine feed-forward and feedback to provide a fluent transition between novice and expert use. Marking menus take advantage of novice user's hesitation when they are unsure of a gesture or command. Users flick the pen or mouse in a particular direction in order to indicate a command. After a "press and wait" gesture, a circular feed-forward display appears around the mouse cursor, showing each available command. Highlighting the current selected item during input provides feedback on how a user's input is being interpreted. This approach offers a good compromise between learning and efficient use. Novices often pause to take advantage of the feed-forward display. As they become experts, they move more quickly and no longer needing the feed-forward menu, significantly increase overall performance.

In the same vein as marking menus, Bau et al. [5] designed a dynamic guide called "Octopocus" that combines on-screen feed-forward and feedback to help users learn, execute and remember surface gesture sets. Octopocus continuously updates the state of the recognition algorithm by gradually modifying the thickness of possible gesture paths, based on its 'consumable error rate'. They show that users can better learn, execute and remember gesture sets if one reveals, during input, what is normally an opaque process, the current state of recognition, and represents gestures in a graphical form that shows the optimal path for the remaining alternatives.

One challenge with gesture-based systems is that end-users need to be made aware of the gestures that can be performed to invoke commands. Alongside this awareness, as users are learning the mechanics of gestures, they must also have the opportunity to practice and receive feedback on the gestures they attempt. To satisfy these goals, Bragdon et al. [8] designed a unique training system, GestureBar, which can be incorporated into gesture-based systems for pen-tablet computers. Figure 2.2 shows the

GestureBar interface. GestureBar is, conceptually, a simple scratch pad which allows the user to select a gesture and then attempt the gesture within a region of the display. Feedback depicting the deviation

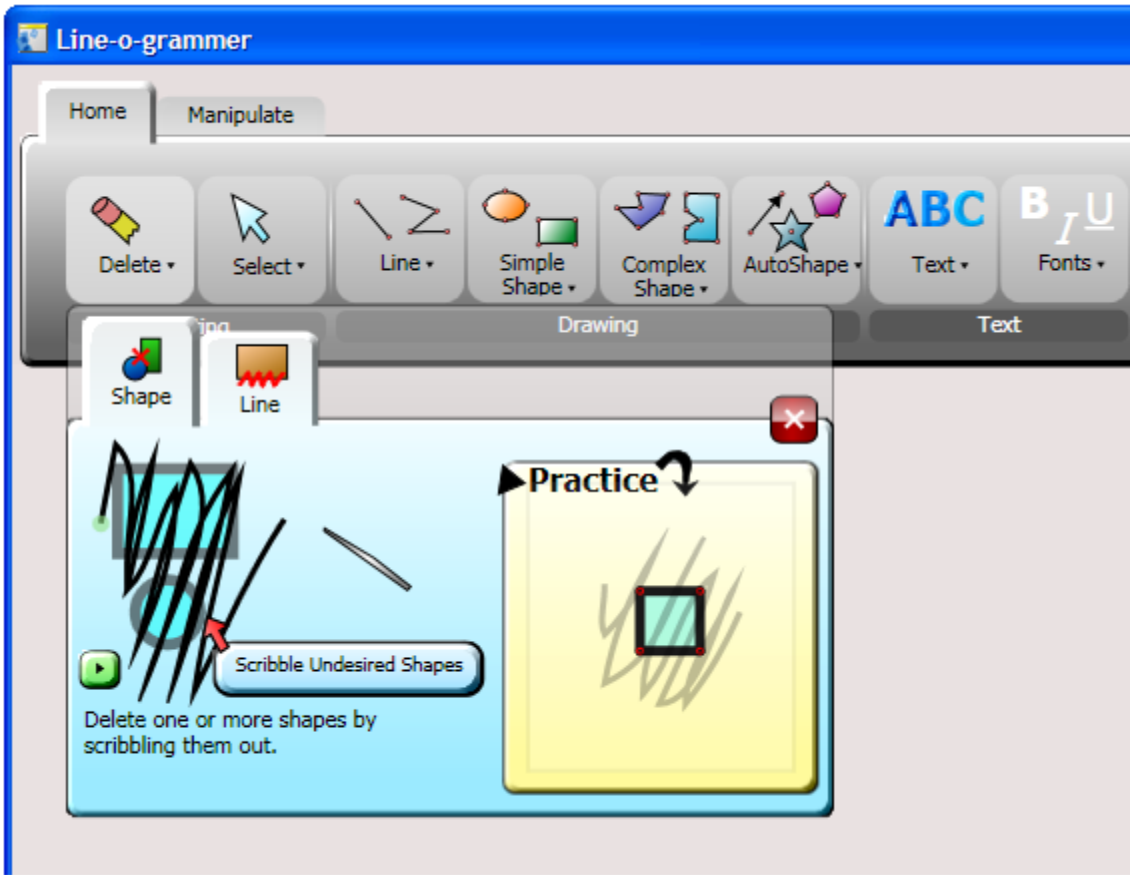


Figure 2. 2 GestureBar after clicking Delete. Commands are not executed when clicked; rather the Gesture Explorer dropdown displays an illustrative animation with detail tips, a replay button, a text description and a practice area.

between desired input and the user's input is displayed so the user can modify and correct any errors in the pen strokes that they draw on the screen. In their research, Bragdon et al. describe the design iterations, the final GestureBar system, and its effectiveness as a training tool based on subjective user feedback.

One of our goals is to adapt aspects of the training systems described above to motion gestures on Smartphones. However, how we communicate motion gestures to end users is somewhat ambiguous. With Marking menus, Octopus, and GestureBar, because users were drawing on a two-dimensional surface, the system could render the two-dimensional shape. Users could start out with an animation of

the movement, then over time simply see the final, complete gesture. However, a smartphone cannot move itself through space. Communicating the relative displacement obviously requires some form of a movie that displays motion relative to the end-user. To the best of our knowledge, no previous research has been done to train people to perform motion gestures.

2.2 Work on Motion Gesture Recognizers

2.2.1 Designing Motion Gesture input

Much of the past research on gestural interaction has been from the perspective of gestures in support of human discourse [46]. However, free-space hand gesture interaction (as in the movie Minority Report) has been perceived of as a novel, futuristic input technique, despite known problems with fatigue, i.e. gorilla arm. Bolt designed a "put-that-there" system in 1980 that combined pointing with voice commands [7]. More recently, Vogel and Balakrishnan [42] explored the design space for freehand gestural interaction for large vertical displays. Multimodal interaction frequently leverages physical gestures alongside other input modalities [27, 43], and toolkits have been developed to simplify the design and deployment of gesture sets [2, 6].

When used as input to smartphones, a motion gesture leverages on board sensors such as the accelerometer and gyroscope to sense changes in orientation. In this vein, early work by Rekimoto [33] demonstrated how mapping motion to tilt can be used for selecting menu items, interacting with scroll bars, panning or zooming around a digital workspace, and performing complex tasks such as 3D object manipulations. Tilt sensors have also been used to navigate through widgets on mobile devices [4, 14, 39]. Modern smartphones use tilt to change screen orientation, an innovation creditied to Hinckley et al. [17]. As well, motion input has also been used for variety of other input tasks, for example, text input [18,32,45], controlling a cursor [43], user verification [25], and accessing data on virtual shelves around a user [23].

Motion gestures as an input modality have been studied by the research community, particularly in work by Jaime Ruiz et al. Ruiz et al. [37] elicited a consensus set of motion gestures for a set of smartphone tasks. In analyzing the consensus set, they noted that their participants tended to specify gestures that had low overall degrees of freedom to the movement, i.e. gestures that represented translation or rotation around a single axis (e.g. double-flip, flick-left, flick-up, etc.). As well, movements tended to exhibit low to moderate intensity in magnitude and change in acceleration, i.e. low kinematic impulse, a result of the propensity of end-users to bias toward movement profiles that

minimize abrupt changes in acceleration [13]. Ruiz and Li [36] also examined everyday smartphone movement and proposed using a specialized motion gesture, the double-flip, as a delimiter for other motion gestures. The use of a delimiter partially mitigates the challenges associated with disciminating between everyday smartphone movement and intentional motion gestures, but at the expense of performing two input actions per command.

The use of motion gestures as an input modality for invoking commands on smartphones has seen some commercial success. The use of a shake motion gesture to shuffle music is one common example of controlling a smartphone or personal music player (e.g. iPod) via a motion gesture. As well, some modern smartphones allow the user to place the smartphone face-down on a desk to mute the ringtone for an incoming phone call. Finally, the Moto X leveraged Ruiz and Li's double-flip gesture [36] to activate the camera.

2.2.2 Recognizing Motion Gestures

Computational recognition of gestural input has a long history. In the domain of surface gestures, Rubine's recognizer [35] is a widely used, single-gesture recognizer that uses a simple set of geometric properties to interpret a gesture. Other variants of spatial recognizers exist, notably variants of elastic matching [40], including the 1$ recognizer [47] and Protractor [12]. Recognition of gestures need not be limited to elastic matching of spatial templates; machine learning algorithms such as Hidden Markov Models [34, 38] have also been used to interpret gestural input.

When interpreting spatial movement of a smartphone, the displacement of the phone is sensed indirectly through sensors including an accelerometer and gyroscope. As a result, input data streams provide data that is not purely spatial. While simple spatial template algorithms such as elastic matchers may be modified to match smartphone sensor data to templates, elastic matchers also assume that the start and end of a template gesture can be accurately identified. This is easy with gestures performed on a display: The gesture is delimited by an explicit pen/finger/mouse down action, and a pen/finger/mouse up action. However, with smartphones which are always in motion and sensing, cleanly delineating start and end can be challenging. When start and end of an input signal cannot be clearly identified, there are algorithms that monitor data streams and recognize templates within those stream. Two common algorithms which have been used to recognize motion gestures on smartphones are dynamic time warping [36, 41] and HMMs [29, 38].

The overall goal of any recognition algorithm is to support high precision and recall [12]. More specifically, we want each gesture to be correctly recognized as that gesture and no other (high

precision) and we want all instances of the gesture to be identified (high recall). When characterizing the performance of recognizers, techniques used to represent precision and recall include confusion matrices and receiver operating characteristic (ROC) curves [12]. The goal of these representations is to help researchers identify correct thresholds, i.e. criterion values, to discriminate between what is a specific gesture and what is not. However, frequently precision and recall are at odds. To prevent confusion between gestures, a tighter threshold can increase precision and avoid misrecognition, but, with a tighter threshold, recall can suffer as certain gestures may not be recognized at all. In the presence of noisy input, these issues are often discussed using terms such as false positives (where a gesture is misrecognized as another or where random noise such as everyday movement is recognized as a gesture) and false negatives (where specific movement is not recognized as a gesture and is, instead, incorrectly labeled as noise or everyday movement) [29].

The most basic strategy to optimize precision and recall is to set appropriate criterion values from ROC curves such that the confusion matrix is optimized [12, 29]. Alongside criterion values, recognizers that learn from end-users, either by manually allowing end-users to specify templates [35] or by providing some form of feedback or learning [16] can be used to refine recognition algorithms on the fly. The selection or criterion values and tailored training of recognizers are complimentary and orthogonal approaches to improving recognition. In this work, we focus specifically on criterion values and, specifically, how one mechanism to mediate between false positives and false negatives -- bi-level thresholding -- affects perceived efficacy of motion gesture interaction.

2.2.3 Bi-Level Threshold recognizer

Bi-level thresholding is a recognition strategy that uses two thresholds, a more restrictive threshold (tight threshold) designed to limit false positives and recognition errors, and a more permissive threshold (relaxed threshold) which prevents repeated false negatives. The recognition strategy can be represented via a 3-state state machine (Figure 2.3).

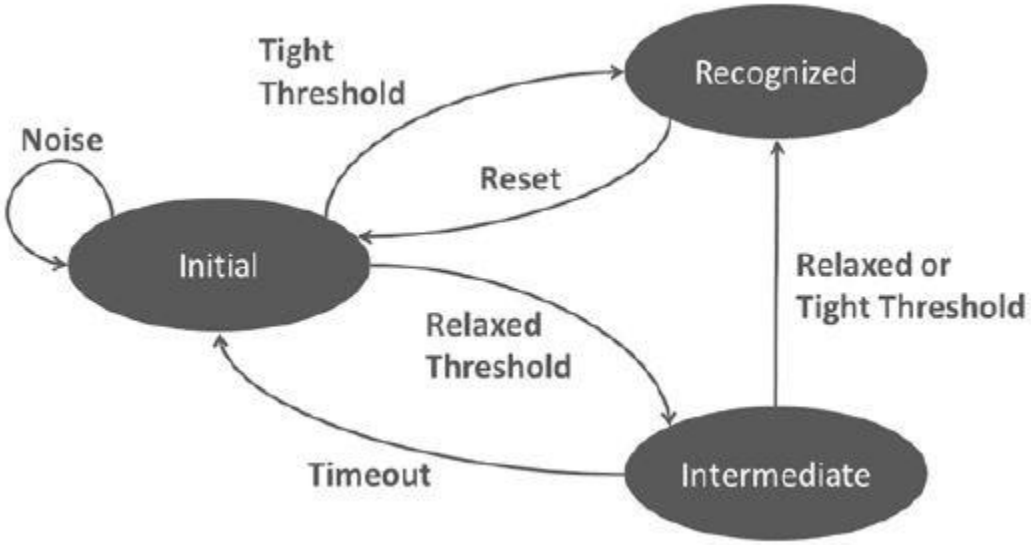


Figure 2. 3 Bi-Level threshold Recognizer

The recognizer begins in an Initial state. Most sensor data received by a smartphone is simply noise, i.e. everyday device movement, and this everyday movement does not cause a state change. From the Initial state, if the recognizer observes a movement which exceeds the tight threshold for a candidate gesture in the template library, the system moves to the Recognized state and the gesture is recognized. If, in contrast, candidate movement exceeds the relaxed-threshold for a template gesture, the system moves to an Intermediate state. In this state, if the system receives either a tight-threshold or relaxed-threshold input for the same gesture, the system moves to the Recognized state and the gesture is recognized. If no such gesture occurs the system moves back to the Initial state after a timeout, typically 3 seconds in our implementation.

The design of bi-level thresholding was motivated by observing participants performing motion gestures while traversing a walking path [29]. It was noted that, if a recognizer misfired, participants would simply try again. However, if subsequent errors occurred, participants would slow down or even stop and would try to perform the gesture in a different fashion. It seemed that a single failure resulted in very minimal disruption to participants, but that repeated failures were significantly more costly. Bi-level thresholding is designed to protect against these repeated failures while preserving a tight initial threshold that prevents false positives. As we noted in the introduction, in early experimental results [29], bi-level thresholding seemed to enhance recognition: 95.3% of gestures were recognized with bi-

level thresholding. Only 35% of gestures would have been captured within two attempts using a single, tight threshold (25% on first attempt and an additional 10% on second attempt).

However, one issue left unaddressed by past work is the usability of bi-level thresholding as a recognition strategy. Good usability isn't just about reliable recognition, but also about how well is the user's perceived reliability of the recognizer. As a result, we evaluate the workload and user acceptance of bi-level threshold as a strategy for recognizing motion gesture input.

Chapter 3

Observational Study to Teach Motion Gestures

3.1 Gesture Recognizer Design

Ruiz et al. [37] note that, when end-users design motion gestures, the gestures they select tend to be simple (non-compound), single-axis movements with low kinematic impulse. As a result, we base our study around four single-axis gestures – *right flick, left flick, flick up towards face* and *flick down away from face*.

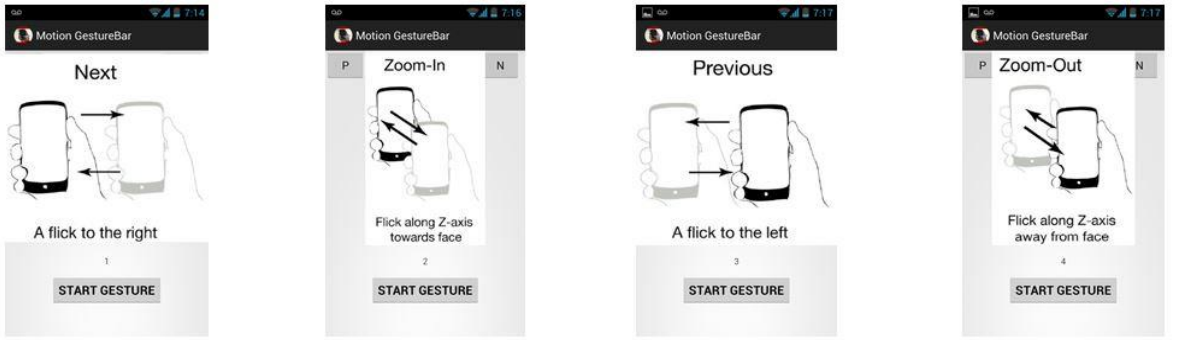
Our four gestures were chosen from the user defined set in Ruiz et al. [37], and we would argue that they represent the simplest set of useful motion gestures for smartphone control. Nominally, the gestures correspond to next, previous, zoom-in and zoom-out gestures respectively. Essentially, we chose the gestures we did because these are the types of gestures – single axis, low kinematic impulse – users specify when we elicit gestures from them [37].

Our recognizer was developed in Java using the Android SDK [1] for use on Nexus S phones with an ARM Cortex A8 1GHz processor and a three-axis accelerometer. Sensor input, i.e. filtered acceleration data, is matched to gesture templates using Dynamic Time Warping (DTW) [28]. DTW is a dynamic programming algorithm that measures the similarity of two time series with temporal dynamics [28] when given a function for calculating the distance between the two time samples. The result is a warp distance that can be used to determine how similar a set is to the reference set. A warp distance of 0 (zero) indicates absolute identical sets. The bigger the distance, the more different the sets are. Our implementation of our gesture recognizer uses a weighted Euclidean distance function for calculating the distance between the quantized time series of acceleration data to the template. As a full discussion of DTW is beyond the scope of this paper, we refer the reader to Wobbrock et al. [47] for more information. The sampling rate of acceleration data was 32 Hz.

One challenge with the gestures we select is that, because they are single-axis and because they have low kinematic impulse, the gestures are virtually indistinguishable from everyday movement of a smartphone. The typical way designers of recognizers address a collision between noise and signal is

via a tight criterion function to discriminate true positives from false positives [29]. The challenge with a tight criterion function is the propensity to cause false negatives. In other words, seeking to avoid accidental activation of a motion gesture, we require greater precision in the performance of a motion gesture. This, in turn, makes it more essential to teach end users the careful kinematics needed to successfully invoke a motion gesture; otherwise, they repeatedly fail to invoke their desired motion gesture.

To simulate this tight criterion function, the DTW templates for each gesture type were created by



(a) Right Flick (b) Flick away from face (c) Left Flick (d) Flick towards face

Figure 3. 1 Iconic representation of the motion gestures

an expert user, specifically one of the authors of the work presented here. The expert performed the correct gesture 20 times. Each gesture was compared to the 19 other gestures using DTW. Then, the average warp distance for the respective gesture was calculated, and the gesture with lowest average warp distance from all other gestures was selected as the gesture template for that particular gesture. This is a common approach found in related work (Kar et al. [20]). In a second step, the selected gesture template was compared to the remaining 19 gestures. The 19 warp distances were then used to calculate the mean, median, minimum, maximum and standard deviation of distances. These values were used to calculate the threshold of the DTW Distance metric within which an input gesture is considered as valid. The threshold was chosen as the median plus the standard deviation. The result of the use of a single expert user is that, to successfully invoke a motion gesture on a smartphone, the end-user must perform the gesture in nearly the same manner as the expert from whom the template was elicited.

In the following sections, we describe our experiments where we explore various types of mechanisms to teach end users motion gestures, examining two factors. The first factor we explore is how to represent motion gestures: as icons that describe movement, video that depicts movement using the smartphone screen, or as video on an external screen. The second factor we explore is feedback, i.e.

a simple representation of the proximity of a motion gesture to the desired motion gesture based either on a distance metric extracted from the recognizer or based upon movement tracked by Kinect.

3.2 Preliminary Study – Depicting Motion Gestures

We performed a preliminary study to compare the performance of two basic representations of motion gestures as teaching methods: icons describing the movement (see Ruiz et al. [37]) and short videos depicting the movement on the smartphone screen. Feedback mechanisms were not explored in this study. The reason for conducting this study was to check if very basic representations of the motion gesture, e.g. icons or simple videos, are sufficient to teach motion gestures.

3.2.1 Recruiting and Participants

We recruited 12 participants (8 male, 4 female, ages 20-35) from the general student body of our institution. We advertised the study widely to get a sample of participants with diverse backgrounds and levels of experience using computers. All participants owned a smartphone. and knew what motion gestures were, but not with respect to movement of the smartphone device. All were familiar with motion gestures pertaining to the Nintendo Wii or Kinect based games, but none were familiar with smartphone-based motion gestures (beyond shake-to-shuffle).

3.2.2 Experimental Design

We used a between-subjects design with the two conditions - teaching via icons and via videos. The reason for choosing a between-subjects design is that if a user masters a gesture using one technique, the evaluation of the other technique becomes invalid. Six participants were asked to perform motion gestures based on the iconic representation of the motion gesture shown and the other six were asked to perform gestures based on the video shown on the phone. The iconic representations that were displayed on the Android device are shown in Figure 3.1. These iconic representations of the motion gestures were taken from the user-defined set created in the work done by Ruiz et al. [37]. The videos of gestures were captured from gestures performed by an expert user, and they depict the gesture used for the correct template in our recognizer from an eyes-view. i.e. as if one was looking at the smartphone while performing the gesture, Figure 3.2.

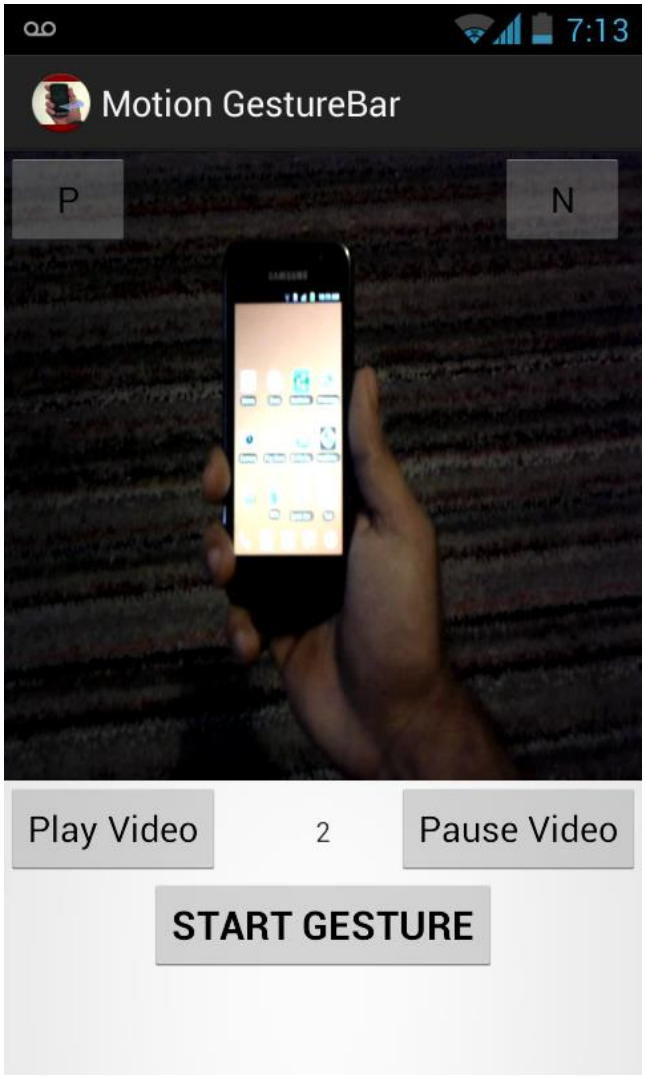


Figure 3. 2 A screen-grab of the video on the phone describing the correct motion gesture

3.2.3 Procedure

Each participant was asked to perform four blocks (corresponding to the four gesture types) of thirty gestures each, i.e., 4 x 30 = 120 gestures. The order of the blocks of gestures to be performed was presented randomly. We did not give any hint to our participants regarding the correct gesture. When our DTW recognizer recognized a correct gesture, a beep sound was generated, indicating the completion of the correct gesture. Participants could refer to the icons or watch the videos as many times as they wanted. Some of them did watch the videos many times. A total of 4 x 30 x 12 = 1440 gestures were performed.

3.2.4 Metrics

We extracted two metrics from our participants:

No. of correct gestures: The number of correct gestures out of the total of 120 performed by each user. This is a measure of performance of the user.

Average converging gesture count: This value is the average number of gestures it took for the participant to converge (or learn) to the correct gesture. Convergence is essentially, the point after which the gesture is performed consistently correctly. In our data, we chose to define it as 80 - 100% success rate, and represents the speed of learning a specific gesture.

3.2.5 Results

Figure 3.3 shows the number of correct gestures (out of 120) performed by all 12 participants for the two conditions – videos and icons. A Student's t–test showed significant differences for the number of correct gestures performed. Participants performed significantly better ($p < .001$) in terms of number of correct gestures performed with videos ($M = 98.5$, $S.D = 1.87$) versus icons ($M = 83$, $S.D = 3.74$). Figure 3.4 shows the average number of gestures (out of 30) over the four kinds of gestures at which the participants converged to the correct gesture. A Student's t–test showed significant differences for the average converging gesture count. Participants performed significantly better ($p < .001$) in terms of average number of gestures to converge to the correct gesture with videos ($M = 5.5$, $S.D = 0.55$) than with icons ($M = 9.5$, $S.D = 1.04$). The primary reasons for not performing a correct gesture were differences in speed and direction (acceleration of the device along a particular axis to be precise). If the DTW distance (which was based on acceleration along a particular axis) was within a specified threshold, then the gesture was considered correct.

Our initial results indicate that, by only showing icons, participants take a significant amount of time to converge to the correct gesture (10 gestures on an average) and perform poorly. Videos perform significantly better (6 gestures on an average) than icons as a teaching method, but scope for improvement exists. In the next section, we describe a set of designs that support feedback on the accuracy of performing motion gestures. We also evaluate our mechanisms for teaching gestures and assessing gesture accuracy.

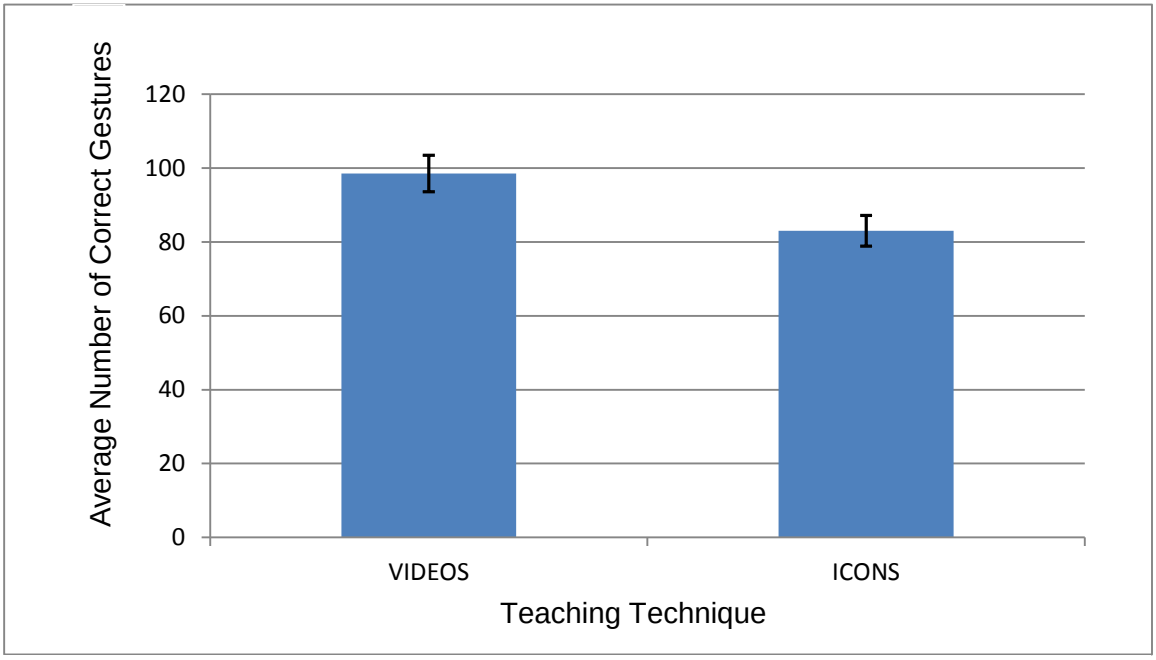


Figure 3. 3 Average number of correct gestures (out of 120) performed by participants. 95% CI shown

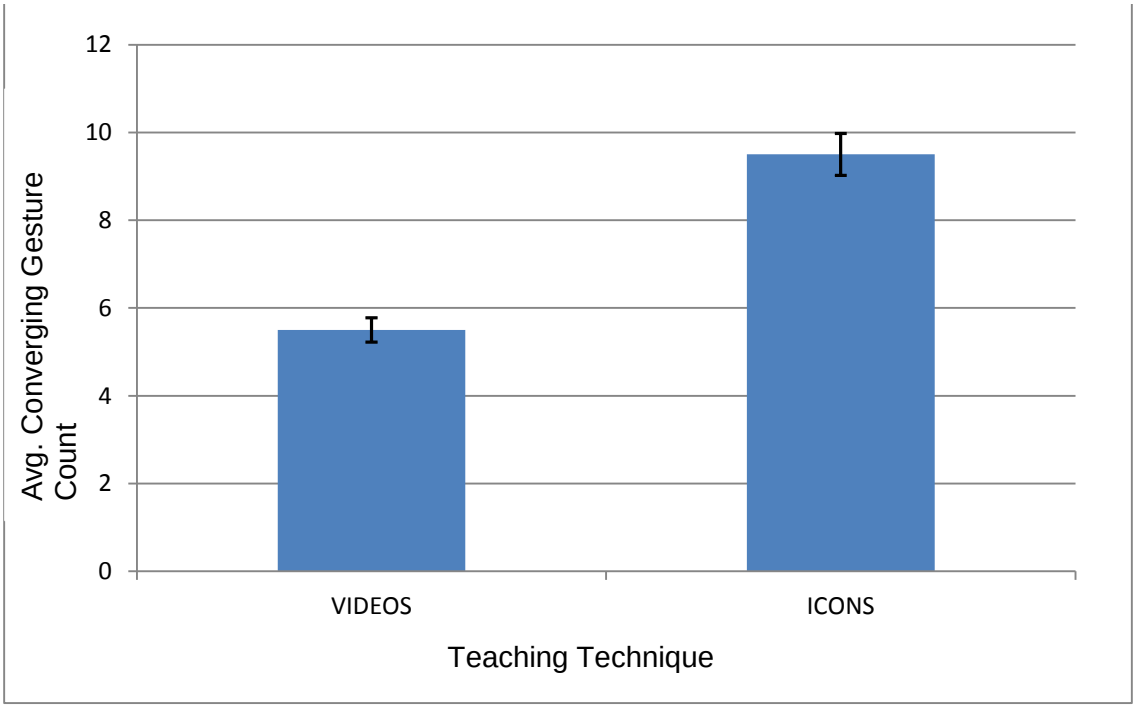


Figure 3. 4 Average gesture count (out of 30) over the four types of gestures at which participants converged to the correct gesture. 95% CI shown

3.3 Evaluation of Techniques to Teach Motion Gestures

Our second user study explores additional representations of motion gestures, specifically examining the presentation of motion gestures on an external screen. We also explore additional feedback mechanisms for depicting motion gestures. These include using the Kinect to provide feedback, and also providing feedback from our recognizer using a distance metric extracted from the DTW algorithm.

3.3.1 Recognizer Feedback design

In their work on bi-level thresholding, Negulescu et al. [29] note that, when users repeatedly fail to perform a motion gesture, they begin to vary the parameters of movement, attempting, essentially, to re-acquire the correct movement parameters needed to perform the motion gesture. We use the term *annealing* to describe this process of exploration.

Feedback that allows end users to assess the accuracy of a gesture exists on a continuum, from simple to more complex forms. The simplest form of feedback is some indication of correct versus incorrect from a recognizer. However, given the annealing process of users who fail to perform motion gestures, the goal of our feedback designs was to guide users to the correct gesture, i.e. to guide this annealing process. We design two feedback mechanisms, one on smartphone and one on an external display.

For feedback on the smartphone, our approach was one of minimal feedback, as in we tried to adopt the simplest feedback we could while still guiding the annealing process. We performed a series of pilot studies to design our recognizer feedback. We began with simple, three-level textual feedback (correct, near, far), a feedback mechanism that corresponds to the children's game Cold, Warm, Hot, where someone hides an object and then guides a child's search. In early pilots, we found that the textual information was difficult to acquire and so was ignored. We also explored simple colors to provide feedback, but this, too, seemed insufficient. As a result, we moved to a numerical scale, based directly upon DTW distance.

Figure 3. 5 A screen shot of the recognizer

Figure 3.5 shows a screen shot of the final design of our recognizer feedback mechanism. An arrow maps to the DTW distance metric of the recognizer. A feedback bar provides some basic context for the position of the arrow. The feedback bar is divided into 3 equal mini bars: Green is sufficiently close to be correct, yellow is within a loosened threshold, and red quite far from the correct template.

When feedback is enabled, after every gesture performed by the user, the recognizer displays the performance feedback bar with the arrow indicating the proximity to the gesture template. A distance of 0, i.e. a perfect gesture, would result in the arrow being positioned at the top of the green region. We continue to provide auditory feedback of recognition, specifically using three distinct sounds depending on which of the three regions the arrow points to. For gestures that pass the threshold for correctness, the bar comes up immediately after the gesture is performed. For gestures that land in the almost correct or incorrect area, the feedback bar is displayed after a pause of 2-3 seconds. This is because any gesture below the threshold cannot be recognized immediately.

We also designed a more complex feedback mechanism that used an external screen and Kinect to provide users with a three dimensional depiction of movement and an ability to directly contrast their movement with the desired template, shown in Figure 3.6. Users were shown a Kinect skeleton performing a motion gesture and a video of the expert performing the gesture. As a user performed the motion gesture, the movement was displayed in an adjacent skeleton. After completing the gesture, they could replay their movement and the correct template simultaneously to identify deviations between their motion and that of the perfect template.

We used the nuiCapture Analyse (Trial Version) software [31] to capture and display the motion data from the Kinect sensor. The Kinect depth sensor provides various motion tracking views. We captured videos of the expert performing the correct gestures along with the corresponding skeletal movement

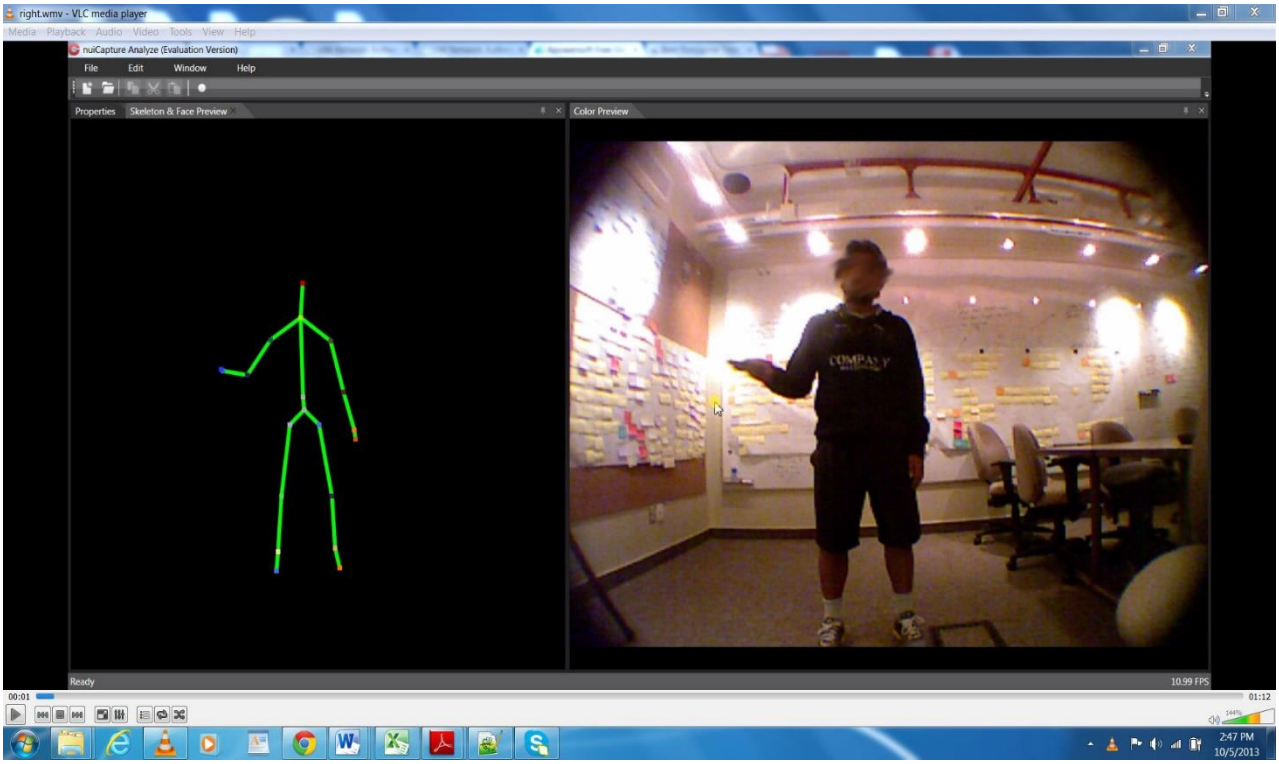


Figure 3. 6 A screen shot of the Kinect-based teaching mechanism that captures and depicts the gestures on an external display in three-dimensional space

as shown in Figure 3.6. On one PC screen, the user could see a large sized captured video of the human (RGB) and skeletal movement of the correct motion as shown in Figure 3.6. On another adjoining screen they could see their own motion- both skeletal and human. Both, the prerecorded video and their own motion was presented as a mirror image so that their movements correspond to the correct side.

We could also record the motion of the participant and play it against the correct motion to compare movements.

The end result is that, for the purposes of evaluating feedback, we provide three different levels of feedback to end users: correct/incorrect (no feedback); numerical scale depicting DTW distance (DTW feedback); and full motion feedback with Kinect (Kinect). Our initial hypothesis was that Kinect feedback would best teach users to perform motion gestures. However, using the Kinect requires external hardware, whereas our other forms of training and feedback can be provided using only the smartphone device. The specific question we ask is how much worse than Kinect other forms of presentation and feedback for teaching motion gestures are.

3.3.2 Recruiting and Participants

We recruited 50 participants (23 male, 27 female, ages 20-35) from the general student body of institution. As in our earlier study, we advertised the study widely to get a sample of participants with diverse backgrounds and levels of experience using computers. All participants owned a smartphone and knew what motion gestures were, but not with respect to movement of the smartphone device. Some of the participants were familiar with some hand gestures above the screen that can be performed on the Samsung Galaxy S4 Android device. All participants were remunerated with a $10 Tim Horton's gift card after the completion of the experiment.

3.3.3 Experimental Design

We again used a between-subjects design for this study. The rationale for choosing a between-subjects design is that, if a user masters a gesture using one technique, the evaluation of the other technique becomes invalid. In this experiment, we evaluate the following 5 motion gesture teaching techniques – icons with DTW feedback, videos, videos with DTW feedback, Kinect, and Kinect with DTW feedback. We did not evaluate icons as a teaching mechanism in this study due to their poor performance in our preliminary study.

3.3.4 Procedure

As in the preliminary study, the gestures that the participants were asked to do were *right flick, left flick, flick towards the face* and *flick away from face*. Participants were required to perform the gesture presented to them 30 times. Thus, each participant was asked to do four blocks (corresponding to the four gesture types) of thirty gestures, i.e., 4 x 30 = 120 gestures. Each gesture block was presented to them randomly. We described the presentation and feedback mechanisms for the desired gesture, but

did not provide any guidance on when or how to use feedback during the experiment. The goal of the participants was to perform as many correct gestures as possible. Given 50 participants in our study, for each of the 5 teaching/feedback mechanisms, we had 10 participants. Thus a total of 4 x 30 x 50 = 6000 gestures were performed, 1200 per feedback mechanism. In the case of the Kinect with DTW feedback, the DTW feedback, was displayed on the smartphone after each gesture. After performing all the gestures, each participant was asked to complete an exit questionnaire, followed by a semi-structured interview. The questionnaire examined the subjective preferences of our participants, and the interview was intended to obtain their opinion on motion gestures in general as an input modality for smartphones.

3.3.5 Metrics

As in our previous study, we capture the following measures:

No. of correct gestures: The number of correct gestures out of the total of 120 performed by each user. This is a measure of performance of the user.

Average converging gesture count: This value is the average number of gestures it took for the participant to converge (or learn) to the correct gesture. Convergence is essentially, the point after which the gesture is performed consistently correctly. In our data, we chose to define it as 80 - 100% success rate, and it measures how quickly users can learn motion gestures.

3.3.6 Results

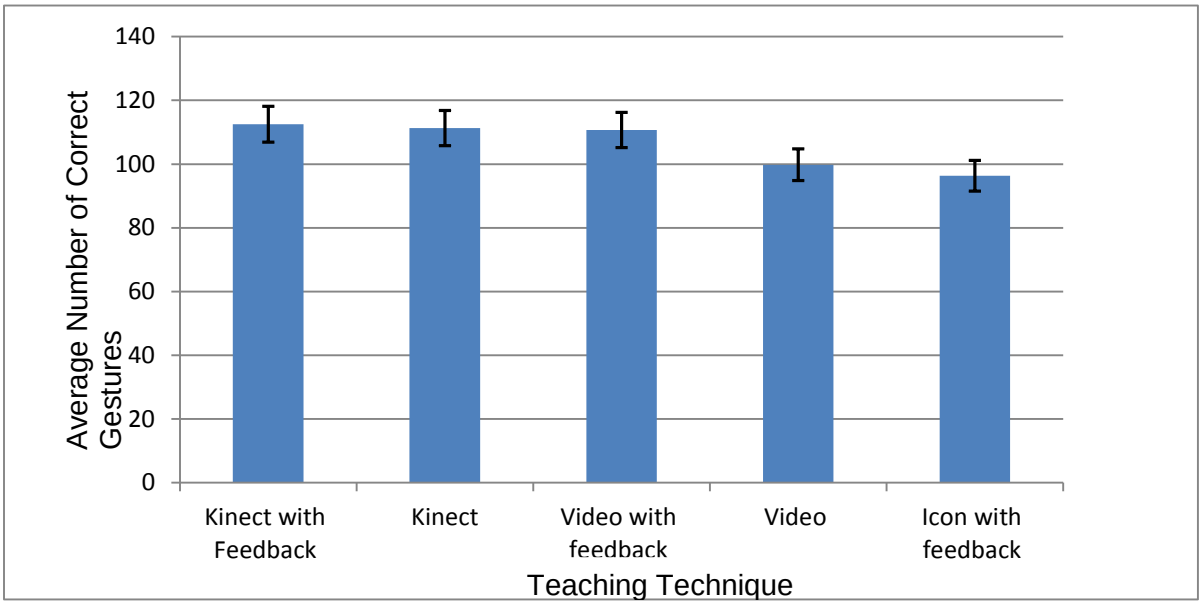


Figure 3. 8 Average number of correct gestures (out of 120) performed by participants. 95% CI shown

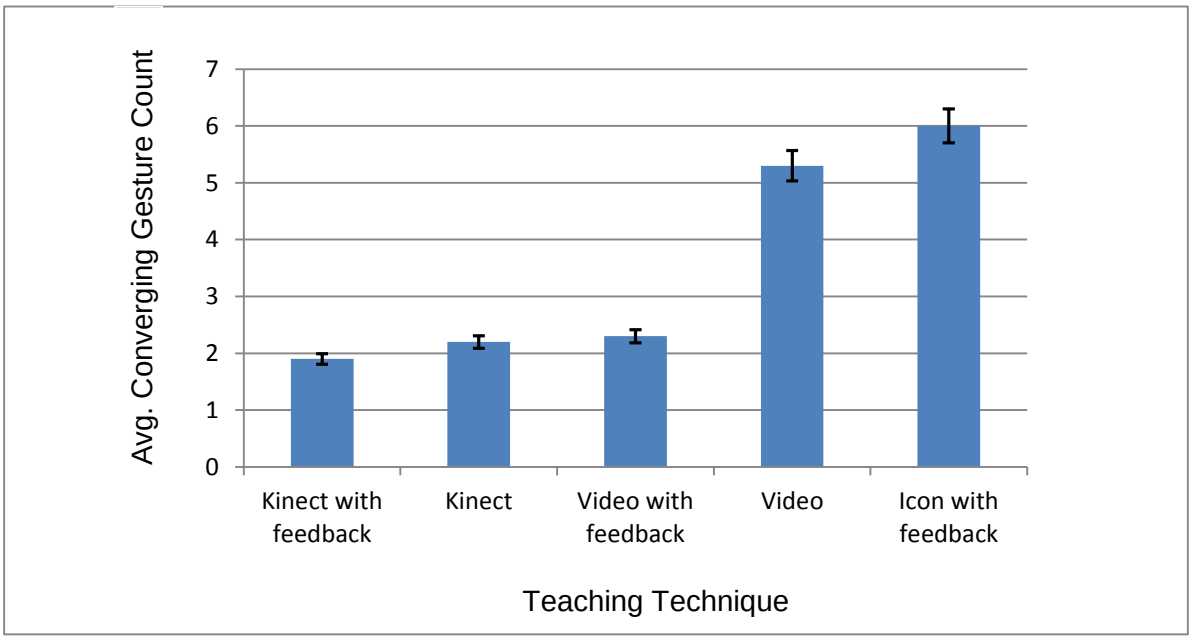


Figure 3. 7 Average gesture count (out of 30) over the four types of gestures at which participants converged to the correct gesture. 95% CI shown

Figure 3.7 shows the number of correct gestures (out of 120) performed by all 50 participants for the 5 teaching mechanisms - icons with the DTW feedback, videos, videos with DTW feedback, Kinect, and Kinect with DTW feedback. A one-way analysis of variance shows that teaching technique had a significant effect on number of correct gestures performed ($F_{4,45} = 43.9$, $p < 0.001$). Post-hoc analysis using Bonferroni correction showed significant differences between Kinect with feedback and videos ($p < 0.001$), Kinect with feedback and icons with feedback ($p < 0.001$), Kinect and videos ($p < 0.001$), Kinect and icons with feedback($p < 0.001$), videos with feedback and icons with feedback($p < 0.001$) and finally between videos with feedback and videos($p < 0.001$). Participants performed significantly better in terms of number of correct gestures with video with feedback (M = 110.7, S.D. = 3.5), Kinect (M = 111.3, S.D. = 3.4) and Kinect with feedback (M=112.5, S.D. = 2.9) than with videos (M = 99.8, S.D. = 3.5) or icons with feedback (M = 96.3, S.D. = 4.2). No significant differences were found in terms of number of correct gestures among Kinect with DTW feedback, Kinect, and videos with DTW feedback teaching mechanisms ($p = 1.00$ for all). This shows that, by combining video with DTW feedback, participants perform almost equally well as end users that learn using a Kinect or Kinect with DTW feedback.

Figure 3.8 shows the average number of gestures (out of 30) over the four kinds of gestures at which the participants converged to the correct gesture. A one-way analysis of variance shows that the teaching technique had a significant effect on the average converging gesture count ($F_{4,45} = 37.9$, $p < 0.001$). Post-hoc analysis using Bonferroni correction showed significant differences between Kinect with feedback and videos ($p < 0.001$), Kinect with feedback and icons with feedback ($p < 0.001$), Kinect and videos ($p < 0.001$), Kinect and icons with feedback ($p < 0.001$), videos with feedback and icons with feedback ($p < 0.001$) and between videos with feedback videos ($p < 0.001$). Participants performed significantly better in terms of average number of gestures to converge to the correct gesture with video with feedback (M = 2.3, S.D. = 1.15), Kinect (M = 2.2, S.D. = 1.03) and Kinect with feedback (M=1.9, S.D. = 0.73) than with videos (M = 5.3, S.D. = 1.2) or icons with feedback (M = 6, S.D. = 1.05). No significant differences were found in terms of the average number of gestures to converge to the correct gesture among Kinect with feedback, Kinect and videos with feedback teaching mechanisms ($p = 1.00$ for all). This shows that, by combining video with DTW feedback, participants perform equally quickly (i.e., just after 2 incorrect gestures) as end users that learn using a Kinect or Kinect with DTW feedback.

These results demonstrate the viability of training end users to perform motion gestures using only the smartphone display. In particular, given some graphical representation of distance from correct

gesture and a video depicting kinematics of movement, participants performed as well and learned as quickly as participants trained using full graphical feedback via the Kinect.

3.3.6.1 Subjective Preferences of exit questionnaire

We further examined the subjective preferences of our participants via an exit questionnaire. Participants were to circle choices on a Likert Scale from -3 to 3. The following questions were asked in the questionnaire:

1. How did you like the motion gesture teaching technique in this experiment? Here, a rating of -3 corresponded to very poor and 3 to very good.
2. Would you like to have motion gestures for device commands along with surface gestures? Here, a rating of -3 corresponded to least preferred and 3 to most preferred.

Figure 3.9 shows the results of the first question. i.e., How much did the participant like the teaching technique. A one-way analysis of variance shows that teaching technique had a significant effect on the average rating of how much the users liked it ($F_{4,45} = 6.142$, $p < 0.001$).

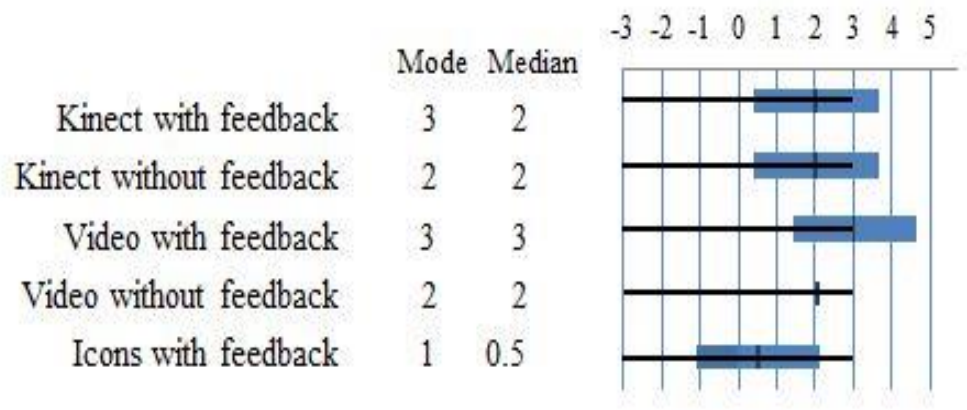


Figure 3. 9 Median Likert rating from -3 to 3 for how much participants liked the teaching technique. The bars show 95% CI for median

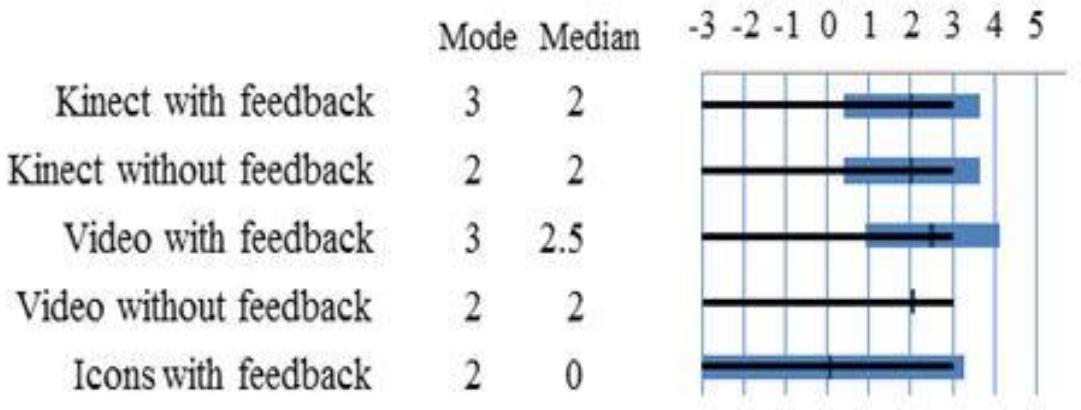


Figure 3. 10 Median Likert rating from -3 to 3 for participant's opinion of motion gestures as an input modality along with surface gestures. The bars show 95% CI for median

Post-hoc analysis using Bonferroni correction showed significant differences between Kinect with feedback and icons with feedback ($p < 0.05$), Kinect and icons with feedback ($p < 0.05$), videos with feedback and icons with feedback ($p < 0.05$) and finally, videos and icons with feedback ($p < 0.05$). Participants gave significantly better ratings for video ($M = 2.1$, S.D. = 0.87), video with feedback ($M = 2.4$, S.D. = 1.15), Kinect ($M = 1.7$, S.D. = 0.96) and Kinect with feedback ($M=1.9$, S.D. = 1.28) than icons with feedback ($M = 0.1$, S.D. = 1.37). Video with feedback got the highest average ratings. One of the reasons for this could be that videos along with the recognizer feedback are much more suitable to display on the smartphone and no external display is needed.

Figure 3.10 shows the results of the second question. i.e., if participants would like to have motion gestures as an input modality along with touch (surface) gestures. A one-way analysis of variance again shows that teaching technique had a significant effect on the average rating of whether participants would like to have motion gestures as an input modality along with surface gestures ($F_{4,45} = 5.045$, $p < 0.05$). Post-hoc analysis using Bonferroni correction showed significant differences between Kinect with feedback and icons with feedback ($p < 0.05$), videos with feedback and icons with feedback ($p < 0.05$) and finally, between videos and icons with feedback ($p < 0.05$). Participants gave significantly better ratings in the case of video ($M = 1.5$, S.D. = 1.5), video with feedback ($M = 2.4$, S.D. = 0.69) and Kinect with feedback ($M= 2$, S.D. = 1.05) than icons with feedback ($M = 0.3$, S.D. = 1.7)

Again, video with feedback got the highest average ratings, higher than both the Kinect-based teaching mechanisms. One of the reasons for this could be that videos along with the recognizer feedback are much more believable as a prospective teaching method on the smartphone than those that require any external display like the Kinect based- mechanisms.

Chapter 4

Discussion and Limitations - Teaching Motion Gestures

4.1 Teaching Motion gestures

Our experiments demonstrate that, as a teaching mechanism, showing a video of a desired gesture on the phone along with some feedback of how close a gesture is to optimal can effectively aid learning of motion gestures. This clearly demonstrates the viability of training end users to perform motion gestures using only the smartphone display. Teaching on the phone itself is more pragmatic, as you do not need specialized hardware. This means that people can learn motion gestures over time easily.

Also, we saw the same "annealing" process in our experiment that Negulescu et al. [29] observed when evaluating motion gesture interaction. Based on the feedback provided on the phone, participants changed their gesture accordingly till they mastered it. Given that we saw slower improvement with video only, it is not just an instance of needing a couple of tries but some guidance on how to improve the gesture helps.

4.2 Limitations

We acknowledge that the four gestures in our evaluation were simple gestures, requiring only lateral or vertical motion of the phone. For complicated gestures, e.g. gestures using twists or curves, only providing feedback about how close a person is to the desired gesture may not be sufficient. The exact path of the gesture may need to be shown to the user in an efficient manner.

We also conducted semi-structured interviews. These interviews led to many interesting themes and qualitative feedback regarding motion gestures and are described in the next section below.

4.3 User's opinions and qualitative feedback on Motion Gestures

We also conducted an exit semi-structured interview after each participant completed the experiment. Transcripts of the recorded interviews were used to identify common themes that emerged from our study.

Subtle gestures

16 out of 50 participants commented about the kind of motion involved in the motion gesture. A common theme that emerged was that the gesture should involve as little movement as possible.

Well, I mean motion gestures are ok and all, but I would rather use my wrist than using my arm.[P28].

The four kinds of gestures in this study did involve some lateral and vertical arm movement. Participants felt that too much arm movement in any gesture would be strenuous and might also invade an adjoining person's private space.

If I'm in a packed place or say on the bus, my arm might accidentally bump into the person next to me while doing the gesture. [P12].

Social Acceptability

36 out of 50 participants indicated their fondness for motion gestures and mentioned that, just like any new technology, motion gestures would eventually be accepted and used in public.

I don't mind these in public. I think they're pretty cool. [P21]

I think motion gestures could go mainstream really soon. It's kind of a cool new technology after all. Eventually everyone would be using them. [P40]

However a few of the participants indicated that motion gestures may become "awkward" in public places.

I would feel weird doing them in public. If all of us start doing motion gestures, it'll feel like a crazy world.[P3]

Fatigue

12 out of 50 participants mentioned that with prolonged use, motion gestures may cause some damage to the arm, especially for older people.

With prolonged use, my arms could pain and the older folks, say my grandfather, wouldn't want to do these at all. [P11]

Individual privacy

9 out of 50 participants indicated that, if motion gestures are standardized, then observers may be more aware of their actions, i.e. that the observability of motion gestures may result in a loss of privacy.

If all motion gestures are the same, your motion might indicate what you're doing. Other people might see me doing actions on the phone which I, you know, don't want to show them. [P12]

False positives/negatives

The last theme that came up from the quotes of many participants (33 out of 50) was the problem of false positives and negatives. Participants mentioned the problem of distinguishing everyday motion

from motion gestures and minimizing false positives. They also said that the recognizer should be very responsive and should have a minimal false negative rate.

What if I'm like, running with the phone in my pocket or maybe stretching? Then if I accidentally start calling someone, that would be a big problem. [P35]

As we note earlier, significant past work addresses the question of balancing false positives and false negatives [13, 16].

Gamification of Recognizer Feedback

31 out of 50 participants indicated that one of the reasons they liked the DTW feedback in our study was that it challenged them to get the arrow point to the correct (green) area of the bar and as high as possible on each attempt.

I felt like, you know, I can totally do this. I just didn't want to let that arrow to drop down. It was fun. [P17]

4.4 Summary

In chapter 3, we looked at different ways of training motion gestures. Our experiments demonstrate that, as a teaching mechanism, showing a video of a desired gesture on the phone along with some feedback of how close a gesture is to optimal can effectively aid learning of motion gestures. This clearly demonstrates the viability of training end users to perform motion gestures using only the smartphone display.

However, another aspect that we need to explore is how to support reliable recognition. The following chapters – Chapter 5 and Chapter 6 deal with the aspect of reliable recognition of motion gestures.

Chapter 5

Observational study to analyze the Cognitive Effects of Bi-level Thresholding.

5.1 Introduction

This chapter addresses the reliability of motion gesture recognition, and, in particular, whether an approach such as bi-level thresholding (described previously in chapter 2) can be used to enhance the perceived reliability of recognition. It may be the case that when recognition rates are similar, bi-level thresholding helps a lot over fixed level thresholding (single threshold). However, it may also be the case that, when recognition rates are similar, bi-level thresholding does not help.

In the following sections in this chapter, the usefulness of bi-level thresholding as a recognition strategy is explored. We do this by via an empirical evaluation of bi-level thresholding versus a standard optimal recognition algorithm. Our results support the utility of bi-level thresholding as a technique to lower the mental workload associated with performing motion gestures.

5.2 The Experiment

To assess the usability of the bi-level threshold recognizer, we conducted an experiment that evaluated bi-level thresholding (BL) against fixed-level thresholds (FL). We simulated 3 levels of recognition rates: 50%, 60% and 70%. Finally, because a smartphone is used in the various situations, we tested our recognizers in two separate conditions; walking and seated condition. As a result, our experimental design consisted of a 2X3X2 mixed design with bi-level/fixed-level threshold (BL/FL) as a within subjects factor and recognition rate (50, 60, 70) and seated/walking as between subjects factors. This work was done in collaboration with Keiko Katsuragawa, our lab member.

5.2.1 Experimental System

In this subsection, we describe the experimental system we developed.

5.2.1.1 Gestures

Ruiz et al. [37] noted that when end-users design motion gestures, the gestures they select tend to be simple (non-compound), single-axis movements with low kinematic impulse. As a result, we base our study around five single-axis gestures – *right flick (1), left flick (2), flick up towards face (3), flick down away from face (4)* and *double flip (5)*. The images of five gestures are shown in the Figure 5.1; these gestures were drawn directly from Ruiz et al.'s consensus set.

We argue that the selected gestures represent the simplest set of useful motion gestures for smartphone control. Nominally, the gestures correspond to next, previous, zoom-in, zoom-out and mode switch (delimiter) gestures, labeled 1 through 5 respectively in Figure 5.1. Essentially, we chose the gestures we did both because they represent a useful subset of potential commands issuable via motion gestures and because the selected gestures are the types of gestures – single axis, low kinematic impulse – the participants specified when gestures were elicited from them [37].

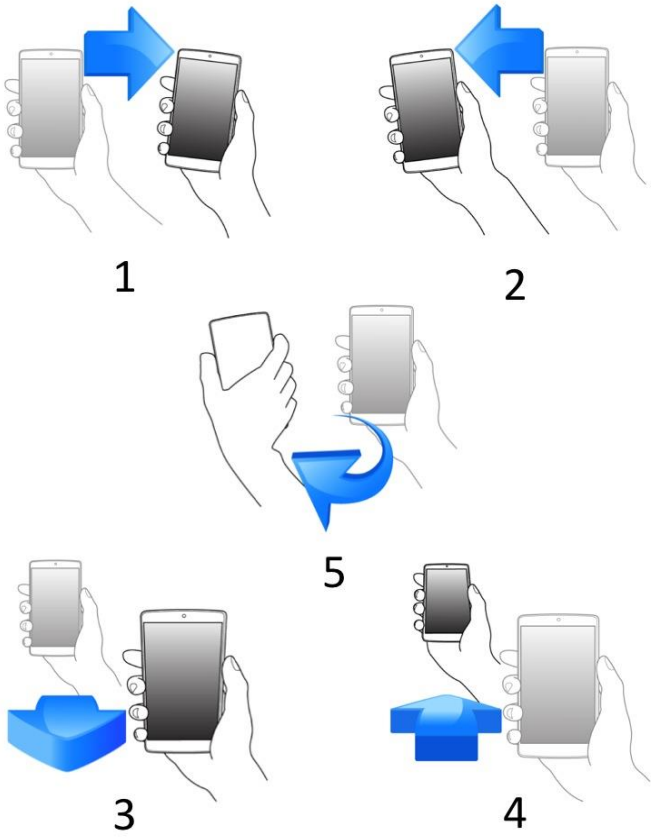


Figure 5. 1 Five Gestures for the experiment.

5.2.1.2 Apparatus

Our experimental softare was developed in Java using the Android SDK [1]. Software ran on Nexus 5 phones with a 2.26 GHz quad-core Krait 400 processor and a three-axis accelerometer and gyroscope. The Android version used was KitKat 4.4.4.

5.2.1.3 Recognizer

To discriminate between a deliberate gesture and noise, we used a dynamic time warping (DTW) algorithm [41]. Two expert users iteratively decided on the threshold values for each of the 5 gestures within the recognizer. We found that, for our experiment where participants were constantly performing motion gestures, a low threshold simply ensured that participants would need to perform an action to activate the threshold. However, the thresholds were sufficiently permissive that we observed no false negatives. On the other hand, if used in practice these thresholds would result in a high false positive

rate. The thresholds were only appropriate for a situation where a simulated recognition algorithm was being used.

To control error rate, we simulated (wizard-of-ozzed) the recognition such that gestures that exceeded our base-level threshold for a gesture (tuned above) were either correctly or incorrectly recognized based on a probability function. When the recorded acceleration or gyroscope value would reach beyond the threshold, recognition result (either correct or incorrect based on our desired recognition rate) would be displayed. Since we simulate the recognition rates, there will be some instances where even perfectly performed gestures will fail, but the important thing is that is the case anyway in actual use since accelerometer data is noisy.

We collected the raw accelerometer and gyroscope data. All raw sensor data was saved with timestamps into files. The sampling rate for both accelerometer and gyroscope was 50 Hz. We also collected the timestamps for all important events like the cue of each task, and successful or unsuccessful gestures.

We controlled error rate by equalizing the number of attempts that participants needed to perform 50 successful motion gestures. To understand how recognition rate was equalized, consider Table 5.1. The columns 1 to 6 show the number of gestures attempts and their corresponding row values indicate the frequency for each attempt. For example, for a recognition rate of 70% and for the fixed threshold recognizer, recognition was presented as correct on the first attempt 35 times (70% of 50 gestures), correct on the second attempt 11 times (70% of 15 (50 - 35) gestures), correct on the third attempt 3 times (70% of 4 (15 -11) gestures) and correct on the fourth attempt once. This gives a total of 35x1 + 11x2 + 3x3 + 1x4 = 70 gesture attempts and 20 unsuccessful gesture attempts (errors). Similarly for the 70% case and bi-level threshold recognizer, there were a total of 30x1 + 20x2 = 70 gesture attempts and 20 unsuccessful attempts. As a result, 50 correct gestures were recognized out of 70 gesture attempts, giving an overall recognition rate of 50/70 or 71.4%.

		Required attempts						Total attempts
		1	**2**	**3**	**4**	**5**	**6**	
70%	Fixed	35	11	3	1			70
	Bi-level	30	20					70
60%	Fixed	30	12	5	2	1		82
	Bi-level	18	32					82
50%	Fixed	25	13	6	3	2	1	97
	Bi-level	0	50					100

Table 5. 1 The number of required attempts and frequency.

The 50% case is particularly interesting from the perspective of bi-level thresholding. To preserve parity in recognition rates (so that we could determine whether overall recognition rate or bi-level thresholding was most effective at enhancing usability) participants performed 50 correct gestures out of approximately 100 gesture attempts. For the bi-level case, this means that, for 50% recognition, the gesture was always reported as incorrect on the first attempt and correct on the second attempt.

While we could have chosen different error rates, for the length of our study, we were reluctant to raise the error rate above 70% for two reasons. First, at 80%, it becomes highly unlikely that more than two attempts are needed to recognize a gesture: at 80% recognition, only two gestures would have used a third attempt, making 80% virtually identical for bi-level and fixed-level thresholding. As well, past experience preventing false positives in motion gesture input results in first-instance recognition rates that are closer to 25% [29], not the 50% rate that is the lowest recognition rate we use in this study. Given that gestures are often single-axis movements with low kinematic impulse, we find it unlikely that recognition rates would ever reach as high as 80% for first-instance recognition without resulting in prohibitively high false-positive rates.

5.2.2 Recruiting and Participants

We recruited 67 participants (43 male, 24 female, ages 20 -39) from the general student body of our institution. We advertised the study widely to get a sample of participants with diverse backgrounds and levels of experience using computers. 36 participants were assigned to the walking condition and 31 participants to the seated condition. One participant in the seated condition did not follow the

instructions in the experiment, instead trying to fool the recognizer by performing incorrect gestures, so the data of this participant had to be eliminated from the final analysis, yielding 66 data points, 36 walking and 30 seated.

All participants owned a smartphone and knew what motion gestures were, but not with respect to movement of the smartphone device. Some of the participants were familiar with some hand gestures above the screen that can be performed on the Samsung Galaxy S4 Android device. All participants were remunerated with $10 cash after the completion of the experiment.

5.2.3 Measures

We capture both self-report and objective measures from our participants.

5.2.3.1 Self-Report Measures

Our self-report data consists of two questionnaires and an interview. After each block, participants completed a computerized version of the NASA Task Load Index (NASA-TLX) [15]. The NASA-TLX is a widely used questionnaire, the details of which are beyond the scope of this paper.

At the end of the study, participants completed a post-experiment questionnaire comprised of a set of ten point Likert scaling ratings for each block. The questionnaire included both positive questions (Likable", "Easy To Use", "Fun to use", "Comfortable", "I felt relaxed", "The application was stable", "I performed well" and "The application performed well") and negative questions ("Inefficient", "Difficult to use", "Confusing", "Tired" and "Boring") on each block. Each Likert scale was labeled with strongly disagree (0) to strongly agree (9). An additional set of four questions asked participants to compare blocks. The comparison questions were:

1. Did you oberve any difference between first and second session?
2. Which session did you like better?
3. Do you think you performed the gestures differently in those two sessions?
4. Do you think the application performed differently in those two sessions?

Question 1, 3 and 4 were a six point scale Likert labeled with "Absolutely Yes", "Probably Yes", "Somewhat Yes", "Somewhat No", "Probably No" and "Absolutely No". Question 2 was a seven point scale labeled with "Absolutely First one", "Probably First one", "Somewhat First one", "Indifferent", "Somewhat Second one", "Probably Second one" and "Absolutely Second one". Finally, two free-form questions asked participants if they or the application performed differently in the two sessions and, if

so, what the difference was. Exit interviews were primarily used to cross-validate the results of the post-experiment questionnaire.

5.2.3.2 Objective Measures

For the walking condition, we used walking speed as an objective measure in our experiment, and for the seated condition we attached physiological sensors to participants to measure Galvanic Skin Response (GSR) and Blood Volume Pressure (BVP).

During the walking condition in our study, participants were asked to walk to and from two specified points, 14.5 meters apart at their normal walking speed. We measured the walking speed by the number of steps the participants made and the distance they walked. We developed a step counter application to save the timestamp of each step on the Nexus 5 phones used in the experiment. The distance they walked was measured by counting the number of trips made between the two points. The experimenter manually recorded the timing the participant made during turns. A pair of video recorders was placed at each end of the walking path and all the gesture sessions were video recorded. All data was synchronized with the gesture input data using time stamps to assess walking speed.

In the seated condition, the physiological sensing system consisted of Galvanic Skin Response (GSR) and Blood Volume Pressure (BVP) sensors. Galvanic Skin Response, also known as skin conductance is a measure of sympathetic arousal [3,9]. GSR is often used as an index of stress. For example, Bach et.al [3] derived a summary statistic for sympathetic arousal as indexed by spontaneous fluctuations (*SF*) of the skin conductance. From the BVP signal, we obtained Heart Rate Variability (HRV) to assess the sympathetic-vagal balance of an organism. Low-frequency/high-frequency ratio (LF/HF) is an indicator of mental stress [21,26]. Mean Heart Rate (HR) and Heart Rate standard deviation (HRsd) is also obtained from BVP. We recorded physiological data using a ProComp Infiniti5 (Thought Technology Ltd.,) encoder with Biograph Infiniti software that ran on a Lenovo laptop PC, thinkpad T430s, Core i5 processor. Although the laptop PC was in the same room, the display of the laptop PC was hidden from the participant's view. Signals were sampled at a sampling rate of 256 for GSR and 2,048 for BVP

The sensors were attached to the first three fingers of the participant's hand. The participant chose the hand to use for the motion gesture and the sensors were attached to the other hand.

5.3 Study Procedure

Before the study began, we conducted a briefing session where detailed instructions about the study were communicated to the participants. During this briefing, we informed participants that they would perform a training block and two experimental blocks, but did not inform them that the recognizers were different in the two experimental blocks. After the briefing, there was a training block in which participants familiarized themselves with the gestures.

This was followed by collection of participants baseline measurements. In the case of walking condition, the baseline was just a measure of normal walking speed to and from two specified points, 14.5 meters apart. We had spotters to ensure their safety while walking. In the case of seated condition, we wrapped the GSR sensors and BVP sensor on their fingers of the hand not used for the gestures, typically their non-preferred hand. The electrodes were sanitized with rubbing alcohol between participants to maintain hygiene. The baseline measurement for the seated condition involved sitting in a relaxed state for two to three minutes. In the seated condition, the participants were asked not to talk during the baseline measurement and the gesture task because conversation can affect the physiological signal. Participants then performed the experimental blocks.

As noted above, during experimental blocks we used five gestures in our experiment - right flick, left flick, flick up towards face, flick down away from face and double flip. Each gesture was performed ten times per block, yielding 50 gestures per block. The order of gestures displayed to the participant was randomized within the block, and the order of the fixed versus bi-level block was counterbalanced. After each gesture attempt where the simulated recognizer reported a correct result, a check mark was displayed on the screen for 1 second. Following the check mark, there was a 3 second pause before the next task (gesture) was presented as an image. Dependent measures, both self-report and objective, were collected as described above.

5.4 Hypotheses

We analyze data with respect to the following hypotheses:

H1: Workload scores are lower for bi-level thresholding (BL).

H2: Subjective ratings of users were higher for BL.

H3: Walking speed was negatively affected by repeated errors in FL.

H4: Physiological measures were impacted by recognition strategy (BL/FL).

5.5 Results

5.5.1 Self-Report Measurement Results

5.5.1.1 NASA-TLX

Subjective workload was measured using the composite score of the NASA-TLX weighted workload (WWL). The estimated marginal means of WWL is shown in Figure 5.2. A three-way MANOVA of between-subjects and within-subjects effects for threshold strategy, recognition rate, and scenario was performed. The overall workload scores on the bi-level threshold recognizer shows significantly lower workload compared to the fixed threshold recognizer ($F(1,60) = 8.214$, $p < .01$). As a between subjects effect, the effect of recognition rate was significant ($F(2,60) = 4.272$, $p < 0.05$, 50% > 70%). No significant effect of condition (walking or seating) was found ($F(1,60) = 1.938$, $p = 0.169$).

The significance of the effect of thresholding (Fixed/Bi-level) on the Nasa-TLX subscales is shown in Table 5.2. A linear mixed analysis indicates significant differences for Performance(PF) $F(1,60)=7.642$, $p < .01$ and Effort(EF) $F(1,60)=8.349$, $p < .01$ and a tendency to significance was found for Frustration(FR) $F(1,60) = 3.987$, $p = .05$

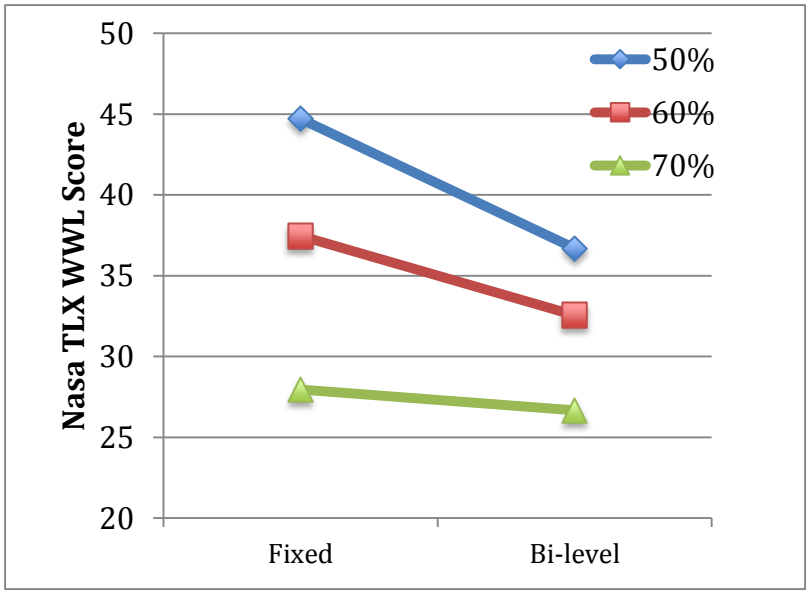


Figure 5. 2 Estimated marginal means of NASA-TLX scores.

	Fixed		**Bi-level**			
	M	SD	M	SD	F	p
WL	36.5	17.4	31.7	17.2	8.214	< .01
MD	24.4	22.4	24.2	22.4	0.02	.888
PD	35.8	24.1	33.9	25.0	1.042	.311
TD	32.4	22.0	29.0	20.9	2.183	.145
PF	28.5	19.4	22.3	17.5	7.642	< .01
EF	37.2	24.8	30.8	23.9	8.349	< .01
FR	34.7	24.9	29.9	23.8	3.987	.05

Table 5.2 MANOVA result of Nasa-TLX scores.

5.5.1.2 System Impression

Impression of the system was assessed using a 10-point scale Likert scale (0-9). The mean of positive questions are shown in Table 5.3 and negative questions are shown in Table 5.4. In the positive questions, the higher score represents more positive impression and in the negative questions, the higher score represents more negative impression. In order to analyze the influence of the threshold type difference on the system impression, a Wilcoxon Signed-Rank test was performed. The test results are also shown in Table 5.3 and Table 5.4.

Overall, the bi-level threshold recognizer had better scores than the fixed threshold recognizer. Significant differences were found for “Comfortable” ($Z = -2.069$, $p < .05$), “I felt relaxed” ($Z = -2.011$, $p < .05$), “The application was stable” ($Z = -2.827$, $p < .01$), “I performed well” ($Z = 2.87$, $p < .005$) and “The application performed well” ($Z = -1.967$, $p < .05$). For all of these statistically significant differences, the bi-level threshold recognizer outperformed the fixed threshold recognizer.

	Fixed mean	Bi-level mean	Z	p
Likeable	5.4	5.5	-0.4	.968
Easy to use	5.9	6.2	-1.623	.105
Fun to use	4.7	4.6	-0.14	.989
Comfortable	5.3	5.7	-2.069	< .05
I felt relaxed	5.6	6.1	-2.011	< .05
App was stable	5.5	6.2	-2.827	< .01
I performed well	6.4	7.0	-2.87	< .005
App performed well	5.4	5.8	-1.967	< .05

Table 5. 3 Wilcoxon rank test of positive questions result.

	Fixed mean	Bi-level mean	Z	p
Inefficient	3.8	3.6	-.808	.419
Difficult to use	3.2	2.8	-1.865	.062
Confusing	1.7	1.7	-.198	.843
Tired	3.6	3.5	-.437	.662
Boring	4.5	4.4	-.225	.822

Table 5.4 Wilcoxon rank test of negative questions result.

5.5.1.3 Preference of the system usage experience

Participants were asked to score the preference of the entire experience of each session. Participants scored the preference by choosing from a 7-point scale (Absolutely First one to Absolutely Second one). The scale was converted to three categories; fixed threshold recognizer preferable (FL), bi-level threshold recognizer preferable (BL), or indifferent/equal preference (EL). The number of choices of each category is shown in Table 5.5.

The majority of participants (n=36, 54.5%) chose the block with the bi-level threshold recognizer as their preferred block (BL), followed by fixed-threshold (FL) (n=17, 25.8%) and no difference (EL)

(n=13, 19.7%). A Chi-square test showed that there was a significant difference in these numbers ($\chi 2$ (2) = 13.727, p < .005, significant difference was shown between BL-FL and BL-EL). This result suggests that, if the overall recognition rate is the same, the bi-level threshold recognizer is more preferred than the fixed threshold recognizer.

	Fixed (FL)		**Bi-level (BL)**		**Either (EL)**	
	Walk	Seat	Walk	Seat	Walk	Seat
70%	1	3	6	4	5	3
60%	2	4	9	6	1	0
50%	4	3	7	4	1	3
Total	17(25.8%)		36(54.5%)		13(19.7%)	

Table 5. 5 The number of participants that preferred each session.

5.5.1.4 Observed difference between systems

Participants were not informed that they were using two different artificial recognizers. In the 70% recognition rate condition, more than half (55.5%) of the participants declared they did not observe any difference between the two sessions. This is shown in Figure 5.3

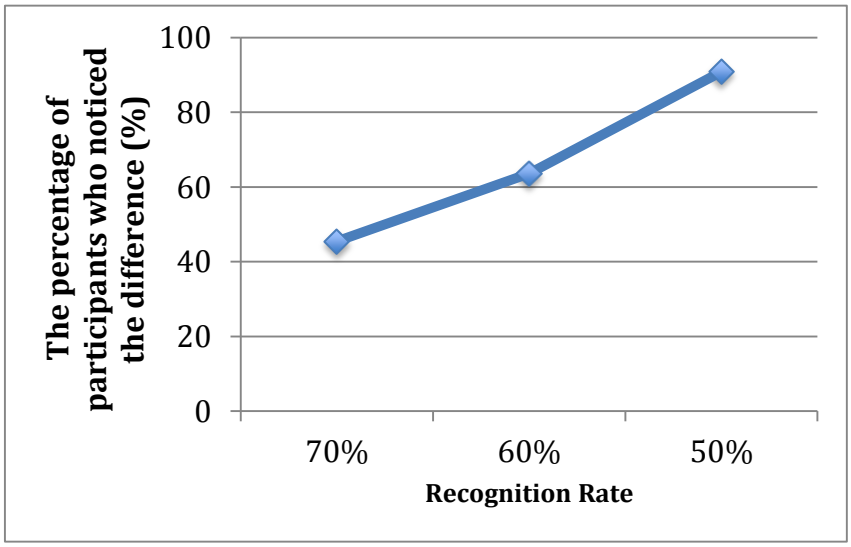


Figure 5. 3 Difference observation ratio.

Differences between the two blocks (Fixed vs Bi-Level thresholding) could be caused either by differences in the application or in the participants themselves (i.e. they performed better or worse). In

the 70% and 60% recognition rate conditions, the percentage of the participants who thought the cause of the difference was the application and the cause was themselves was about same (Figure 5.4). In the 50% recognition rate, more participants noticed the application performed differently in the two blocks, but still 27% of the participants did not notice the application performed differently.

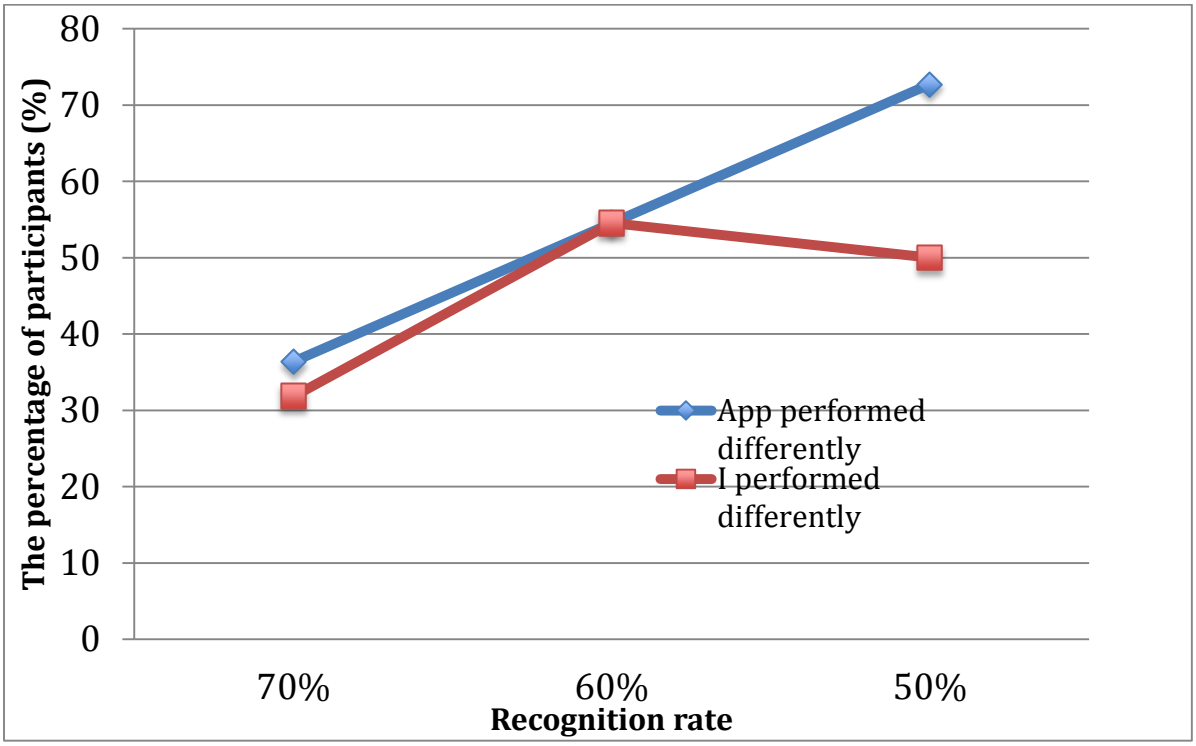


Figure 5.4 The cause of the difference in perception.

5.5.2 Objective Measures

Neither walking speed in our walking condition nor skin conductance and blood volume pressure in our seated condition showed any statistically significant differences for our study.

Chapter 6

Discussion and Limitations – Bi-level Thresholding for Perceived Reliability

6.1 Cognitive effects of Bi-Level Thresholding

Revisiting our hypotheses, we see that hypotheses H1 and H2 which are based upon our self-report data were both supported by our results. Bi-level thresholding places statistically significantly lower mental workload on our participants, and participants prefer the bi-level thresholding strategy. However, H3 and H4, our hypotheses grounded in our objective measures – walking speed and physiological sensors – did not reveal statistically significant differences, leading us to reject H3 and H4 and conclude that, despite higher reported workload, this higher workload is not being demonstrated in physiological arousal.

The lack of significance in our physiological data is perhaps unsurprising. Our task were relatively benign, and much of the work on physiological measures of stress is grounded in risky decision making tasks [9], not in more benign tasks like the one used in our experiment.

However, the success of bi-level thresholding in this work is, perhaps, more surprising. Consider : each of our participants performed exactly the same number of gesture attempts in both blocks of our experiments. There are no *physical* workload benefits to the bi-level thresholding condition because, at any one recognition rate, participants still perform exactly the same number of gesture attempts to complete 50 gestures. For example, participants who received the 70% recognition rate performed 70 gesture attempts for both fixed and bi-level thresholds; it simply is the case that, with bi-level thresholding, they fail more frequently on the first attempt, but achieve more reliable first or second attempt recognition than does a user providing input to a system with a fixed-level threshold.

One of the observations that motivated bi-level thresholding was the high cost of repeated errors. Both in Negulescu et al. [29] and here, we claim that first-instance failure seems much lower cost than subsequent failures – the observation that participants would stop after a second or third failure and try to diagnose why the error was occurring. Our results validate that the cost of repeated errors is

disproportionately more important than the overall error rate of a system, specifically because in this experiment we hold error rate constant.

6.2 Implications for interaction design

6.2.1 Usability Improvement with Bi-Level Tresholding

In many situations, designers and developers need to make difficult decisions about allocating resources to improve systems. In the case of motion gestures, one of the trade-offs developers must make is whether to improve overall recognition or simply to improve systems such that repeated errors become less frequent. While doing both would undoubtedly be the ideal, in the real world resources are often tight.

Given the need to trade-off limited resources, exploring additional ways to guard against repeated errors may prove an effective long-terms solution to enhancing the perceived reliability of recognition algorithms for motion gestures. Our results, together with other recent results [29] seem to demonstrate that you can both enhance user satisfaction and improve overall recognition rates [29] by considering any candidate motion in the context of movement immediately preceding or following the candidate motion.

6.2.2 Preventing False Positives More Viable

Errors of commission (false positives) can be very costly in user interfaces. In many ways, a false negative simply requires that a user try again, whereas a system that performs an incorrect action requires that the user determine that an incorrect action was performed, undo that incorrect action, and then try to perform his or her desired action again. One of the tensions to recognizer design in interfaces is the trade-off that must occur when selecting criterion values. Sufficiently tight that false positives are rare, but sufficiently loose that false negatives are not prohibitively high is the rule of thumb.

Our results argue that it may, potentially, be possible to satisfy both goals through a more restricted criterion function for first attempts followed by a looser function for subsequent attempts. It seems theoretically possible that such a strategy may represent the ‘best of both worlds’.

6.2.3 Workload of the Motion Gesture Interface

One positive aspect of our study is the lack of effect of motion gestures on walking speed. Motion gestures, in that they leverage proprioperception, can potentially be performed eyes-free. Given the

relatively visible output on our screen, it seems that participants were able to parallelize perceiving recognizer output and attending to another task with low mental workload, i.e. walking.

6.3 Limitations

In our study, one challenge with generalizing results is that the task was quite simple and may not be fully ecologically valid. Participants were cued and then performed a specific motion gesture. In real-world use, participants may be more concerned about reliability in interaction than our participants.

As well, in many ways our study design unfairly penalizes bi-level thresholding. In Negulescu et al.'s earlier work [29], they found that bi-level thresholding enhances recognition rate overall. For example, a 70% recognition rate with bi-level thresholding might increase recognition accuracy to 75%. For a 50% recognizer, accuracy would increase to approximately 67%. This is because false positives are more limited on second attempts. Overall, our study shows that, beyond the recognition benefits of bi-level thresholding, even when recognition rates are the same, bi-level thresholding is preferred to fixed-threshold recognition.

Chapter 7

Conclusion and Future Work

7.1 Introduction

In this thesis, we were particularly interested in motion gestures as an input modality for modern smartphones. The reasons for this are twofold. First, modern smartphones contain an evolving set of sensors for recognizing movement of the phone, including accelerometers, gyroscopes and cameras, so the technology already exists to support motion gesture input. Second, using a motion gesture provides many attendant benefits, including an expanded input space and the ability to issue commands eyes-free without using the touch screen by leveraging proprioception [30]. In this thesis, we – **(1) explore mechanisms to teach end users motion gestures and (2) analyze the user's perceived reliability of motion gesture recognition**. This thesis contributes in both of these areas and the conclusions are presented in the following sections.

7.2 Teaching Motion Gestures

We addresses the challenge of teaching people to do motion gestures. Specifically, we examine two factors. The first factor is how to represent motion gestures: as icons that describe movement, video that depicts movement using the smartphone screen, or a Kinect-based teaching mechanism that captures and depicts the gesture on an external display in three-dimensional space. The second factor we examine is recognizer feedback, i.e. a simple representation of the proximity of a motion gesture to the desired motion gesture based on a distance metric extracted from the recognizer. We show that, by combining video with recognizer feedback, participants master motion gestures almost equally quickly as end users that learn using a Kinect and perform equally well.

7.3 Perceived Reliability of recognizers

Overall, the lesson to be drawn from this part of our research is simple: If a user's input is a near miss to something that may be a specific command, then that near miss provides valuable information which can be used to enhance the perceived reliability of recognition-based interactions. We show that, by doing this, we lower the mental workload of end-users and increase their satisfaction, even when the overall number of attempts they make to perform actions remains constant.

The significant effect we see on mental workload remains surprising because, in our experimental design, bi-level thresholding did not save any physical effort. Participants still performed exactly the same number of gesture attempts, but the reduced first-attempt reliability was more than offset by the enhanced second-attempt reliability.

We feel that the overall benefit to these results is specifically in the perceived reliability of interfaces that incorporate recognition algorithms. Overall, the promise seems to be that we can be slightly more aggressive in preventing false positives while leveraging near-misses to prevent repeated false negatives.

7.4 Future Work

One of the things to work upon in the future is the design space of more complicated motion gestures like curve gestures and how to teach them. Is the kind of feedback we provided in our work sufficient or would we need more enhanced techniques of giving feedback in order to properly convey the path constraints of the gesture? Another question to ask is do we need delimiters for motion gestures? Are these delimiters necessary to distinguish noise from the gesture or can we use something else instead? Also, merging surface and motion gestures together in the future may arise to a better and more expanded input modality for smartphones. Then, enabling efficient use of motion gestures in a multi device environment is something to consider.

7.4.1 Broader Implications to Gestural Input

Overall, we have found only limited use of techniques like bi-level thresholding in other domains. However, in any domain where computational intelligence is used to interpret input (sketch recognition, assistive technologies, speech input), bi-level thresholding may make sense as a recognition strategy. Exploring this would be a useful area of future work as well.

7.5 Final Conclusion

Despite the rising popularity of the phablet form factor, the trade-off between portability and input will continue to place limitations on what can practically and efficiently be supported on a smartphone display screen. Screen size and the multi-touch paradigm limits the number of display widgets and the size of widgets, thus forcing feature rich applications to either provide multiple input screen or to seek outside input modalities.

We believe that the expressive nature of everyday movement in the form of motion gestures can be leveraged by modern smartphones as an attendant input modality and, with this in mind, this thesis explores techniques to teach users a gestural language and explores the perceived reliability of gestural recognition systems.

REFERENCES

[1] Android Software Development Kit (SDK), http://developer.android.com/sdk/index.html.

[2] Ashbrook, D., Starner, T., MAGIC: a motion gesture design tool, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (2010).

[3] Bach, D.R., Friston, K.J. and Dolan, R.J., "Analytic measures for quantification of arousal from spontaneous skin conductance fluctuations". International Journal of Psychophysiology, 76 (2010), 52-55.

[4] Bartlett, J., "Rock 'n' Scroll Is Here to Stay", IEEE Computer Graphics and Applications, 20, 3 (2000), 40-45.

[5] Bau, O., and Mackay,W., "OctoPocus: A Dynamic Guide for Learning Gesture-Based Command Sets.", Proc. of UIST'08, 37-46.

[6] Björn, H., Leith, A., Mittal M. and Klemmer S., "Authoring sensor-based interactions by demonstration with direct manipulation and pattern recognition", Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, (2007).

[7] Bolt, R. Put-that-there: Voice and gesture at the graphics interface, Computer Graphics, 14, 3 (1980), 262-270.

[8] Bragdon, A., Zeleznik, R., Williamson, B., Miller, T., and Laviola, J. J., "Gesturebar: improving the approachability of gesture-based interfaces." Proc. CHI 2009, 2269–2278.

[9] Boucsein, W., Electrodermal activity (2nd Ed). New York: Springer, 2012.

[10] Bucolo, S., Billinghurst, M. and Sickinger, D., "User experiences with mobile phone camera game interfaces," Proc, MUM 2005, 87-94.

[11] Callahan, J., Hopkins, D., Weiser, M. & Shneiderman, B.,"An empirical comparison of pie vs. linear menus", Proc. CHI'88, 95-100.

[12] Fawcelt, T., An Introduction to ROC Analysis. Pattern Recognition Letters 27, 8 (2006) 861-874.

[13] Flash T. and Hogan N. The coordination of arm movements: an experimentally confirmed mathematical model, Journal of Neuroscience, 5 (1985), 1688–1703.

[14] Harrison, C. and Hudson, S. E. Scratch input. Proceedings of UIST '08 (2008), 205.

[15] Hart, S.G. and Staveland, L.E., Development of NASA-TLX (Task Load Index): results of empirical and theoretical research. In Human Mental Workload (1988).

[16] Hastie, T., Tibshirani, R and Friedman, J., The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer (2009), ISBN 0387848576.

[17] Hinckley K., Pierce, J., Sinclair, M. and Horvitz, E., Sensing techniques for mobile interaction, In *Proc. UIST 2000*, ACM Press (2000), 91-100.

[18] Jones, E., Alexander, J., Andreou, A., Irani, P., and Subramanian, S. GesText: Accelerometer-based Gestural Text-Entry Systems. In Proc. CHI 2010, ACM Press (2010).

[19] Kamal, A., Li, Y., Lank, E., Teaching motion gestures via recognizer feedback. IUI 2014: 73-82.

[20] Kar, B., Dutta, P. K., Basu, T. K., Vielhauer, C., Dittmann, J., "DTW based verification scheme of biometric signatures", Proc. ICIT 2006, 381-386.

[21] Kristalboneh, E., Raifel, M., Froom, P. and Ribak, J., Heart-rate-variability in health and disease, Scandinavian Journal of Work Environment & Health, 21 (2) (1995), 85-95.

[22] Kurtenbach, G. (1993) "The Design and Evaluation of Marking Menus", Ph. D. Thesis, Dept. of Computer Science, University of Toronto.

[23] Li, F., Dearman, D., Truong, K.N, Virtual Shelves: Interactions with Orientation Aware Devices. In *Proc. UIST 2009* (2009), 125-128.

[24] Li, Y., Protractor: A fast and accurate gesture recognizer. In *Proc. CHI 2010,*.ACM Press (2010), 2169-2172.

[25] Liu, J., Zhong, L., Wickramasuriya, J., and Vasudevan, V. User evaluation of lightweight user authentication with a single tri-axis accelerometer. In Proc. MobileCHI '09, ACM Press (2009), 1–10.

[26] Malik, 1996, Heart rate variability. Standards of measurement, physiological interpretation, and clinical use. Task Force of the European Society of Cardiology and the North American Society of Pacing and Electrophysiology, European Heart Journal, 17 (3) (1996), 354–381.

[27] Mignot, C., Valot, C. and Carbonell, N. An experimental study of future "natural" multimodal human-computer interaction. Proceddings of INTERACT '93 and CHI '93 (1993), 67-68.

[28] Myers, C. and Rabiner, L., "A comparative study of several dynamic time-warping algorithms for connected word recognition", The Bell System Tech Journal 60, 7 (1981), 1389- 1409.

[29] Negulescu. M., Ruiz, J. and Lank, E., A Recognition Safety Net: Bi-Level Thresholding for Mobile Motion Gestures, *Proc. MobileHCI 2012* (2012), 147-150.

[30] Negulescu, M., Ruiz, J., Li., Y. and Lank, E., "Tap, Swipe, Move: Attentional Demands for Distracted Smartphone Input", Proc. AVI 2012, 173-180.

[31] nuiCapture Analyze, Cadavid Concepts Inc.

[32] Partridge, K., Chatterjee, S., Sazawal, V., Borriello, G., and Want, R. TiltType: accelerometer-supported text entry for very small devices. In Proc. UIST '02, ACM Press (2002), 201–204.

[33] Rekimoto, J., Tilting operations for small screen interfaces. In *Proc. UIST '96*, ACM (1996), 167–168.

[34] Rigoll, G., Kosmala, A. and Eickeler, S., High Performance Real-Time Gesture Recognition Using Hidden Markov Models. LNCS (1998) 69-80.

[35] Rubine, D., Specifying gestures by example. SIGGRAPH Comput. Graph. 25, 4 (1991), 329-337.

[36] Ruiz, J. and Li, Y., DoubleFlip: a motion gesture delimiter for mobile interaction, In *Proc. CHI 2011*, ACM Press (2011), 2717–2720.

[37] Ruiz, J., Li, Y. and Lank, E., User-Defined Motion Gestures for Mobile Interaction, In *Proc. CHI 2011*, ACM Press (2011), 197-206.

[38] Sezgin, T. M. and Davis, R., HMM-based efficient sketch recognition. In *Proc. IUI '05*, ACM (2005), 281-283.

[39] Small D. and Ishii H., Design of spatially aware graspable displays, CHI '97 Extended Abstracts on Human Factors in Computing Systems (1997), 22-27.

[40] Uchida, S., A Survey of Elastic Matching Techniques for Handwritten Character Recognition. IEICE TRANS. INF. & SYST. E88-D (8).

[41] Vintsyuk, T.K. Speech discrimination by dynamic programming, Kibernetika, 4 (1968), 81-88 .

[42] Vogel, D. and Balakrishnan, R., Distant freehand pointing and clicking on very large high resolution displays. UIST 2005, 33-42.

[43] Voida, S., Podlaseck, M., Kjeldsen, R. and Pinhanez, C., A study on the manipulation of 2D objects in a projector/camera-based augmented reality environment, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (2005).

[44] Weberg, L., Brange, T., and Hansson, \.W. A piece of butter on the PDA display. CHI '01 extended abstracts, ACM Press (2001), 435–436.

[45] Wigdor, D. and Balakrishnan, R. TiltText: using tilt for text input to mobile phones. In Proc. UIST '03, ACM (2003), 81–90.

[46] Wobbrock, J.O., Morris, M.R. and Wilson, A.D., User-defined gestures for surface computing, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (2009).

[47] Wobbrock, J.O., Wilson, A.D. and Li, Y., Gestures without libraries, toolkits or training: A $1 recognizer for user interface prototypes, In *Proc. UIST '07*, ACM Press (2007), 159-168.