Near Touch Interactions: Understanding Grab and Release Actions

by

Aras Balali Moghaddam
B.Sc., Thompson Rivers University, 2009

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

© Aras Balali Moghaddam, 2012
University of Victoria

All rights reserved. This thesis may not be reproduced in whole or in part, by photocopying or other means, without the permission of the author.

Near Touch Interactions: Understanding Grab and Release Actions

by

Aras Balali Moghaddam
B.Sc., Thompson Rivers University, 2009

Supervisory Committee

Dr. Melanie Tory, Co-Supervisor
(Department of Computer Science)

Dr. Colin Swindells, Co-Supervisor
(Department of Computer Science)

Supervisory Committee

Dr. Melanie Tory, Co-Supervisor
(Department of Computer Science)

Dr. Colin Swindells, Co-Supervisor
(Department of Computer Science)

ABSTRACT

In this work, I present empirically validated techniques to realize gesture and touch interaction using a novel near touch tracking system. This study focuses on identifying the intended center of action for grab and release gestures close to an interactive surface. Results of this experiment inform a linear model that can approximate the intended location of grab and release actions with an accuracy of $R^2 = 0.95$ for horizontal position and $R^2 = 0.84$ for vertical position. I also present an approach for distinguishing which hand was used to perform the interaction. These empirical model data and near touch tracking system contributions provide new opportunities for natural and intuitive hand interactions with computing surfaces.

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

I would like to thank:

My Supervisors Dr. Melanie Tory and Dr. Colin Swindells, for their guidance, support and encouragement to learn and grow.

Members of the VisID lab, for participating in discussions and providing critical feedback.

Members of NUI group, for providing technical advice and participating in discussions related to this project.

"Your hand opens and closes, opens and closes. If it were always a fist or always stretched open, you would be paralysed. Your deepest presence is in every small contracting and expanding, the two as beautifully balanced and coordinated as birds' wings."

Rumi, Essential Rumi

DEDICATION

To Anastasia

Chapter 1

Introduction

This work contributes empirical results to support parameterization of grab and release actions. The high level goal of this research is to improve quality of human computer interaction using hands. It is not difficult to observe in daily life that interaction with computers using hands is far more limited compared to how we manipulate and interact with the physical world around us. Many people are capable of – and enjoy – performing complex asynchronous bimanual tasks such as playing the guitar, throwing clay pots, and carving wooden sculptures. I believe lack of accurate parameterization of hand actions is one of the primary obstacles against incorporating support for such rich hand interactions within software applications.

Most contemporary multi-touch devices only receive and use contact information with the surface. This approach limits sensing bandwidth, making it very difficult or impossible to recognize what type of motion a user is performing with their hand on near the surface. Furthermore, due to the lack of any hand tracking in most contemporary systems, it is not possible to assign modes or states to the hands. These limitations hinder the design of bimanual as well as co-located collaborative interfaces.

1.1 Guiding Research Questions

Existing research projects that explore near touch surface interactions mostly focus on developing novel hardware solutions or the design of new interactions. In addition to building upon related systems research, the current work presents empirically validated techniques that can be applied to future systems and interaction techniques. This work is guided by the following research questions.

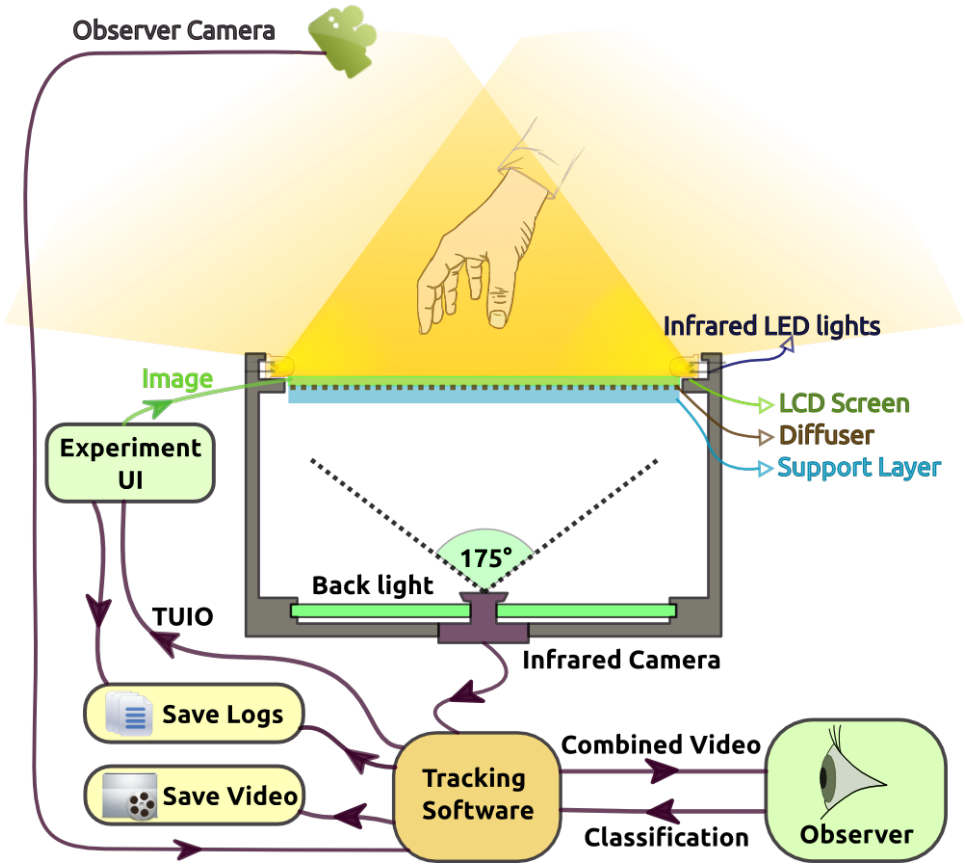

Figure 1.1: Interplay between the hardware and software components of my system to process near touch interactions. Tracking software receives frames from both an infrared camera and an observer camera. Only frames from the infrared camera are processed by computer vision algorithms. The frames from the two cameras are combined into one video stream that are superimposed with annotations. This video stream was shown to the wizard operating the Wizard of Oz system, and was stored in real time for post study analyses. The actions recognized by the wizard are sent to the tracking software by pressing shortcut keys, which in turn each trigger an event to send a TUIO message to the experiment UI in front of the participant. Both the tracking software and the experiment UI save a record in their own CSV log file containing all the data relevant to the current task. In addition to these two log files, tracking software records an OpenCV YML file containing a feature matrix with all available information from the temporal window prior to performing the task. This data can be used for training classification algorithms in future research.

1. What is an appropriate, accurate model for detecting the center of a user's action? In other words, where should a hand action (for example grab) be applied to so it is closest to where a user expects?

2. What data from typical tracking software is needed to accurately model the actions

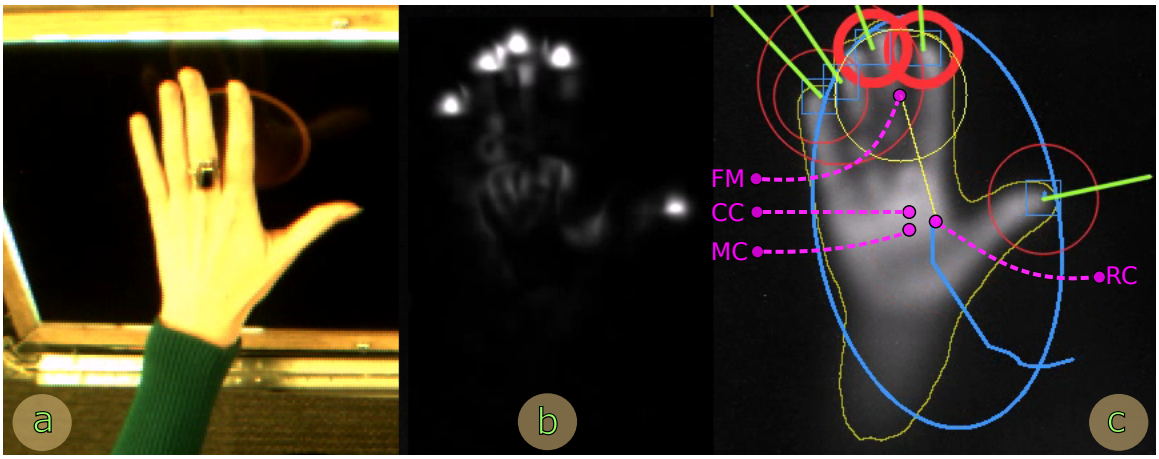

Figure 1.2: Three views of my near touch processing for a user's hand: **a)** a snapshot of a participant's hand after releasing an object on a target, **b)** the intermediate sharpness image used to approximate depth of prominent hand features such as finger tips, and **c)** the hand image as viewed through an infrared camera mounted behind the LCD display; colored annotations illustrate processing results by my system; the thin yellow line represents to the outline of the hand; the thin red circles represent fingertips near the surface but not touching; the bold red circles represent finger touches; the green lines represent finger orientation with respect to rectangle center; the thin blue line represents the bounding oval used to estimate the hand's center of mass; the magenta labels represent the four independent parameters used in my study – (i) minimum enclosing circle center (CC), (ii) minimum enclosing rectangle center (RC), (iii) feature mean location (FM) and (iv) center of mass (MC).

performed by users and what size of temporal window would be appropriate for collecting these data?

1.2 Near Touch Apparatus Summary

I introduce a prototype multi-touch and near touch tracking system (see Figure 1.1). I chose a rear mounted camera setup for my hardware prototype because I believe recent advances in display technology, including advances in manufacturing OLED (Organic Light-Emitting Diode) displays, will lead to a new generation of display technologies that are capable of capturing an image of what is directly in front of the display. For example, Hirsch et al.'s [14] BiDi screen is a thin depth sensing prototype that demonstrates feasibility of manufacturing for future systems using a grid of pinhole cameras embedded inside a flat display. My choice of hardware deals with relatively similar types of signals as Hirsch

et al. and can therefore contribute to future near touch sensing technologies in terms of hardware implementation, image processing, and interaction design.

1.3 Empirical Findings Summary

The main focus of this study is to find the most accurate model that would identify the center of grab and release actions performed by a user. Before we can create a model for the center of the hand, we need to identify what factors most influence a user's interactions. To find this information, I conducted a user study where participants performed grab and release actions on an experiment interface using my near touch apparatus. Figure 1.2 demonstrates main features of this apparatus.

Using dependent variables from the user study and based on my initial observation of users, I formulate four primary models to predict the center of users' actions. The dependent variables are parameters extracted from images of the hand in real time from a temporal window prior to performing the action. The performance of the proposed models were very similar, suggesting a high-level robustness in my approach. Another important finding of my study is an approach for distinguishing left and right hands when an action of grab or release has been registered. These center of action and hand distinction findings are detailed in the Results section.

Chapter 2

Motivation

In this chapter I intend to provide some qualitative evidence to justify usefulness of grab and release actions. Grab and release actions were among the five most preferred hand interactions in a study done by Epps et al.[10] comparing preferences of user hand interactions for tabletop displays, and are a core component of many user interaction tasks. For example, users in Epps et al. suggested grab and release to perform actions such as cut, copy, moving icons or changing a slider. I conducted a participatory design and brain storming sessions as well as a field study focused on how people use their hands while playing a board game. A summary of these studies will be presented here, because my choice of studying grab and release actions has been driven also by these studies.

2.1 Backgammon Field Study

Six individuals (three females and three males) volunteered to participate in a study that required them to play a game of backgammon against another participant. The objective of the study was to find and classify hand actions performed by users while they play a board game. Two of the participants were completely new to the game, two of them had some experience but did not identify themselves as experts, and the other two were experts. Players were paired together based on the level of expertise that they reported when they were first approached. The videos of the study were manually coded to compile a list of actions and frequency of their occurrence in the videos. Since the actions performed by users where almost always chained to achieve a task, I combined these actions whenever a common pattern was detected. For instance when I observed grabbing of dice followed by throwing it I would classify that as one *grab throw* action. The actions on this list are

Figure 2.1: A snapshot of of one backgammon session

compared in figure 2.2 grouped by the level of expertise of the participants.

2.1.1 Results

Overall grab and release was the most popular gesture used by participants. Grab and catch and dragging are among other popular gestures in this game. Gestures such as point, point and count, and drag and count were performed more often by novice users. Those participants who identify themselves as experts in the game almost never used point and count or grab and count gestures. On the other hand gestures such as dragging, grab and release multiple, and drag multiple are more common among experts.

2.1.2 Implications

Many different variations of each single gesture were observed during the study. This suggest that a semantically single gesture for example grab can be done in different ways by different users, and even by the same user during her game. Therefore when developing a system to interpret and track hand pose and movement as a gesture, we should consider all of these variations.

Another implication of this study was that people can relate *in the air* gestures to objects that are close to them. For instance when performing *point count* gestures participants move their hand directly above game board to count the columns. Therefore, I believe users can relate gestures directly above an interactive interface to the content of the interface. This

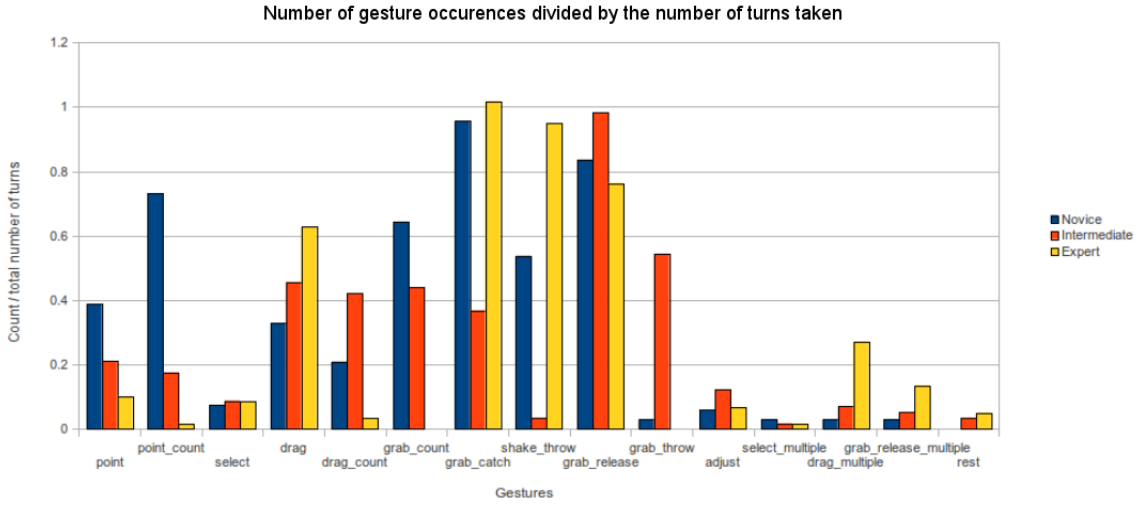

Figure 2.2: Comparison of frequency of actions performed based on participant's level of expertise in the game of backgammon. Since the number of turns in any given backgammon game could be different, the results are divided by the number of turns taken.

result suggests that computer displays would be enriched if they were equipped with a sensing technology that could track the hand directly above the image that is displayed on them.

2.2 Participatory Design Session

Participatory design sessions consisted of researchers in our lab and invited participants who represent potential users of a near touch system. The session was started by providing users with some background information and motivation of this research. It was explained to the users that I envision a system with which hand actions can be accurately detected in the area near the display screen. Each participant was given a list of selected gestures which I prepared in advance based on the result of the backgammon field study. Next, the participants were asked to brainstorm on their own for five minutes about applications of the gestures from the list that they were given. After that I took approximately 40 minutes to discuss these ideas and brainstorm other gesture applications. The application area was intentionally not limited at this stage since I wanted to find out in what type of applications in general users may want to use these gestures. Everyone was asked to use low fidelity prototyping tools that were provided to demonstrate their idea from this final list in front of a video camera so that the selected ideas could be expressed in a short video. What I found

from these sessions is that users can envision using near touch actions in various different application contexts. From the list of actions that was provided to users, *grab* and *release* generated the greatest number of application ideas. Below is a list of some of the ideas that were proposed in these sessions.

- In a file management application, use *grab* and *release* in place of cut and paste to move files and folders. Keep one finger on a file or folder while *grabbing* it with the other hand to obtain a copy of the item
- In a painting application, *grab* a few colors from a palette and mix them by shaking your hand, then splash them on the screen by performing a release action
- In your home with multiple computing devices, *grab* your news feed from a desktop computer and *release* it onto your tablet to continue reading the news on the tablet device
- To organize a dashboard with several widgets, you can *grab* a widget, move your hand above where you want it and *release* it to its new location
- While playing a video game, in order to acquire an item such as a new weapon or a first aid kit you can just *grab* it when you see it
- In a music player, perform a *grab* action above the play-list to obtain it, then shake your hand to shuffle the music, and *release* the play-list back down to start playing your music

Chapter 3

Related Work

Detecting hand actions close to (but not directly touching) an interactive surface can extend interactions in several significant ways. While comparatively smaller than the very large body of research into multitouch, near touch actions show promise as an intuitive and natural form of interaction. For example, Yang et al. [26] demonstrated that touch interactions with small targets can be improved by expanding the targets as the user's hand approaches the surface. Annett et al. [1] suggested presenting a gesture guide as context-sensitive help when a user's hand hovers over a surface. Marquardt et al.[19] argued that an interactive surface and the space immediately above it should be seen as a continuum, and that gestures on the surfaces should flow seamlessly into gestures above the surface. They demonstrated many new types of interactions, including the ability to grab an object, move it in the space above the surface, and drop it on a new location. Hilliges et al.[13] argue that such interactions are particularly useful for 3D scenes, since picking a 3D object up off of a surface is a common action in the real world, and often more natural than sliding it along a surface. Wilson and Benko [25] extended grab and release interactions to multiple surfaces, enabling users to pick up objects on one display and drop them onto another.

In order to realize such near touch interactions, we first need hardware capable of detecting interactions close to the surface. Then we need to use the input from that hardware intelligently, to understand where the user intends to interact and which hand they are using to perform the interaction. In subsections below, I summarize previous research on these topics.

3.1 Hardware for detecting near surface interactions

A wide variety of approaches have been devised for detecting gestures and the distance of the hand from a surface. For example, Takeoka et al.[22] were able to accurately detect the depth of hands interacting near a surface by stacking multiple infra-red planes above the surface. In contrast, Benko et al. [2] use a muscle sensing device to detect gestures through features such as finger detection and pressure. Still another approach is to place a light source behind or above the user and track shadows cast by the user's hands [9] [4].

My own approach uses a rear-mounted infrared (IR) camera to view diffuse IR illumination reflected off of the user's hands. Similar approaches have been used by Hilliges et al. [13], Pyryeskin et al. [21], and de FO Araújo et al. [7]. Hodges et al. [15] demonstrated that such an approach could be integrated into thin hardware devices by using a grid of infrared sensors rather than a single camera. My approach differs from these methods in that I use a type of illumination that provides an image of the hand that appears sharper as it approaches the surface, enabling a rough approximation of depth with a relatively inexpensive hardware setup. Furthermore, my method allows for tracking prominent features of the hand (such as fingertips) near the screen using an optical flow algorithm; these features are good candidates for describing actions performed by the hand.

My method uses image sharpness to distinguish touch actions from gestures above the surface. An alternative is to use totally separate input for touch actions and gestures. One common approach is to use a typical touch sensing mechanism for touch, plus shadow tracking or direct computer vision for hand tracking [8], [4]. Izadi et al. [18] similarly collect two inputs by using switchable diffusers to rapidly alternate between a view of touch points and a view of the whole hand.

3.2 Identifying the center of interaction

Regardless of the hardware, whenever we track hands above a surface, we need to understand the intended target of a user's action. Particularly when targets are small, identifying an incorrect target could lead to unexpected results. The primary contribution of my work are models to identify the intended center of interaction for whole hand actions near a surface. I am not aware of prior research that has solved this problem, but some related studies have been done for direct touch, and these inspired the design of my study. For instance, Holz and Baudisch [16] modeled the intended center of interaction for individual fingers on a touch surface. Similarly, Wang and Ren [24] measured the shape and center of finger

touches and approximated the shape as a bounding rectangle or ellipse, like some of my models.

3.3 Hand distinction

It is also important to understand which hand (left or right) is doing the interaction. As I will show later, left versus right hand impacts the intended target location. However, hand detection can also be used for many other purposes. For example, menus can be made to appear in a convenient location depending on the hand used, as described by Brandl et al [4]. For bimanual interaction, knowing the hand helps to support non-symmetric division of labour as suggested by the kinematic chain model [11]. Annett et al. [1] and Brandl et al. [3] describe numerous examples of interfaces that take advantage of knowing which hand is interacting with a surface.

Some methods exist to distinguish left and right hands, mostly for different hardware setups than my own. Annett et al. [1] identified hands on a tabletop display by using concentric rings of proximity sensors. Hands could be identified by relating them to a known position of the user's body. For FTIR touch input, Dang et al. [6] used position and orientation of the fingertips to map them to the left or right hand. Similarly, Zhang [27] distinguished hands by relating the position of a tracked index finger to the position of the hand contour; this method relies on either single finger interaction or the ability to identify the index finger. Holzammer's approach [17] relies on the fact that there is a large distance between the thumb and forefinger. Distances between detected finger positions of a fully spread hand can be used to identify the thumb and thus the hand. As a contrasting approach, Walther-Franks et al. [23] implemented a classifier using a decision tree algorithm to detect handedness from input blob sizes, blob positions, and arm blob orientation. In this work, I present a new method for distinguishing hands from camera images, using orientation of the bounding rectangle. Advantages of my method are that it is quite straightforward to implement and has relatively high accuracy.

Chapter 4

Method

I conducted a user study based on a repetitive task consisting of grab and release actions. For each task, data describing the state of users' hands and the experiment interface were logged. These data logs were then analyzed to suggest parameters that would reliably predict center of action as well as other interesting information about the action, such as which hand was used to perform it.

4.1 Study Design

Two independent variables were studied: position and size of object and target. Positions were generated randomly within a predefined square region inside the screen with the resolution of 1600 by 900. Sizes were generated randomly between 20 and 500 pixels which span the range of 0.5 cm to 13.5 cm. The four empirically tested models used a total of *eight* dependent variables:

- **(CC)** *center position* and *radius* of the minimum enclosing circle
- **(RC)** *center position*, *size* and *angle* of the minimum enclosing rectangle
- **(FM)** *mean* and *standard deviation* of feature position
- **(MC)** *position* of the center of mass

Because even slight system errors and inconsistencies can negatively impact a participant's performance or lead to undesired subconscious learning, I used a Wizard of Oz approach to classify user actions. This Wizard of Oz classifier was used to create a system that was nearly 100% accurate and enabled us to negate the classifier as a significant source of data noise in my models.

4.2 Participants

Twenty six paid participants (7 female, 19 male), were individually tested in a study that took approximately 30 minutes to complete. Their ages ranged from 20-44 years (M = 27, SD = 5.17). 24 participants were right handed, one participant was left handed, and one participant was ambidextrous. Twenty of the participants reported using at least one touch enabled device regularly.

4.3 Apparatus

During the study, participants sat on a non-adjustable, fixed chair placed in front of the display as shown in figure 4.1. The angle between the display and the horizontal plane was $60°$; the height of the display from the ground to the bottom edge was 75 cm; and, the experiment was performed indoors under uniform, controlled lighting conditions. The display used within the prototype near touch system was a $23in$ (or approximately $51x28cm$) LED-LCD with a resolution of $1920x1080$ and an aspect ratio of 1.77.

Figure 4.1: A participant about to perform a grab action on the prototype interface. Once the action is performed the yellow circular object will fade away and the target will be highlighted. A short distinct sound is given after the participant performs either the grab or release action.

4.3.1 System Hardware

Typical contemporary multi-touch hardware technologies, such as resistive and capacitive touch screens, do not capture any information about the state of the hand above the sur-

face and are not suitable for this study. A number of computer vision based techniques for building multi-touch interfaces have existed for several years [12]. The ultimate goal in the construction of most of these interfaces is to optimize tracking of finger touches on the display surface. Because these systems typically use infrared illumination and video processing techniques that filter away most of a user's hand from the picture, such techniques are unsuitable for my goal of processing whole-hand near touch interactions.

Figure 1.1 summarizes the components of my prototype system that accomplishes the empirical study goals. My prototype uses a high resolution LCD screen and a rear mounted infrared camera. The primary difference between my prototype and existing camera based interactive surface designs is in the illumination. Instead of optimizing illumination for extracting touch information, I fully illuminate the hand using an array of 72 infrared LEDs mounted just above the edges of the display as shown in figure 1.1. The LEDs have a $5mm$ diameter and produce $940 \pm 40nm$ infrared light. Motamedi [20] has performed a similar experiment and reported that by modifying the threshold algorithm, the system would be able to see the hovering hand and that further analysis of this gray-scale image could determine the distance of the hand from the screen. As a result, hands are captured by the camera up to $25cm$ away from the screen. A soft light diffuser with diffusion angle of 15° is placed under the LCD screen. The addition of the diffuser improves the quality of the image displayed, as well as providing a medium for capturing the depth of features recognized by the camera.

Algorithm 1 is used to calculate the relative distance of each identified feature from the display. This algorithm takes advantage of the fact that objects closer to the diffuser become sharper. The depth value calculated in this way also allows us to reliably distinguish touch from hovering fingers.

4.3.2 Tracking Software

I developed two software applications for the purpose of running this study: a computer vision based tracker and an interactive interface. Both software applications were executed on the same machine running an Ubuntu Linux 11.10 64 bit operating system.

To develop the tracking software I used the OpenCV computer vision library. The source code for my tracking software can be forked from https://github.com/arasbm/Gibbon. This software is capable of simultaneously tracking two hands, extracting their features, and finding fingertips and their relative distance to the screen. The distance of fingers to the screen is computed using an algorithm based on image sharpness around the fingertip.

This value is used to distinguish touch from hovering hands or fingers. The tracker communicates to interactive client applications using the TUIO (Tangible User Interface Objects – see tuio.org) protocol and my custom defined messages.

Calibration

The tracking software required the following one-time calibration process that suited the needs of all study participants. Calibration was done using OpenCV's built-in undistortion algorithm using a printed chessboard. A $16x23cm$ chess board with 8 horizontal and 6 vertical squares was placed on the screen at different locations and registered by the system 6 times. As a result the calibrated input image closely mirrors what actually is in front of the screen.

Preprocessing

After applying calibration, each frame retrieved from the camera was preprocessed using two filters which are explained below. First, a copy of the image was converted to binary using a threshold value of $T_{noise} = 30$. This binary image was used to find the contour of the hand. The shape of this contour is the basis for three of the models for the center of action that presented in this paper (CC, MC, and RC). Second, a median filter using $width = 9$ was applied to the original copy of the image after calibration. The median filter replaces each pixel in the image with median of its neighboring pixels that would fit inside a 9×9 matrix around that pixel. The purpose of the median filter is to reduce salt and pepper noise which can be described as very bright or very dark pixels randomly distributed in each frame. After this preprocessing, the image is passed through feature tracking algorithm to identify and track prominent features in the image. Each feature is a 26×26 matrix that is found using a feature detection algorithm provided by OpenCV (Open Source Computer Vision) library called *GoodFeaturesToTrack*.

Feature Depth Calculation

Consider F to be the gradient matrix representing a feature. This matrix is used to calculate sharpness and consequently relative depth of the feature as shown in algorithm 1 below. This algorithm works by first calculating the minimum eigenvalue of a matrix around each pixel in F. This data is achieved by using OpenCV *CornerMinEigenVal* function and stored in a new matrix S. Finally, obtain average of all values in S which gives a single value representing sharpness of the feature. Because of properties of the diffuser and the way

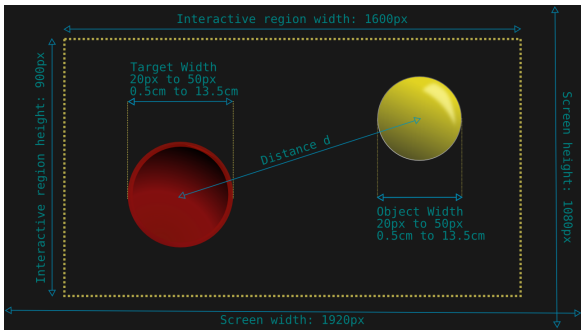

Figure 4.2: The interactive user interface used in the experiment. For each trial, an object and a target were displayed on the screen at random positions within the interaction region. The algorithm for positioning targets and objects ensured no collisions occurred, and the edges of the screen were excluded to avoid potential issues with image recognition and to have a full image of the hand in every trial. Shading was chosen based on informal pilots to minimize object vs. target confusion during grab and release actions.

it is used in construction of the hardware prototype, this sharpness value also represents relative distance of the feature from the screen.

Algorithm 1 A method for detecting relative depth of a feature based on its sharpness

F ▷ Matrix representing a feature
S ▷ Matrix to store minimum eigenvalues corresponding to each element in F
D ▷ Relative depth of feature – to be calculated
for $S_i \in S$ **do**
  T ▷ Obtain a new gradient matrix centered at S_i
  $S_i \leftarrow \lambda_{min}(T)$ ▷ Assign minimum eigen value of T to S_i
end for
$D \leftarrow 1/N \sum_i S$ ▷ Average all elements in S
return D

4.3.3 Interaction Software

The interaction software is an application that is independent from the tracking software. It is written in python using the Kivy framework [5], which has built-in components for receiving TUIO messages. Figure 4.2 shows the characteristics of the experiment user interface.

4.4 Procedure

The study used a targeting task involving grabbing an object and then releasing it on a target. In each trial, the participant was presented with an object and a target, both circular but colored differently as shown in Figure 4.2. At the beginning of a trial, the object was highlighted with a higher opacity than the target. Once the participant grabbed the object, a short sound was played and then the target was highlighted. When the participant released the object on the target, another sound was played and the screen was cleared in preparation for the next trial. I minimized biasing the participants by purposely not informing them that I was studying the center of their hand placements. Details of the study design, such as the Wizard of Oz approach, were also hidden from the participants.

After greeting participants, the study purpose was briefly explained as an exercise to understand two actions called grab and release and to find how a computer system can detect these actions. I used a printed sheet of paper to show a sample object and target of the study interface. Using the piece of paper, participants were first shown an example and then given the opportunity to practice the actions of grab and release without system feedback. I was careful not to give the participants any special instructions as how exactly to perform grab or release actions. It was expected that there will be variations in the participants' way of performing grab and release actions. If a user asked how many fingers they should use to perform grab or release, I would tell them to perform the action in any way they are more comfortable. During the study participants prominently used anywhere between 2 to 5 fingers for performing the actions. Furthermore, it was explained to the participants that:

- You can perform actions in whichever way you prefer or find most comfortable.
- The interface is capable of tracking your hand near the surface or on the surface. You do not have to touch the surface when you are performing the actions. However if you do touch the screen, that does not interfere with the task.
- You can perform the tasks with either hand and you can switch hands at any time during the study, but I ask that you only use one hand at a time on the interface.
- Please perform the tasks as quickly and accurately as possible.

After the introduction, participants were given a brief background questionnaire. To begin the study, participants were instructed to sit in front of the prototype and put on a set of headphones to block any distracting ambient sounds and to hear the sound feedback

to each trial. Figure 4.1 shows a participant as she is about to perform a grab action on my prototype. Each participant first performed 20 practice trials to become comfortable with the study prototype and trial task. The data from practice trials were not analyzed. The participant then performed 400 trials in two sessions (200 each), taking a five minute break between sessions. Object and target positions were randomized for each trial. A

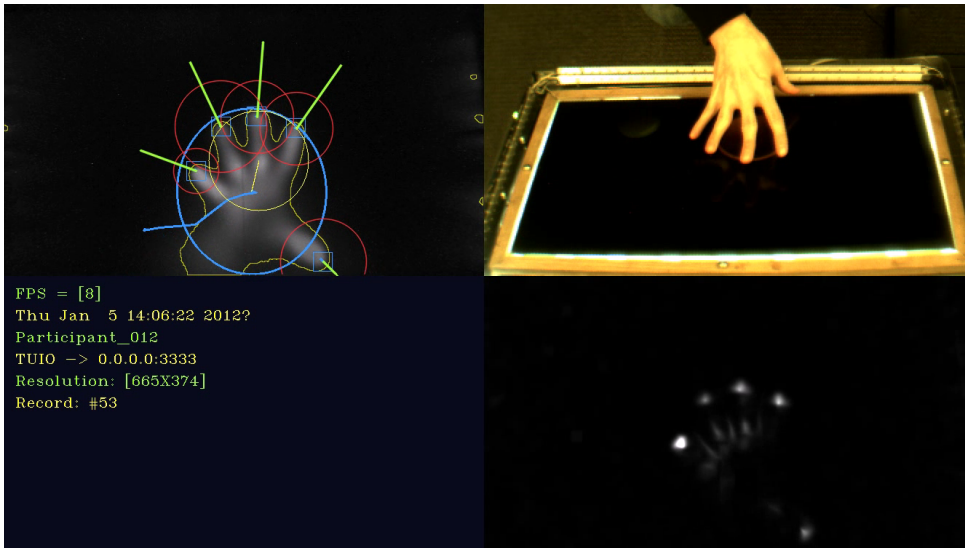

Figure 4.3: Example Wizard of Oz screen. The wizard is able to anticipate participant actions by viewing the overhead camera video (top right). Near touch and contact interactions can be viewed with the hand video (top left) and intermediate sharpness video (bottom right). Logging actions can be monitored on the record screen (bottom left). Actions were registered by pressing either the *J* or *K* key on the keyboard.

human wizard was in charge of detecting and sending commands corresponding to grab and release actions, while the system simultaneously tracked a user's hand. Figure 4.3 shows an example Wizard of Oz screen viewed by the wizard. For each action detected by the wizard, the system stored a record describing the shape, features, and position of the hand for twelve frames. The object size and location, target size and location, participant number, and time of action were also appended to each record. Each trial included two records, one for grab and one for release.

After the study, users were given a brief questionnaire. Participants were asked to rate fatigue on a multiple choice scale and then provide open-ended comments suggesting (i) any new actions that they thought should be detected by a future system, (ii) interaction ideas based on grab and release actions, and (iii) other general comments.

4.5 Data Collection

For the purpose of the current experiment three log files were recorded in real time – two files by tracking software and one by the experiment user interface. The tracking software identifies features of the hand and records several attributes of each feature and additional data about the state of the hand such as contour of the hand in a CSV (comma separated values) file. Additionally, all the available parameters are stored redundantly as a feature matrix in a second file using the YAML (YAML Ain't Markup Language) format. The purpose of this second file is to store descriptive parameters in a format that is suitable for training algorithms. This data was stored with the hope that in future myself or someone else could use it to train classifier algorithms for grab and release actions. The third log file was stored by the experiment user interface in a CSV (comma separated values) format. This file contains parameters about the state of the application interface when an action message was received.

At the time of designing the study and developing the experiment software, it was not known which attributes would play an important role in modeling center of action, therefore I decided to collect any attributes of features and other descriptive parameters based on image of the hand that I suspected may have an effect on models for the center of action. The choice for including these parameters is based on my initial observation of users during pilot studies and the availability of the parameters when developing the tracking software. The log files were combined after each session into one larger data file based on matching record number and time stamp. Each record in the combined data contains all the data collected during a temporal window just before either a grab or a release actions was detected. Table 4.5 shows a list of all relevant fields in the combined data file and sample values that were used during the analysis and modeling process. Other parameters that were also stored but not used during the modeling process include:

- 10 Spatial and central image moments which are standard statistical properties of an image
- Position of each feature at every frame within the temporal window
- Vector describing moving direction and speed of each feature
- Relative depth value for each feature calculated using Algorithm 1 based on image sharpness

Table 4.1: List of parameters stored by the tracking system and experiment user interface that were used in the study. This list includes an example record and description of parameters.

Parameter	Example	Description
$participant\_number$	20	1 to 26
$part$	1	1 or 2 – each participant performed the experiment twice
$record\_number$	82	counted by tracker and reset to 0 for each part
$frame\_number$	1696	reset to 0 for each part performed by a participant
$hand\_number$	0	0 for left and 1 for right
$action\_type$	grab	either "grab" or "release"
$time\_stamp$	Fri Jan 6 12:48:04 2012	end of temporal window, calculated by tracker
fps	9	number of frames per second being processed
The fields below were repeated 12 times – once for each frame		
$min\_rect.center.x$	495.546	X position of minimum enclosing rectangle
$min\_rect.center.y$	174.59	Y position of minimum enclosing rectangle
$min\_rect.size.width$	277.155	width of the minimum enclosing rectangle
$min\_rect.size.height$	119.426	height of the minimum enclosing rectangle
$min\_rect.angle$	-71.9958	the angle of rectangle between $-90°$ and $90°$
$min\_circle.center.x$	497.5	X position of the minimum enclosing circle
$min\_circle.center.y$	177.5	Y position of the minimum enclosing circle
$min\_circle.radius$	141	radius of the minimum enclosing circle
$mass.center.x$	501.903	X position of center of mass for the contour of the hand
$mass.center.y$	166.845	Y position of center of mass for the contour of the hand
$feature\_mean.x$	543.737	Average X coordinate of features
$feature\_mean.y$	92.0414	Average Y coordinate of features
$feature\_StdDev$	13.5157	Standard deviation of features
$num\_of\_features$	5	number of features between 0 and 5
The following fields were stored from experiment UI (user interface)		
$record\_number$	82	counted by experiment UI, should match the tracker record number
$datetime$	2012/01/06 12:48:04.143	time stamp of when action message was received by UI
$trial\_duration$	00:00:02.159	time taken since displaying an object until an action is performed
$action\_type$	grab	action type as expected by the experiment UI
$actual\_x\_position$	1188.375	X position of object or target as set by UI
$actual\_y\_position$	544.655	Y position of object or target as set by UI
$object\_size$	153	size of grab or release object: $20px$ to $500px$

I had included simple descriptors for the shape of hand image such as center of mass and minimum enclosing rectangle and circle, therefore I decided not to include spatial and central image moments to reduce redundancy in the models. Alternatively, one could use image moments instead of those parameters proposed in this study and I suspect they will obtain very similar results in terms of accuracy of models for predicting center of action. Location of feature mean combined with standard deviation of features sufficiently capture spacial properties of features in case of grab and release actions, therefore I did not include position and relative depth of individual features in the modeling phase.

Chapter 5

Results

In this section, I first describe my data preparation efforts. Next, I present models for locating the center of action and distinguishing left and right hand. Finally, I present some qualitative observations based on post-trial questions.

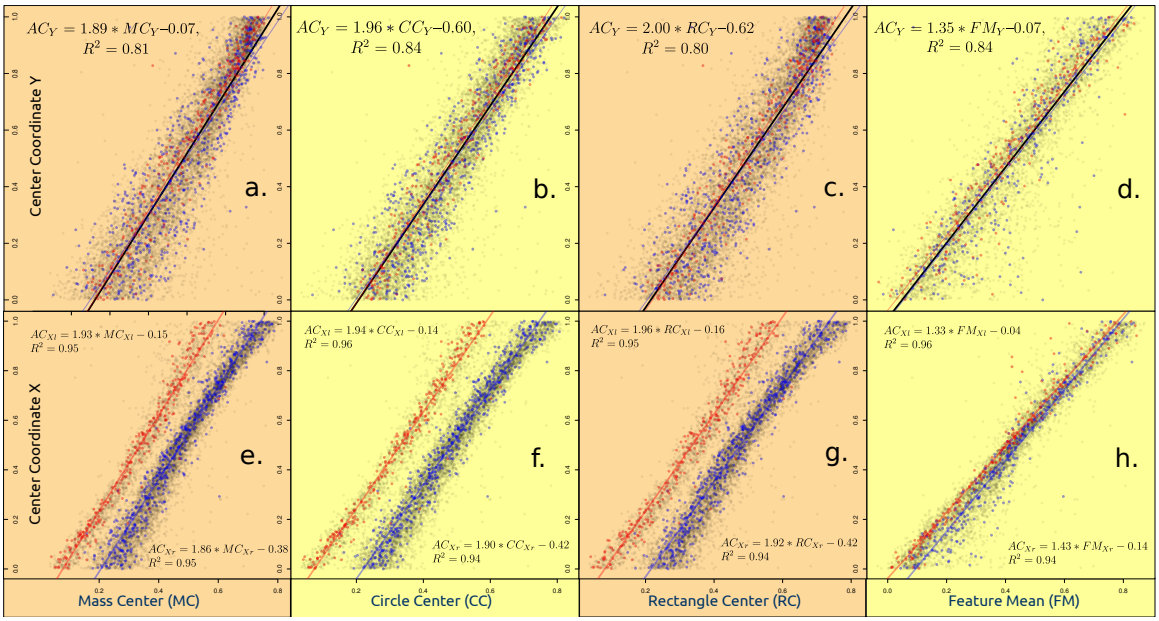

Figure 5.1: Participant data of four models to predict X and Y coordinates of the center of action.

5.1 Data Preparation

I filtered outliers and invalid records as follows. Data points outside the range $[quantile_1 - (1.5 * IQR), quantile_2 + (1.5 * IQR)]$ where IQR stands for Inter Quartile Range, were identified as outliers. Records were invalid when they did not contain valid data from the tracking software; e.g., feature mean would be invalid when no features were identified by the optical flow tracking algorithm.

Mean values described in the following sections are based on an average of the last three video frames for a particular trial. In other words, I reduced the dimensionality of the multiple video frames per trial into a single representative value. Prior to this study, I did not know an appropriate temporal length for describing grab and release actions. I conducted informal pilots to estimate that 12 consecutive video frames would be a very conservative upper bound for all potential participants. I recorded 12 consecutive frames prior to registering each action for all study trials. The tracking software performed at a minimum of 7 FPS (Frames Per Second) with both median and maximum rates of 9 FPS. Therefore, the length of temporal window in which the tracker collected its data was approximately 1.33 seconds. I created correlation matrices from 12 video frames aggregated across all participants, and then visually identified frames that contained the highest correlation between predictors and response values - in my case X and Y positions of center of action. Since the last three frames showed substantially higher correlations (above a threshold of 0.80), I reduced my data to contain only data from those three frames and thus reduced the length of the temporal window to 0.33 seconds. For each variable on each trial, I then take the mean value from those three frames.

5.2 Model for Locating the Center of Action

Center of action is the location on the 2D display where the user intends to center their grab or release interaction. I investigate four potential models for identifying the center of grab and release actions. Figure 5.1 captures a summary of the four linear regression models that I analyzed: (MC) mass center, (CC) circle center, (RC) rectangle center, and (FM) feature mean. For the top row of Y coordinate data, the black line represents the linear regression over the whole dataset. Blue and red points represent the subsets of data that were identified to be performed by the right and left hands, respectively. Hand side identification was performed by manually reviewing randomly selected sections of the study videos. For the

Table 5.1: ANCOVA results of hand side for each of the four models: Mean Center (MC), Circle Center (CC), Rectangle Center (RC), and Feature Mean (FM). $\Delta Slope$ and $\Delta Intercept$ are the absolute value of the difference in slope and intercept, respectively, of the left and right hand data for each model.

$H_0 : Model_{right} = Model_{left}, \alpha = 0.01$				
Model	F value	P value	$\Delta Slope$	$\Delta Intercept$
MC_X	6.8984	0.00878	0.07	0.23
CC_X	12.355	0.00046	0.04	0.28
RC_X	11.927	0.00058	0.01	0.26
FM_X	16.649	5.158e-05	0.10	0.10

bottom row of X coordinate data, blue and red lines represent linear regressions over subsets of data that were identified to have been performed by right and left hands, respectively. All models are normalized plots of the Y and X coordinates for a hand shape model against the Y and X coordinates of the actual object or target. The Y and X coordinates were normalized by dividing them by height and width of the screen respectively to obtain a value between 0 and 1. Plots a-d illustrate the unimodal distributions of data points for the Y coordinate results of each model, whereas plots e-h illustrate the bimodal distributions of data points for X coordinates, corresponding to the two hands. This result suggests that a system for detecting near touch actions needs to identify which hand is employed by the user.

5.2.1 Effect of Hand Side

I use analysis of covariance (ANCOVA) to compare left and right hand regression lines. I analyze this effect only on the X coordinate because I observed a very small difference between the left and right hands along the vertical (Y coordinate) axis, which in my view is currently of little practical value. However, the horizontal difference between the models is large, especially the distance between intercepts for left and right regression models. For each of the four models, I consider the null hypothesis that there is no difference between regression lines for the left and the right hand. Table 1 presents the ANCOVA results of hand side (HS) for each of the four models.

I observed statistically significant p-values to a level of $p < .01$ in all four cases, so I can reject the null hypothesis in all four models. Thus, hand side is a significant variable for the

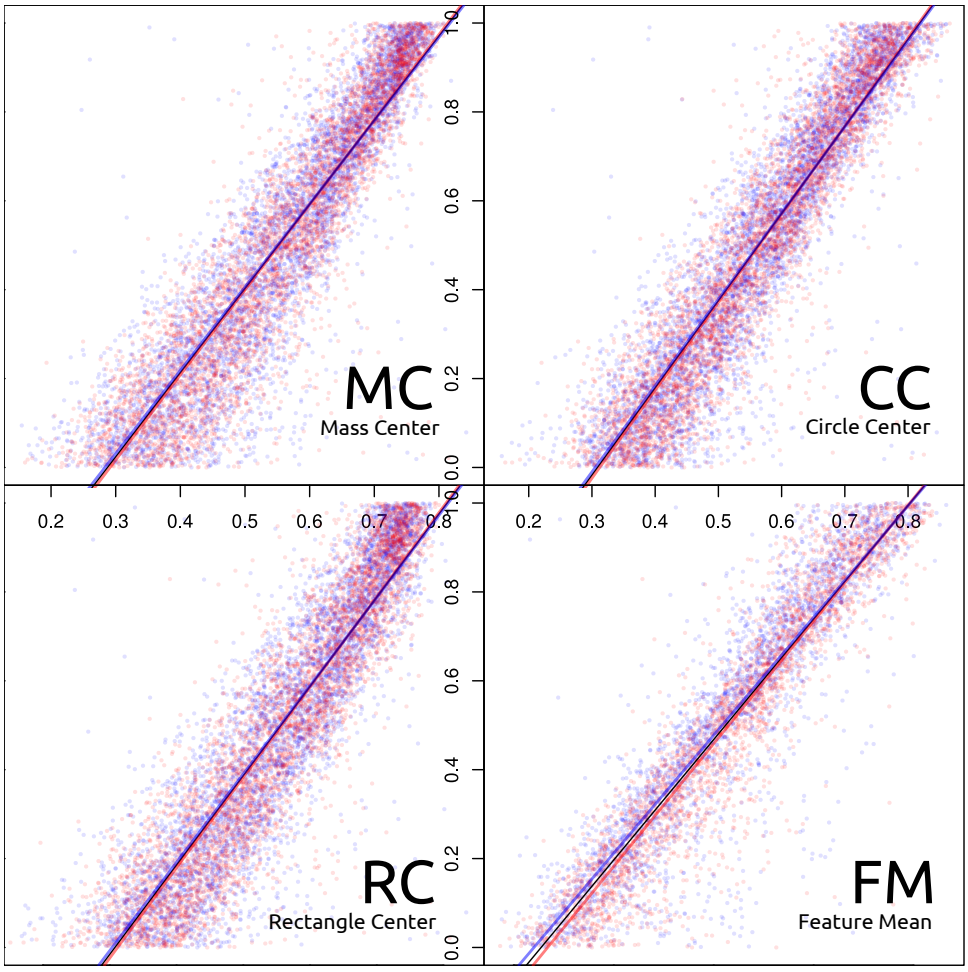

Figure 5.2: Influence of action type on the regression models. Red lines show the regression model only for the grab action, blue lines are for release action, and black lines are based on the whole dataset.

X coordinate of a user's center of action. Table 1 illustrates particularly large differences between left and right hand intercepts for the first three models MC, CC, and RC. For the FM model, the slope is larger and intercept difference is smaller compared to the other three models.

5.2.2 Effect of Object Size

Object size was not observed to have a practical effect on the models and was not analyzed further.

Table 5.2: Grab and release action type differences for the Y component of center of action

$H_0 : Model_{grab} = Model_{release}, \alpha = 0.01$		
	Model	P value
Left + Right	$MC : AT$	0.0264
	$CC : AT$	0.0274
	$RC : AT$	0.0269
	$FM : AT$	0.0006

5.2.3 Effect of Action Type

I also tested whether the type of action (grab versus release) performed by the user was a defining factor in determining the center of action. Figure 5.2 visualizes the alternative regression models considering the action type as a covariate factor for all four proposed models for center of action. As shown in the Figure, there is very little difference between the first three models, whereas the slope of the two regression lines are noticeably different for the feature mean (FM) model. For each of the four models in Figure 5.2, I consider the null hypothesis $H_0 : Model_{grab} = Model_{release}, \alpha = 0.01$. Similar to effect of hand side, I use analysis of covariance (ANCOVA) to compare regression lines for grab and release actions.

For models of the Y component of center of action, ANCOVA tests are listed in Table 2. For the Feature Mean (FM) model, I can reject the null hypothesis and conclude that action type is a covariate factor which that has a significant effect on the rate at which Feature Mean predicts actual center coordinate. I also observe that at an $\alpha = 0.05$ threshold, $p < .05$ for the other three models.

For the X component, I first divided the data based on hand side, since it has a known effect. I then examined the difference introduced by the factor of action type which has two possible levels – grab and release. The results are summarized in Table 3.

For the MC and RC models, statistically significant results were observed, $p < .01$, between grab and release actions. However, when considering $\Delta Slope$ and $\Delta Intercept$ one can see that this difference is relatively small and likely has little practical implication. In the case of FM this difference is nearly zero, which is not surprising since grab and release actions are temporally reverse of each other. For this reason I decided to exclude action

Table 5.3: Grab and release action type differences for the X component of center of action

$H_0 : Model_{grab} = Model_{release}, \alpha = 0.01$				
	Model	P value	$\Delta Slope$	$\Delta Intercept$
Right	$MC : AT$	0.0020	0.11	0.06
	$CC : AT$	0.0104	0.10	0.05
	$RC : AT$	0.0070	0.10	0.06
	$FM : AT$	0.8785	0.00	0.00
Left	$MC : AT$	0.0069	0.14	0.04
	$CC : AT$	0.0191	0.12	0.04
	$RC : AT$	0.0065	0.15	0.05
	$FM : AT$	0.8180	0.00	0.00

type as a factor in my model for the center of grab and release actions. Algorithm 2 shows a selection of the potential models that performed best in this study. Mass center (MC) and feature mean (FM) are used to model center of action (AC).

Algorithm 2 A method for detecting the center of action.

$$AC_x = \begin{cases} 1.93 * MC_x - 0.15 & R^2 = 0.95, \quad if\ left\ hand \\ 1.86 * MC_x - 0.38 & R^2 = 0.95, \quad if\ right\ hand \end{cases}$$
$$AC_y = 1.35 * FM_y - 0.07 \;\; R^2 = 0.84$$

5.3 A Method for Distinguishing Left and Right Hand

Distinguishing a user's left vs. right hand during interactions with a computer system is a difficult problem that impacts bimanual interaction techniques.

I now provide an approach to identify which hand was used to perform a grab or release action. This method exploits the fact that the angle of the minimum enclosing rectangle is dependent on the hand used (left or right) as well as the location of that hand. To visualize why this algorithm works, place your hands in their natural orientation in front of you and then place one hand on top of the other. While keeping both hands in their comfortable orientation, move them around in front of you. Notice that at no location does the most comfortable orientation of the two hands overlap. Algorithm 3 provides the pseudo-code.

Algorithm 3 An algorithm for detecting hand side when a grab or release action has been detected

```
FM.X                                   ▷ X component of feature mean
Θ                                      ▷ Angle of minimum enclosing rectangle
side ← right
if Θ < 120 − 100 ∗ FM.X then
    side ← left
end if
if Θ < 90 − 100 ∗ FM.X then
    side ← right
end if
if Θ < 45 − 100 ∗ FM.X then
    side ← left
end if
```

The tracking software currently does not use the direction of the hand to find the top of the minimum enclosing rectangle. The angle provided for the this rectangle is in the range

$(0° \ 90°]$. To evaluate the prediction of the hand side with my model, I took a subset of the data that was manually coded for hand side. From 401 points tested, 352 were correctly classified while 49 were misclassified; giving an accuracy of 88%. Figure 5.3 shows a visualization of how this algorithm performs. This detection rate is in line with previous hand distinction research. For example, Walther-Franks' [23] classifier approach achieved 80% accuracy. Zhang's [27] method achieved $91 - 92\%$ accuracy, but relied on the ability to identify the index finger.

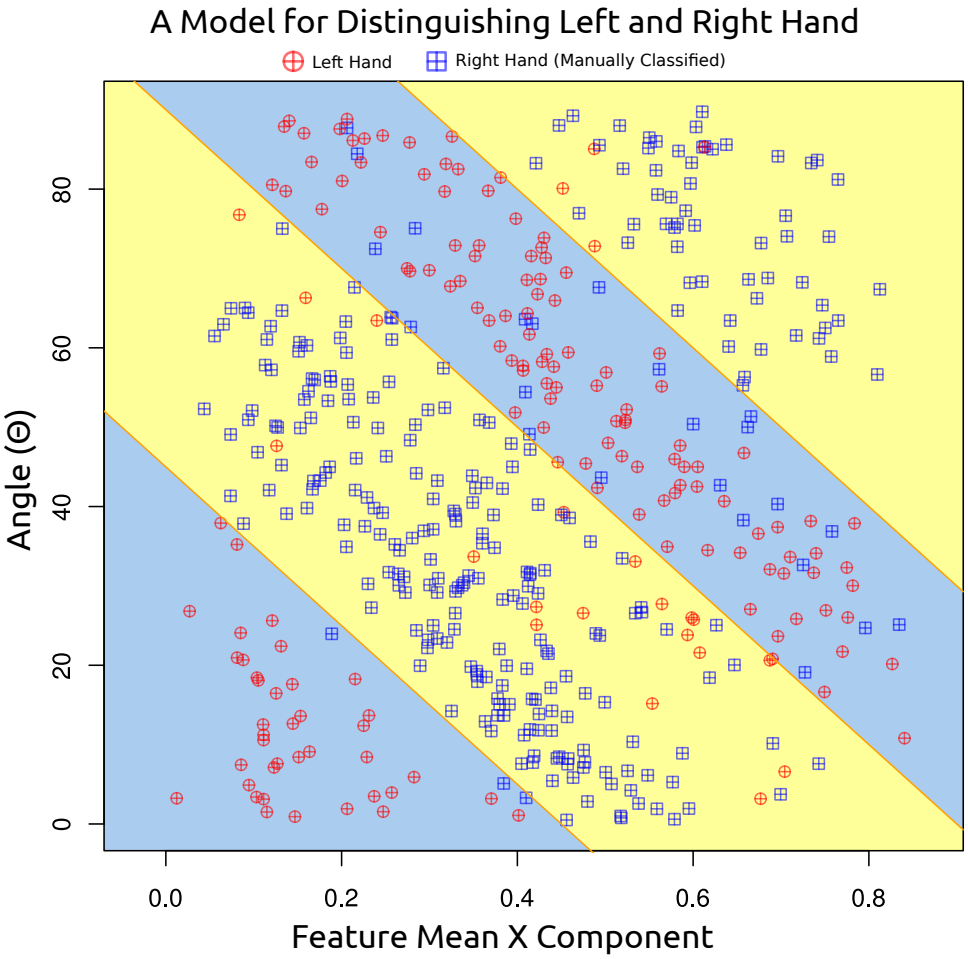

Figure 5.3: A method for distinguishing hand side at the time grab or release actions are identified. Red and blue markers represent actions that have been manually coded as left and right hand respectively, by a human observer after the study. Θ is the angle of minimum enclosing rectangle just before the action was registered by the system. The two regions shaded in blue represent the area classified as left hand by the model, while the region shaded in yellow represents the area that would be classified by the model as right hand. 88% of the points tested were classified correctly.

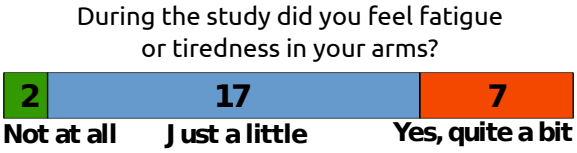

Figure 5.4: User feedback on how tired users felt after finishing the study.

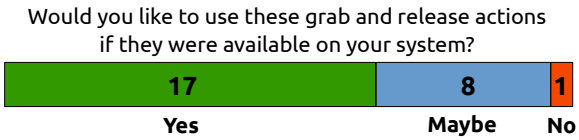

Figure 5.5: Users' interest in using grab and release action on their own computer systems.

5.4 Post Study Questionnaire Results

At the end of study, I provided a brief background questionnaire to the users. This questionnaire asked about fatigue, general acceptance of the actions, potential new actions, user interaction feedback, and general feedback. Figure 5.4 summarizes the responses from participants to the multiple choice question related to fatigue and figure 5.5 shows users' interest in using grab and release actions.

Chapter 6

Discussion

6.1 Classification of Near Touch Actions

The results of this study suggest that algorithms for classifying near touch actions need to be aware of which hand is used in performing the action. This can be either explicitly designed into the algorithm, or will be an implicit outcome if the algorithm for classifying the action and detecting its center is accurate enough. The feature matrix data collected in this study is available to anyone interested in training a classifier algorithm for detecting grab and release actions and the center of action (https://github.com/arasbm/NearTouchThesis).

6.2 Implications for Interaction Design

Grab and release actions for computing surfaces are simple to perform and remember because they closely resemble actions commonly performed with physical objects. Leveraging these actions can enrich the vocabulary of hand interactions on and above the surface of an interactive display. Near touch interaction can be complementary to existing touch interaction and my prototype system enables one to explore and evaluate ways of combining the two.

One potential pitfall to those designing games or other applications that require continuous interaction for extended periods of time is that grab, release, and other near touch interactions can cause user fatigue, as shown by the questionnaire results. This is likely also true of direct touch interactions on vertical surfaces. However, careful mixing of grab and release action into the vocabulary of interaction may be helpful as it allows users to change the posture of their hands, reducing repetitive strain. Additionally, I expect that

these actions would be most valuable for short duration walk-up-and-use scenarios, such as museum displays.

Another result of the current study that may be of interest to interaction designers is the model for distinguishing a user's left and right hand when a grab or release action has been performed. Understanding which hand is used can be useful for displaying contents such as menus in an appropriate location, and for supporting asymmetric bimanual interactions. My approach to hand detection is straightforward to implement, does not depend on finger identification, and maintains high accuracy.

6.3 Limitations of Findings

In this section I discuss the limitations of main results of this study. Perhaps the biggest limitation of this study is that it considers only two actions. The models for locating center of action are based on grab and release actions only. While these models are likely to hold true for several other actions that are similar to grab and release such as *rotate*, they will likely fail in other cases such as *point* action. Therefore these models can not be applied to other hand actions without further user testing.

In order to reduce complexity of study design some parameters were fixed. Participants were guided to sit on a chair, therefore fixing participant body position and orientation. To apply the algorithm for distinguishing left and right hand to the more general case, in which user can freely walk around the screen, body position and orientation would need to be considered in the algorithm. Screen size, height, and orientation were also fixed in this study and it is not known if changes in these parameters would have an effect on the models proposed in this study. I believe changes in head position relative to the hand will have minimal effect on the models. When screen orientation does not change, human hand-eye coordination can compensate for changes of had position relative to body. However the changes in the screen orientation will likely have a significant effect on the models because it may change the way users perceive the current position of an item as well as their perception of up direction. On the flat screen of a tabletop users will likely grab an item and lift it in the up direction. It is not clear how user would approach an item for grabbing and in what direction they will lift their hand after grabbing an item on vertical screen. The up direction may be perceived as the direction against gravity as is in the real world, or as the direction perpendicular to the surface of the display. More user testing is required to answer this question.

Another important limitation of this study is the use of one pair of object and target per

trial – both of which were circular. While I was able to show that changes in the size of the object did not have a significant effect on the way users perform grab or release actions on them, I can not assume a similar claim for changes in the shape of the object. For example if items presented on the screen resemble physical objects that afford different forms of grabbing (for example a hammer) or otherwise look significantly different from the representation used in this study – users may change the way they perform grab and release actions on the objects and as a result invalidate any of the models. Furthermore, placing several objects and targets on the same screen may prompt users to perform more accurate or possibly different forms of grab and release actions to avoid accidentally obtaining another object.

Chapter 7

Conclusions and Future Work

7.1 Future Work

The best models presented as Algorithms 1 and 2 can be used to design applications with near touch and bimanual interaction capabilities, which can be evaluated for usability in future studies.

In order to apply my results to a more general context in which the user can move around the display, one could study a user's position with respect to the interaction surface. One approach that has been used for localizing presence of users near an interactive surface is proximity sensing [1]. In this study I used a static angle of 60° and a fixed size display for the interface. Further research could measure the effects of changing the angle and size of the display for my presented models. In addition, future research should examine the effect of a crowded scene on gesture interactions, since this could change the way in which users grab and release objects.

Because grab and release actions did not produce a large difference in the proposed models, I excluded action type from the model for the center of action. For other action types that involve different patterns in the trajectory of features, action type may need to be considered as a factor. Therefore, future research could extend this model to cover other near touch actions besides grab and release. One could parameterize and detect other near touch actions such as flick, scoop, shake, throw, turn, twist, etc. I intend to focus my future research on these types of actions as opposed to symbolic gestures, because of their intuitiveness and cross cultural meanings.

7.2 Conclusions

I have presented empirical findings for parameterizing grab and release actions using a customized near touch technology. I used these findings to define a method for locating the center of grab and release actions as well as an approach for distinguishing which hand was used for the interaction. These empirical model data and near touch tracking system contributions provide new opportunities for natural and intuitive hand interactions with computing surfaces.

All data from my study is available to download for anyone interested in joining me to continue this research (https://github.com/arasbm/NearTouchThesis).

Bibliography

[1] M. Annett, T. Grossman, D. Wigdor, and G. Fitzmaurice. Medusa: a proximity-aware multi-touch tabletop. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 337–346. ACM, 2011.

[2] H. Benko, T.S. Saponas, D. Morris, and D. Tan. Enhancing input on and above the interactive surface with muscle sensing. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, pages 93–100. ACM, 2009.

[3] P. Brandl, C. Forlines, D. Wigdor, M. Haller, and C. Shen. Combining and measuring the benefits of bimanual pen and direct-touch interaction on horizontal interfaces. In *Proceedings of the working conference on Advanced visual interfaces*, pages 154–161. ACM, 2008.

[4] P. Brandl, M. Haller, M. Hurnaus, V. Lugmayr, J. Oberngruber, C. Oster, C. Schafleitner, and M. Billinghurst. An adaptable rear-projection screen using digital pens and hand gestures. In *Artificial Reality and Telexistence, 17th International Conference on*, pages 49–54. IEEE, 2007.

[5] Kivy Community. Kivy: Crossplatform framework for nui, 2012.

[6] C.T. Dang, M. Straub, and E. André. Hand distinction for multi-touch tabletop interaction. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, pages 101–108. ACM, 2009.

[7] T. de FO Araújo, A.M.N. Lima, and A.J.V. dos Santos. Detecting hands, fingers and blobs for multi-touch display applications. In *High Performance Computing & Simulation, 2009. HPCS'09. International Conference on*, pages 237–243. IEEE, 2009.

[8] KC Dohse, T. Dohse, J.D. Still, and D.J. Parkhurst. Enhancing multi-user interaction with multi-touch tabletop displays using hand tracking. In *Advances in Computer-*

Human Interaction, 2008 First International Conference on, pages 297–302. Ieee, 2008.

[9] Florian Echtler, Manuel Huber, and Gudrun Klinker. Shadow tracking on multi-touch tables. In *Proc. Advanced visual interfaces 2008*, pages 388–391, New York, NY, USA, 2008. ACM.

[10] J. Epps, S. Lichman, and M. Wu. A study of hand shape use in tabletop gesture interaction. In *CHI'06 extended abstracts on Human factors in computing systems*, pages 748–753. ACM, 2006.

[11] Y. Guiard. Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model. *Journal of motor behavior*, 19:486–517, 1987.

[12] J. Y. Han. Low-Cost Multi-Touch Sensing Through Frustrated Total Internal Reflection. In *Proc. UIST 2005*, pages 115–118, New York, NY, USA, 2005. ACM.

[13] O. Hilliges, S. Izadi, A.D. Wilson, S. Hodges, A. Garcia-Mendoza, and A. Butz. Interactions in the air: adding further depth to interactive tabletops. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, pages 139–148. ACM, 2009.

[14] M. Hirsch, D. Lanman, H. Holtzman, and R. Raskar. Bidi screen: a thin, depth-sensing lcd for 3d interaction using light fields. In *ACM Transactions on Graphics (TOG)*, volume 28, page 159. ACM, 2009.

[15] S. Hodges, S. Izadi, A. Butler, A. Rrustemi, and B. Buxton. Thinsight: versatile multi-touch sensing for thin form-factor displays. In *Proceedings of the 20th annual ACM symposium on User interface software and technology*, pages 259–268. ACM, 2007.

[16] C. Holz and P. Baudisch. Understanding touch. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, pages 2501–2510. ACM, 2011.

[17] A. Holzammer, U. Hahne, and M. Alexa. *Combining Diffuse Illumination and Frustrated Total Internal Reflection for touch detection*. PhD thesis, Masters Thesis, Technische Universität Berlin, Germany, 2009.

[18] S. Izadi, S. Hodges, S. Taylor, D. Rosenfeld, N. Villar, A. Butler, and J. Westhues. Going beyond the display: a surface technology with an electronically switchable diffuser. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*, pages 269–278. ACM, 2008.

[19] N. Marquardt, J. Kiemer, D. Ledo, S. Boring, and S. Greenberg. Designing user-, hand-, and handpart-aware tabletop interactions with the touchid toolkit. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, pages 21–30. ACM, 2011.

[20] Nima Motamedi. Hd touch: multi-touch and object sensing on a high definition lcd tv. In *Proc. Extended abstracts of CHI 2008*, pages 3069–3074. ACM, 2008.

[21] D. Pyryeskin, M. Hancock, and J. Hoey. Extending interactions into hoverspace using reflected light. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, pages 262–263. ACM, 2011.

[22] Y. Takeoka, T. Miyaki, and J. Rekimoto. Z-touch: an infrastructure for 3d gesture interaction in the proximity of tabletop surfaces. In *ACM International Conference on Interactive Tabletops and Surfaces*, pages 91–94. ACM, 2010.

[23] B. Walther-Franks, M. Herrlich, M. Aust, and R. Malaka. Left and right hand distinction for multi-touch displays. In *Smart Graphics*, pages 155–158. Springer, 2011.

[24] F. Wang and X. Ren. Empirical evaluation for finger input properties in multi-touch interaction. In *Proceedings of the 27th international conference on Human factors in computing systems*, pages 1063–1072. ACM, 2009.

[25] A.D. Wilson and H. Benko. Combining multiple depth cameras and projectors for interactions on, above and between surfaces. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 273–282. ACM, 2010.

[26] X.D. Yang, T. Grossman, P. Irani, and G. Fitzmaurice. Touchcuts and touchzoom: enhanced target selection for touch displays using finger proximity sensing. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, pages 2585–2594. ACM, 2011.

[27] H. Zhang. Evaluating finger orientation for position awareness on multi-touch tabletop systems. 2012.

Appendix A

Additional Information

A.1 List of Software Tools

R The R Project for Statistical Computing

OpenCV The open source computer vision library

Kivy An open source library for rapid development of applications. Available from kivy.org

Inkscape An open source scalable vector graphics editor. See inkscape.org

FFmpeg An open source library for recording and converting videos. See ffmpeg.org

Kile KDE integrated *LaTeX* environment. See kile.sourceforge.net

A.2 Background Questionnaire

Table A.1: The questionnaire that was completed by the participants before the study

Participant Number	007
Gender	Female Male
Age	
Name of City and Country where you grow up	
Do you like to work with your hands?	Yes Sometimes No
Do you perform regular activities that requires skilled hands? if so what are they?	
What hand do you write with?	Left Right Both
Which is your dominant hand in most daily activities?	Left Right Depends
If you use any touch devices please name a few of them here.	

A.3 Post Study Questionnaire

Table A.2: The questionnaire that was completed by the participants after the study

Would you like to use these grab and release actions if they were available on your system?	Yes Maybe No
Do you have any suggestions how grab and release actions can be used in an application?	
Can you think of other actions that can be detected near the screen? (If you like, you can perform your suggested actions in front of the screen so it can be recorded.)	
During the study did you feel fatigue or tiredness in your arms?	Not at all Just a little Yes, quiet a bit
You can provide any other comments or suggestions here.	