Master Thesis

Estimating the limitations of single-handed multi-touch input

Guido Soetens
Thesis no. ICA-3228568

September 1, 2012

INFOMGMT
Utrecht University

Supervisor:
drs. A. Kamphuis

Abstract

Multi-touch technology has become commonplace in most portable devices, like tablets and mobile phones. This technology has proven to be an interesting platform for games. Several games have been released which are considered *technology driven*, i.e., they were designed specifically for multi-touch devices. Some of these games require users to provide input using multiple fingers single-handed. In this case, the reachable area of a finger is subject to anatomical constraints. In this thesis, a model is proposed from which we can deduce the area that can be reached by a finger, given the fixed positions of other fingers that are already placed upon the planar surface. This model is created by measuring user input on a multi-touch device and to process these measurements using a variety of regression techniques. In this thesis, an estimation model is fitted in the case that one or two fingers are placed at fixed positions on the screen. These models can estimate the reachable area with reasonable accuracy. We also discuss how this model can be extended to serve the general case, where there are n stationary fingers on the screen.

Keywords: multi-touch, user input analysis, regression

Acknowledgements

First of all I would like to thank Arno Kamphuis for supervising this thesis project, and providing me with most helpful insights on this subject over coffee. Also, I would like to thank Headcandy for allowing me to make use of their multi-touch devices on which I could conduct my research. Finally, I owe a debt of graditude to everyone who were kind enough to take some time for my research, so that they could fill my database with user-input measurements. My sincere thanks to all of you!

Contents

1 Introduction

Multi-touch technology has been around for several decades, but it hasn't gained significant mainstream exposure until 2007 when the first iPhone got very popular. People were introduced with this new technology on a large scale, which led to multi-touch becoming the new standard in contemporary mobile devices and the introduction of tablets. These devices have proven to be a very succesful platform for gaming as well. For instance, Apple's official online application distribution service for iPad already features about 100.000 game titles [5]. Most of these games are not designed to be *technology driven*, since they only require users to provide subsequent tap- and drag-gestures with a single finger. However, several games have been released which make more use of multi-touch technology, and have been designed specifically for this platform.

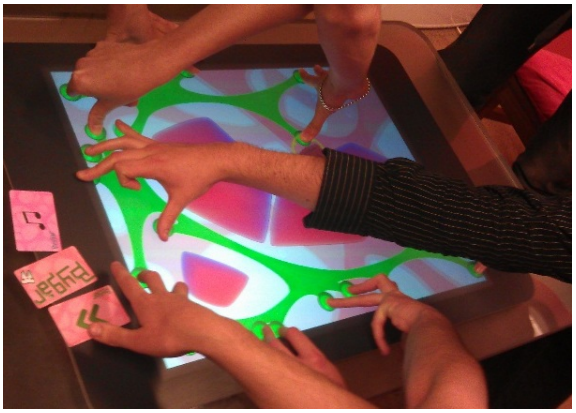


Figure 1.1: people playing Bygår on the Microsoft Surface v1.

Here are a few examples of games that are considered technology driven:

- **Bygår:** In Bygår, players must manipulate an elastic booger-network and guide it through a maze of obstacles. Players can extend the network by pulling at it with their fingers. As can be seen in Figure 1.1, people usually use multiple fingers from one hand to sustain the network. Bygår was created in 2010 as a student project at Utrecht University by Guido Soetens (the author).
- **TwisTouch:** TwisTouch is a one-player game in which you have to drag several orbs to their respective goals. Orbs can only be moved when all other orbs are touched as well and may not be released. The player must consider which finger must be assigned to which orb, since some goals are too far apart (on tablet devices) for some combination of fingers to reach. TwisTouch was created by Ivanovich Games.
- **Fingle:** In this game, players have to track several moving objects with their fingers. In order to do so, they will have to place multiple fingers on their touch-device and move them simultaniously. This mechanic cannot be replaced by single-mouse input or any other sort of contemporary pointing device, and is therefore considered to be specifically designed for the multi-touch platform. Fingle is created by the Dutch game company Game Oven.

In these games, players must place multiple fingers of one hand on the screen. However, no documented research has yet been conducted on how multiple input could be distributed single-handed on these devices. Such research could prove to be very useful in the development of these multi-touch technology driven games. For instance, this research could be used for random level generation by calculating a random (feasible) set of subsequent finger placements. Also, some combinations of required finger placements in a level design might not be physically possible for people with 'smaller' hands. By using our new proposed model, levels could be customized to a player by scaling it down to 'fit' the user's hands.

2 Related Work

In this section, we will discuss several subjects that are related to the research at hand. First, in Section 2.1, we will discuss two different kinds of common technologies used in multi-touch screens, and how user input is measured on these devices. Next, in Section 2.2, we will discuss what research has been conducted on biometrics of the human hand.

2.1 Common Technologies in Multi-Touch Screens

In this section, we present two multi-touch technologies that are most common in multi-touch devices. In Section 2.1.1, we discuss the capacitive touchscreen, which is most common in portable devices, such as mobile phones and tablets. Next, in Section 2.1.2, the vision-based touchscreen is discussed, which is present in larger multi-touch devices like the Microsoft Surface v1 and the Microsoft PixelSense.

2.1.1 Capacitive Multi-Touch Screens

The most common technology that is used for multi-touch screens in contemporary portable devices is the *capacitive* multi-touch screen. This screen contains a layer that holds an electrical charge. Underneath this layer, there is a two-dimensional network of capacitors, which are aligned along a grid. These capacitors monitor changes that occur in the layer's electrical charge. These changes arise when a conductive material, such as a finger, touches the screen. When the touch occurs, the grid segments around the finger respond and send the touch data as electrical impulses to the device's processor. The processor filters out the noise from the raw data and deduces the exact touch point. In Figure 2.1 it is shown how the exact input point is deduced from the electrical impulses in the iPhone processor [2].

The most high-end smartphones feature capacitive multi-touch technology (e.g, the Apple iPhone, HTC Desire, Samsung Galaxy S and the Nokia N8). The main benefits of using a capacitive touch-screen is that it supports multi-touch, it is very responsive and does not require pressure to be applied on the screen. Therefore, the durability of a capacitive screen is greater than pressure based touch-screens (i.e. *resistive* touch-screens). Also, the display of a capacitive device has better visibility in the sunlight (compared to a resistive touch-display). The downside of capacitive touch devices is that it only responds to conductive materials, such as a bare finger. Therefore - for instance - a capacitive touch device does not react to touches that are delivered by the tip of a fingernail, since it is not conductive.

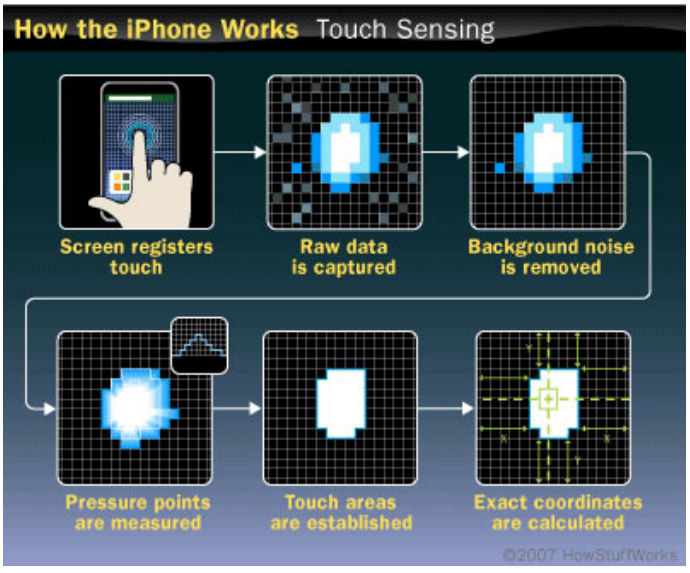
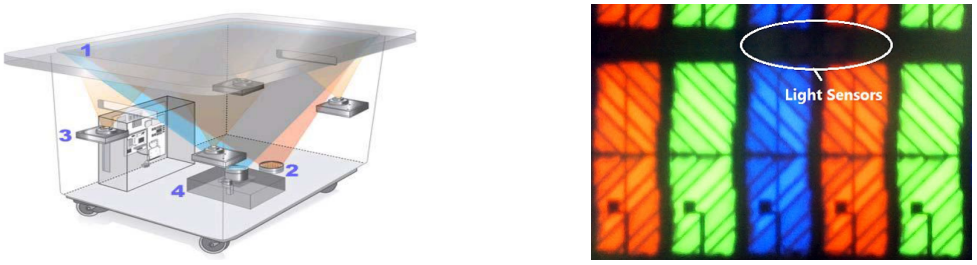


Figure 2.1: To determine the exact touch location, the processor first filters the noise from the raw input. The exact touch location is then deduced from the resulting touch area.

2.1.2 Visual Based Multi-Touch Screens

Larger multi-touch devices, like the Microsoft Surface v1 and the Microsoft Pixelsense, have a visual based multi-touch screen. The system of the Microsoft Surface v1 emits infrared light onto the touch-screen. An object that is placed upon the screen will reflect the infrared light back into the surface. The reflected light is captured by four infrared cameras that are placed inside the system. The structure of the Microsoft Surface is depicted in Figure 2.2a. The images that are captured by the IR cameras contain bright blobs, which are the areas of infrared light that was reflected at the surface contact-points. An example of what the captured IR image data looks like is shown in Figure 2.3a.



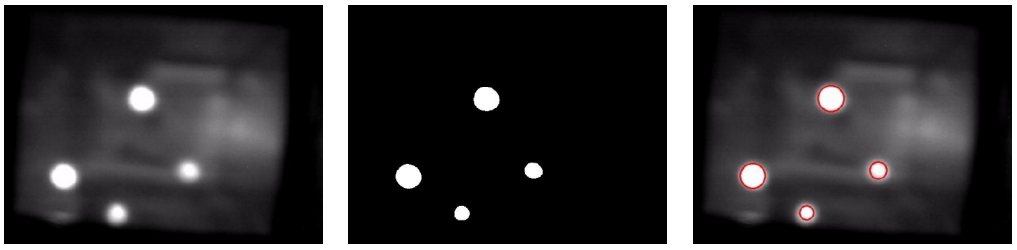
(a) The structure of the Microsoft Surface v1 [21]. 1 is the touchscreen, 2 is the IR emitter, 3 is an IR camera and 4 is the projector.

(b) The IR-sensor technology of the Microsoft PixelSense. The IR-sensors are embedded into the LCD layer of the display.

Figure 2.2: IR-sensing technology in the Microsoft Surface v1 and the Microsoft PixelSense.

The extraction of the touch-input locations from the captured image data is performed in three steps. First, the bright blobs are extracted by applying a threshold filter on the captured image. The threshold value is chosen such that the resulting binary (black-and-white) image only contains white pixels at the areas of the bright blobs. Some illumination noise from the environment can be filtered out at this stage by applying several subsequent erosion- and dilation-filters on the binary image. The result of this procedure is depicted in Figure 2.3b.

In the second step, the touch input pixels in the binary image are grouped into connected regions. Each region represents a separate touch input. The region extraction is depicted in Figure 2.3c. From these regions, the exact input points can be deduced. The input point is usually defined as the centroid of the associated blob region. A tracking algorithm is used to monitor the location of a touch input point over a series of subsequent frames. Sometimes, two adjacent fingers on the surface screen could result to a single large blob region. From the size of the blob and the information on previous frames from the tracking algorithm, it can be deduced that a blob is generated by multiple finger contacts on the display. In that case, multiple input points are sampled from the blob region.



(a) The original captured image from the IR cameras.

(b) The result after applying the threshold filter.

(c) The red circles indicate the connected regions.

Figure 2.3: The first two steps in extracting the touch input points.

In the last step, the sampled input points from the processed camera image must be mapped to the display's coordinate system. Assuming that the camera image is not subject to radial

distortion, the input points can be mapped from the image plane to the device's pixel coordinates by applying a *homographic* transformation on the points.

The successor to the Microsoft Surface v1 is the Microsoft PixelSense. The four IR-camera's of the Surface v1 have been replaced by a series of small IR-sensors. These sensors are located at each pixel of the PixelSense LCD display, which is shown in Figure 2.2b. In this case, the coordinate system of the captured image is equal to the display's coordinate system, which means that the homographic transformation does not need to be applied. Also, since the IR-sensing technology is embedded into the display, the weight and size of the device has been greatly reduced, which makes it possible to install the PixelSense on walls for instance.

As opposed to capacitive multi-touch technology, the visual based touch-screen does not require the input contact to be conductive. However, in the noise reduction step after the threshold filter has been applied to the captured infrared image, small blobs will be filtered out of the touch data. Since the tip of a finger-nail is also relatively small, the reflected light would be filtered out during the input processing since the resulting blob would also be relatively small.

2.2 Biometrics of the Human Hand and Hand Models

Biometrics is the science of measuring and analyzing biological data. In the analysis of the human hand, biometrics can be used to gather information about the lengths of fingers, the degrees of freedom of finger joints and the patterns in fingerprints for example. In the field of information technology, the term biometrics usually refers to the usage of these measurements for authentication purposes. In that case, the measurements are used as a discriminating factor between specific instances. However, for the research that is presented in this thesis, we will use biometrics to determine *general* properties of the human hand.

(a) The CyberGlove and its sensors.

(b) Gathering joint and bone properties from an image.

Figure 2.4: Inference of biometric data.

An example in which biometrics of the hand are used is the research in the field of Human Computer Interaction, and in particular the study of hand gestures [16, 19]. One way to measurue the properties of the hand is by using a *CyberGlove*, which is shown in Figure 2.4a. Various sensor technologies are placed along the surface of the glove, which are used to monitor biometric data such as the rotational properties of finger joints. Another way to gather biometric hand data is by processing and analyzing 2D images of the hand [19], as depicted in Figure 2.4b. The observed biometric data, like the rotation properties of joints, can be incorporated in a *kinematic* hand model. This model consists of joints with different degrees of freedom (DoF). Figure 2.5 shows this kinematic model and the types of joints in this model. The interphalangeal (IP) joint, the distal interphalangeal (DIP) joint and proximal interphalangeal (PIP) joint each have one DoF. The metacarpophalangeal (MCP) and trapeziometacarpal (TP) each have two DoF. The root joint, i.e. the wrist, has full translational and rotational motion in 3D space, so therefore has 6 DoF.

The rotations of these joints are subject to several types of constraints. One type of constraint is the *intra*-finger constraint, which is the range of rotations that are possible for each separate finger joint. For instace, the movable range of rotations of the index finger's MCP joint is approximately

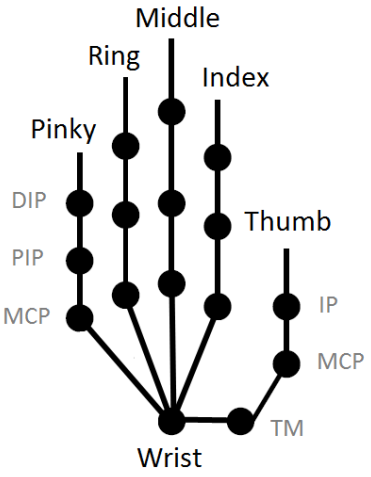


Figure 2.5: Kinematical hand model and joint notations.

$[-40°, 90°]$. Another type of constraint is the *inter*-finger constraint, which is the interdependence between subsequent joint rotations. For instance, on grasping, a DIP joint has a linear relationship with the previous PIP joint, which is defined as follows:

$$\theta_{DIP} = \frac{2}{3} \cdot \theta_{PIP}$$

Using the kinematic model and its observed constraints, the pose of the hand can be reconstructed with inverse kinematics from some key feature points, e.g. the locations of the fingertips. In Figure 2.6, a 3D hand model is reconstructed from feature points that are obtained by processing the image of a hand [19]. Note that the hand model has many DoF, so using inverse kinematics to reconstruct the 3D hand model can become a computationally intensive procedure.

Figure 2.6: Modelling a 3D hand model from feature points using Inverse Kinematics [20].

3 Research Goal and Motivation

As was mentioned in Section 1, the introduction of multi-touch technology in contemporary mobile devices has given rise to many multi-touch technology driven games. Some of these games require users to provide input with multiple fingers single-handed. A model from which possible placements of fingers can be deduced could be a very useful tool in the process of level design in the development of such multi-touch driven games. Also, such a model could be used to automatically generate random (feasible) levels that are based on multi-touch finger placements. In this thesis, a model is proposed with which the reachable area of a right-hand finger is estimated, given the fixed locations of other stationary fingers of the same hand. In Section 3.1, we will discuss which types of touch-input is taken into account in the proposed model, based on the information on different sorts of multi-touch technologies that were discussed in Section 2.1. Next, in Section 3.2, we will discuss how the estimation model will be constructed, as we reflect on the related research on biometrics that was discussed in Section 2.2.

3.1 Constraints on Provided Touch-Input

As was mentioned in Section 2.1, the tip of a fingernail does not trigger an input response on both capactive and visual based multi-touch devices. As opposed to capacitive touch-screens, a visual based touch-screen does react to the *surface* of a finger nail, since this would create a larger contact area. However, if multi-touch input would be provided with the surfaces of fingernails, the pose of the hand could become arguably uncomfortable, as is depicted in Figure 3.1. Also, using the surface of a fingernail to interact with controls on a multi-touch screen does not feel natural in general. Instead, we will only consider input that is provided with the 'skin'-side of a finger. Since skin is conductive, the model would also be applicable for capacitive multi-touch devices.

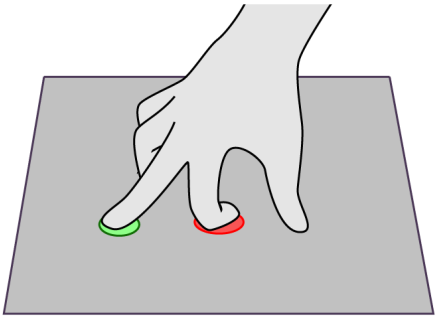


Figure 3.1: Input provided with the surface of a fingernail (the red contact area).

As was mentioned in Section 2.1.2, the exact input point that is deduced from a contact blob is usually defined as the centroid of the contact blob. Instead of placing the fingertip on the surface, someone can also place a larger part of the finger on the multi-touch device, which increases the contact area. In that case, the exact input point that is sampled from the contact blob is not located at the position of the fingertip as depicted in Figure 3.2a. Similarly as providing input with fingernails, the pose of the hand when a larger area of the finger is placed on the multi-touch device can become uncomfortable for most people. Given these considerations, the proposed model will only take touch-input that is provided with fingertips into account.

The size and shape of contact blobs are determined by the type of hardware that is used. Also, the inference method which determines the exact input point from these blobs may differ per device. If it is assumed that the inferred touch input location, relative to the surface contact (i.e. the fingertip), is invariant to the type of multi-touch device that is used, then the model that is proposed in this thesis is applicable for both capacitive and visual based multi-touch devices.

(a) A large contact area. **(b)** The input is provided with a fingertip.

Figure 3.2: Varying sizes of the contact area.

3.2 General Approach in Estimating the Reachable Area

As was mentioned in Section 2.2, a 3D reconstruction of a hand can be made from feature points using inverse kinematics. If the positions of the stationary fingers are known, then we could use these points as the feature points in the 3D reconstruction. One constraint in this reconstruction is that the entire 3D mesh is located above the surface. The model can then be manipulated to determine the reachable area of another finger, while taking the constraints on the kinematic hand joints into account. However, it was mentioned that using inverse kinematics can become computationally intensive, especially if this has to be performed at run-time on portable devices.

Instead, we will use a different approach to determine the reachable area of a finger. For each combination of stationary fingers - and varying positions of these stationary fingers - we will measure the reachable area of another finger (of the same hand) on a multi-touch surface. These measurements are performed over a group of people. By performing an analysis on the acquired measurements, we can determine some general properties of the shape and size of the reachable area under certain conditions. These conditions differ in the number of stationary fingers that are present on the surface and the distance between these stationary fingers for instance. By taking these analyses into account, we can define appropriate models to estimate the reachable areas by means of a variety of regression techniques. As the explanatory variable of the regression models we will use some intrinsic property of the user's hand. This value must have some significant explanatory power, so that the estimations of the regression models will 'fit' the actual reachable areas of the user appropriately. This explanatory variable will be obtained by means of a calibration procedure.

The main purpose of this model will be within the field of game- and level development. Therefore, the calibration procedure must preferably be quick and simple, so that it does not compromise the accessibility of the game itself. However, the brevity of acquiring user information must not compromise the accuracy of the (user specific) estimations. Therefore, we must find an appropriate equilibrium between the accessibility and accuracy of the proposed estimation model through the choice of an appropriate calibration variable.

4 Notations and Definitions

This section contains definitions of sets and notations that are used throughout this thesis.

Index:	Finger:
f_1	Thumb
f_2	Index Finger
f_3	Middle Finger
f_4	Ring Finger
f_5	Little Finger

We refer to each right-hand finger by assigning an index to each fingertip, as presented in the latter table. The set of these indices (i.e. fingertips) is called I. We will also define I_2 as the set which consists of all *tuples* of the elements in I:

$$I_2 = \{(f_a, f_b) : a < b \ \wedge \ f_a, f_b \in I\}$$

The finger tuple (f_a, f_b) is sometimes also written as (a, b).

5 Multi-touch Input Model for 2 Fingers

Suppose we want to place two fingers on a planar surface S. Of course, the first finger can be placed on any position $p \in S$. If this finger remains fixed at p, the second finger will have a limited reach. This limitation arises from the intrinsic properties of the hand, in particular the rotational limits of finger joints and the lengths of fingers. In this section, we will create a model from which we can deduce the area $A \subseteq S$, which denotes the area in which we can place the second finger, relative to the point p. In Section 5.1, we will give an analysis on the properties of the placement area A, and which variables determine its shape. Next, we will formulate how we can estimate these variables and create a general model in Section 5.2. In Section 5.3, it is discussed how the reachable range can be inferred from these regression models. Next, in Section 5.4, we will show how user-input has been measured and how these measurements are used to train our model. We test our model and reflect on the results in Section 5.5. Finally, in Section 5.6, we give a summary of how the proposed model can be applied.

5.1 Properties of The Placement Area

For each tuple $(f_a, f_b) \in I_2$, we want to determine the area A which can be reached by the second finger f_b, given that the first finger f_a is placed fixed at the point p. We will ignore those fingers that are not represented in this tuple. The *minimum* distance and the *maximum* distance between the fingers of tuple i are denoted as d_i^- and d_i^+ respectively. The range of distances with which these two fingers can be placed apart from each other is $[d_i^-, d_i^+]$. We will assume that the hand can be rotated around point p (that is, about the normal-vector of the surface). This rotation is valid, since it ensures that the fingertip of f_a will remain fixed at point p. Therefore, any point $q \in S$, which' distance to p is within the range $[d_i^-, d_i^+]$ is a point in A. Any point that does not meet this condition cannot be a point in A, since it would be out of reach for f_b. We can now give a formal definition of A:

$$A = \{q : q \in S \ \wedge \ \|p - q\| \in [d_i^-, d_i^+]\}$$

In geometric terms, the shape of area A is equal to an *annulus* with outer radius d_i^+ and inner radius d_i^-, which is depicted in Figure 5.1.

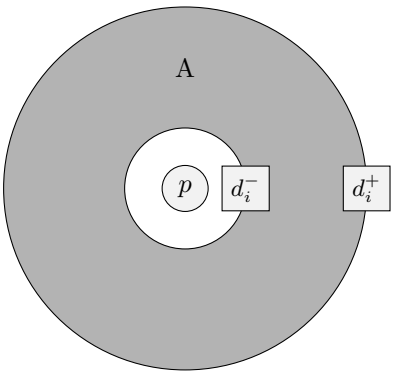


Figure 5.1: The gray area represents the reachable area A.

For each tuple $(f_a, f_b) \in I_2$, the tuple (f_b, f_a) is not contained in I_2. If we want to estimate A for (f_b, f_a), then we let the center of A (i.e. p) coincide with the position of f_b. The radii of the annulus is equal to the estimated radii for the tuple (f_a, f_b).

5.2 Estimation of The Stretch Range

As was mentioned in the previous section, we need to estimate the bounds of the range $[d_i^-, d_i^+]$ in order to obtain the area A. First, in Section 5.2.1, we describe the method with which the value for the maximum stretch d_i^+ is estimated. Next, the method to deduce the minimum distance d_i^- is presented in Section 5.2.2.

5.2.1 Estimation of The Maximum Stretch

In order to estimate the maximum distance between each tuple, we first need to measure these distances over a group of people. These measurements are contained in the dataset D^+. The elements in D^+ are denoted as $x_{n,i}$, which represents the maximum stretch measured between the finger tuple $i \in I_2$ during the measuring session $n \in 1, 2, ..., N$. Or, in short:

$$D^+ = \{x_{n,i} : n \in \{1, 2, ..., N\}, i \in I_2\}$$

The *marginal* variable x_i denotes the maximum stretch of tuple i, regardless of any session-index. The question remains how to generate a model from these measurements. A simple and straightforward approach would be to let the *expected* value of the maximum stretch, $\mathbb{E}[x_i]$, be the estimation of d_i^+ for each finger tuple i. This approach would yield a very crude estimation of the variable, since the maximum stretch is strongly determined by the intrinsic properties of hands, which may differ greatly between people.

Instead of marginalizing the data, we will consider one of these stretch-values x_c to be known *beforehand*, so that we can estimate the other stretches more accurately. The tuple c is called the *calibration* tuple. The other stretches x_i will be estimated by performing *linear regression*, where x_c will serve as the input variable. The linear model which estimates x_i is written as follows:

$$x_i = x_c \cdot w_i + \epsilon$$

In this model, ϵ is the error variable. Note that the latter linear equation only calculates a weighed value of x_c, and does not contain an intercept. This means that our regression line will go through the origin. Regression through the origin is possible, but it does not hold the same properties as 'regular' linear regression in which the fitted model contains an intercept [15, 6]. For instance, there is no trivial quality measure like R^2 for this regression model. Also, the following expression may *not* be true:

$$\mathbb{E}[\epsilon] = 0$$

We will need to define our own quality measure to determine if the obtained model is a good fit or not. We will test if a significant proportion of the estimation errors is within an acceptable boundary, which follows from the standard deviation of ϵ. This will be discussed more thoroughly in Section 5.5. The justification of using regression through the origin is as follows: if the maximum stretch between the calibration tuple c approaches zero, then the hand must be infinitesimally small. This means that the stretch between any other tuple i must also be approximately zero. If the intercept is denoted as a, the estimation can be written as follows:

$$0 = a + 0 \cdot w_i$$

This only holds if a equals zero, i.e. if the intercept is omitted. Now that the regression models for estimating the stretches are found, we need to find the appropriate values for the coefficients w_i. These values are gathered by minimizing the sum of *squared* estimation errors of the models. The sum of squared errors for estimating x_i is written as follows, from [13]:

$$E_{D^+}(w_i|c) = \sum_{n=1}^{N} \{x_{n,i} - w_i \cdot x_{n,c}\}^2$$

To find the values for w_i, which minimizes the latter equation, we must equate the derivative of E_{D^+} (w.r.t. w_i) to zero:

$$\begin{aligned}
\frac{\delta E_{D^+}}{\delta w_i} &= \sum_{n=1}^{N} 2 \cdot \{x_{n,i} - w_i \cdot x_{n,c}\} \cdot (-x_{n,c}) \\
&= -2 \cdot \sum_{n=1}^{N} \{x_{n,i} \cdot x_{n,c} - w_i \cdot x_{n,c}^2\} \\
&= 0
\end{aligned}$$

$\Rightarrow$ { divide both sides by -2, then separate terms into two summations }

$$\sum_{n=1}^{N}(x_{n,i} \cdot x_{n,c}) - w_i \cdot \sum_{n=1}^{N}(x_{n,c}^2) = 0$$

$\Rightarrow$ { bring $-w_i \cdot \sum_{n=1}^{N}(x_{n,c}^2)$ to the right, then divide both sides by $\sum_{n=1}^{N}(x_{n,c}^2)$ }

$$w_i = \frac{\sum_{n=1}^{N}(x_{n,i} \cdot x_{n,c})}{\sum_{n=1}^{N}(x_{n,c}^2)}$$

In Section 5.5, we will determine which value for c would yield the most accurate regression model, and reveal the optimal values for w_i.

5.2.2 Estimation of The Minimum Stretch

In Section 2.1 it was mentioned that input-points are sampled from contact blobs. It also states that there is a minimum size threshold for these contact blobs, since the input processing performs a noise reduction procedure that filters out small contact blobs. Hence, even though when two contact blobs are adjacent, the sampled input points do not coincide, so there is a (non-zero) minimum distance between two input points. We will perform *regular* linear regression to estimate the minimum distance, in which we introduce an intercept in the regression model. In the previous model, the known maximum stretch value of the calibration-tuple c is used as the input variable of the regression model, with which the other stretches are estimated. Since we do not want to burden people with too many calibration tasks, we will try to use the calibration-value that was required to estimate the *maximum* stretch as the input variable for the regression of the *minimum* stretch as well. We will reflect on this choice in Section 5.5 and see if this yields an accurate regression model. Also, we will give the optimal values for the weight-coefficients of the minimum-stretch regression model. This model can be written as follows:

$$x_i = w_{0,i} + w_{1,i} \cdot y_c + \epsilon$$

where y_c is the maximum stretch calibration value and x_i is the estimated minimum stretch for tuple $i \in I_2$. The weight coefficients are calculated as follows [13]:

$$w_{0,i} = \frac{\sum(x_{n,i}) - w_{1,i} \cdot \sum(y_{n,c})}{N}$$

$$w_{1,i} = \frac{N \cdot \sum(x_{n,i} \cdot y_{n,c}) - \sum(x_{n,i}) \cdot \sum(y_{n,c})}{N \cdot \sum(y_{n,c}^2) - (\sum y_{n,c})^2}$$

where $x_{n,i} \in D^-$ and $y_{n,c} \in D^+$. The summations iterate from $n = 1$ to $n = N$.

5.3 Calculating Reachability

To determine if point $q \in S$ lies within the annulus A, which is centered at p, we first test whether q is within the bounds of the *outer radius* of A. To do so, we determine the *probability* that the distance $d = \|p - q\|$ is smaller than the value of the outer radius d_i^+ (i.e. $P(d < d_i^+)$). Suppose we want to calculate this probability for tuple i, given the maximum stretch of the calibration tuple c: d_c^+. The estimation of d_i^+ can be deduced from our linear regression model. We assume that the distribution of the estimation error ϵ is a *normal* distribution. Therefore, the distribution of d_i^+ is as follows:

$$d_i^+ \sim \mathcal{N}(\mu, \sigma^2)$$

where:

$$\mu = w_i \cdot d_c^+ + \mathbb{E}[\epsilon|c, i]$$
$$\sigma^2 = var[\epsilon|c, i]$$

To determine $P(d < d_i^+)$, we have to calculate the area underneath the probability density function (p.d.f.) of the normal distribution, for which integration techniques have to be applied. The primitive of this p.d.f. is the *cumulative distribution function* (d.f.) of a normal distribution, which is usually denoted as $\Phi(x)$. This distribution function cannot be expressed in closed form in terms of elementary functions [14]. Therefore, we'd have to determine the values of the d.f. by numerical approximation. One of such approximations is given by Abramowitz en Stegun [11], which applies to *standard* normal distributions [1], and is defined as follows:

$$\Phi(x) \approx 1 - (a_1 t + a_2 t^2 + ... + a_5 t^5) e^{-x^2}, \text{ for } 0 \leq x \leq \infty$$

where $t = \frac{1}{1+px}$. The value of $\Phi(x)$ equals the probability $P(X < x)$, where the distribution of X is the standard normal distribution. The numerical approximations of the constant terms are shown in the following table:

p	0.3275911
a_1	0.254829592
a_2	-0.284496736
a_3	1.421413741
a_4	-1.453152027
a_5	1.061405429

The maximum approximation error of the latter function is $1.5 \cdot 10^{-7}$. Note that this function only yields correct approximations for $x \geq 0$. However, the probabilities of negative values can still be determined, since the standard normal distribution is symmetric with respect to the point $x = 0$ (since $\mu = 0$), which means that $\Phi(-x) = 1 - \Phi(x)$. From this symmetry property it also follows that $P(X > x) = P(X < -x)$. The random variable Z, which' distribution is the standard normal distribution, can be defined in terms of d_i^+ as follows:

$$Z = \frac{d_i^+ - \mu}{\sigma}$$

Where μ and σ are the parameters of the distribution of d_i^+. We can now calculate the probability that the distance d is smaller than the estimated maximum stretch value d_i^+:

$$\begin{aligned} P(d < d_i^+) &= P(\frac{d-\mu}{\sigma} < \frac{d_i^+ - \mu}{\sigma}) \\ &= P(x < Z) \qquad \text{//for brevity: } x = \frac{d-\mu}{\sigma} \\ &= P(Z < -x) = \begin{cases} \Phi(-x) & \text{if } -x \geq 0 \\ 1 - \Phi(x) & \text{otherwise} \end{cases} \end{aligned}$$

We can deduce the probability that d is larger than the *inner* radius d_i^- in a similar way. The probability is defined as follows for tuple i:

$$\begin{aligned} P(d > d_i^-) &= P(\frac{d-\mu}{\sigma} > \frac{d_i - \mu}{\sigma}) \\ &= P(x > Z) \\ &= P(Z < x) = \begin{cases} \Phi(x) & \text{if } x \geq 0 \\ 1 - \Phi(-x) & \text{otherwise} \end{cases} \end{aligned}$$

What is left to determine is the *simultaneous* probability of d being both larger than d_i^- and smaller than d_i^+. This probability is written as follows:

$$P(d_i^- < d < d_i^+) = P(d < d_i^+) \cdot P(d_i^- < d \mid d < d_i^+) \tag{1}$$

[1] The standard normal distribution: $X \sim \mathcal{N}(\mu = 0, \sigma^2 = 1)$

The right-hand side of the latter expression contains a *conditional* probability. To calculate this probability, we would have to re-define the distribution of d_i^-, so that it is only based on the subset of data that meets the latter condition. That is, this subset only contains measurements where the maximum stretch d_i^+ is *strictly* larger than d. However, we want to create a model that does not have to re-adress the data-set after it has been fitted. Therefore, we will further analyze the properties of the distributions of d_i^- and d_i^+.

First, let us denote $\texttt{Min}[d_i^+]$ as the *minimum* maximum distance that was measured in the entire dataset between tuple i. If $d < \texttt{Min}[d_i^+]$, then the condition $(d < d_i^+)$ holds for each measurement in the data-set. In this case, no elements from the data-set will be filtered out. The distribution of d_i^- will therefore remain unchanged, which is why the conditional probability can be approximated as follows:

$$P(d_i^- < d \mid d < d_i^+) \approx P(d_i^- < d), \text{ if } d < \texttt{Min}[d_i^+] \tag{2}$$

However, if $d \geq \texttt{Min}[d_i^+]$, then some elements of the data-set will not meet the latter condition. Therefore, the distribution of d_i^- will alter, and so does the probability that d is greater than d_i^-. We expect that this probability is always approximately one, regardless of whether the distribution of d_i^- is modelled using a filtered set of data or not. This follows from the expectation that the intersection of the sample spaces of d_i^- and d_i^+ is negligible. In that case, there must always be a sufficient amount of standard deviations between the estimated value of the minimum stretch and the distance value d, where $d \geq \texttt{Min}[d_i^+]$. The relation between d and the distribution of d_i^- is depicted in the Figure 5.2.

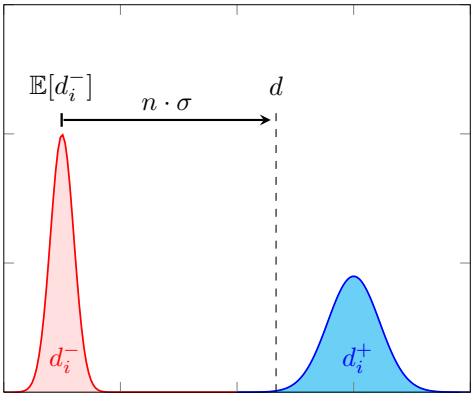


Figure 5.2: The number of standard deviations between $\mathbb{E}[d_i^-]$ and d.

In the image, n denotes the number of standard deviations between $\mathbb{E}[d_i^-]$ and d. If n is always sufficiently large (e.g. at least 4), then the probability that d is larger than d_i^- is always approximately one. Therefore, both probabilities are approximately equal. We will show that this is the case if $d \geq \texttt{Min}[d_i^+]$. For this purpose, the data-set will be filtered using every maximum stretch value in D^+ as the distance threshold d in the condition $(d < d_i^+)$. From this filtered set, we can generate a distribution of d_i^- using every calibration value in D^+. From this distribution and the distribution that was modelled using the entire data-set, we can deduce the difference between $P(d_i^- < d \mid d < d_i^+)$ and $P(d_i^- < d)$. A snippet of pseudo-code with which the *maximum* difference between these two probabilities can be deduced from our data-set is provided in Algorithm 1. The results of running this algorithm for each tuple i are given and discussed in Section 5.5.

If it follows from our results that the maximum difference between $P(d_i^- < d \mid d < d_i^+)$ and $P(d_i^- < d)$ is sufficiently small - given that $d \geq \texttt{Min}[d_i^+]$ - then we can approximate the conditional probability by disregarding its condition. Therefore, we can write:

$$P(d_i^- < d \mid d < d_i^+) \approx P(d_i^- < d), \text{ if } d \geq \texttt{Min}[d_i^+] \tag{3}$$

By combining the equations (2) and (3), we get:

$$P(d_i^- < d \mid d < d_i^+) \approx P(d_i^- < d)$$

Therefore, the simultaneous probability in (1) can be approximated as follows:

$$\begin{aligned} P(d_i^- < d < d_i^+) &= P(d < d_i^+) \cdot P(d_i^- < d \mid d < d_i^+) \\ &\approx P(d < d_i^+) \cdot P(d_i^- < d) \end{aligned}$$

Algorithm 1 Find the maximum difference (ϵ) between $P(d_i^- < d)$ and $P(d_i^- < d \mid d < d_i^+)$, given that $d \geq \texttt{Min}[d_i^+]$

Require: $i \in I_2$, $c \in I_2$, D^-, D^+
1: $\epsilon \leftarrow 0$
2: $D_i^+ \leftarrow \{x_{n,i} : x_{n,i} \in D^+\}$
3: $D_i^- \leftarrow \{x_{n,i} : x_{n,i} \in D^-\}$
4: $\mathcal{M} \leftarrow$ RegressionModel(D_i^-)
5: $\sigma \leftarrow$ EstimationErrorDeviation($\mathcal{M}$)
6: SortIncreasing(D_i^+)
7: **while** SizeOf(D_i^+) > 0 **do**
8: $d \leftarrow$ First(D_i^+)
9: $D_i^- \leftarrow \{x_{n,i} : x_{n,i} \in D^- \wedge y_{n,i} \in D_i^+ \wedge y_{n,i} > d\}$ {the filtered set of minimum stretches}
10: **if** SizeOf(D_i^-) > 0 **then**
11: $\mathcal{M}' \leftarrow$ RegressionModel(D_i^-)
12: $\sigma' \leftarrow$ EstimationErrorDeviation($\mathcal{M}'$)
13: **for all** $x_{n,c} \in D^+$ **do**
14: $d_i^- \leftarrow \mathcal{M}(x_{n,c})$ {estimate the minimum stretch for person n}
15: $d_i^{-\prime} \leftarrow \mathcal{M}'(x_{n,c})$
16: $\epsilon \leftarrow$ Maximum(ϵ,$|(1 - \Phi(\frac{d-d_i^-}{\sigma})) - (1 - \Phi(\frac{d-d_i^{-\prime}}{\sigma'}))|$)
17: **end for**
18: **end if**
19: RemoveFirst(D_i^+)
20: **end while**
21: **print** ϵ

5.4 Implementation and Measuring

The measuring of the minimum and maximum input distances were performed on a Microsoft PixelSense. This device has a 30 inch (76 cm) 4:3 multi-touch display, which ensures us that there is enough space for the measurements to be performed. Only the stretches of the *right* hand were measured. The applications were developed using WPF (.NET 4.0 framework). The minimum and maximum stretches were measured for each finger tuple $(f_a, f_b) \in I_2$. A procedure of measuring a stretch is as follows for tuple (f_a, f_b):

- the user is asked to place f_a on the center location of the screen (on a button control).
- next, the user is asked to place the second finger f_b on the screen and move it as far as possible from f_a or as close as possible to f_a (when measuring the maximum stretch or minimum stretch respectively). One condition is that the positioning of the hand may not feel uncomfortable at any point of the measurement. After f_b has been placed on the screen, other fingers (besides f_a and f_b) may also be placed on the screen for support.
- when the minimum or maximum stretch is reached, a button is pressed by the observer, so that the measurement can be performed for the next tuple.

Figure 5.3 shows how the maximum distance between the fingers f_2 and f_5 is measured. The outer circle of annulus A is also drawn during the measurement.

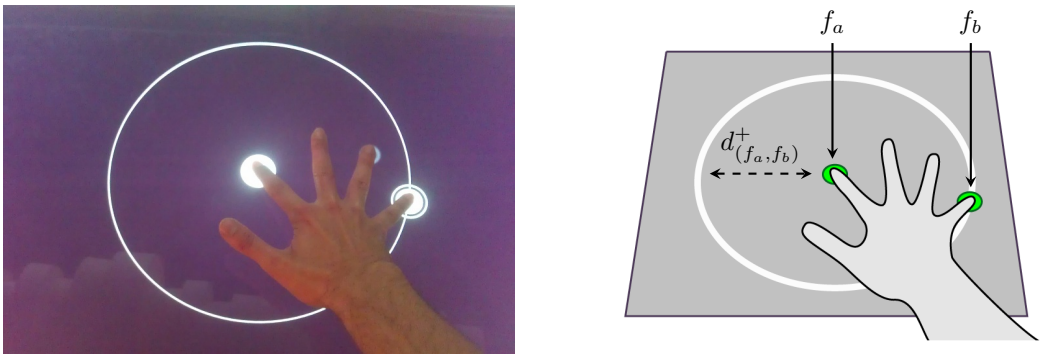


Figure 5.3: Measuring the maximum distance between f_2 (the index finger) and f_5 (the little finger)

5.5 Results

In this section, we will discuss the results of the minimum and maximum stretch measurements. First, in Section 5.5.1, we will show the regression models that have been fitted using the gathered data and the estimation error properties of these models. Next, we deduce which tuple is the best choice for the calibration purpose in Section 5.5.2. Finally, in Section 5.5.3, we will reflect on whether the conditions that allow the probability approximations from Section 5.3 are met.

5.5.1 Fitted Regression Models

At the end of this document, there is a list of appendices. In Appendix A.1, the weight coefficients of the maximum stretch regression model are given. A separate model is fitted for each possible calibration tuple $c \in I_2$. In Appendix A.2, the first table contains the expected estimation error for each of these regression models ($\mathbb{E}[\epsilon|c, i]$). The second table contains the *general* estimation error over all tuples $i \in I_2$ for each separate calibration tuple, i.e. $\mathbb{E}[\epsilon|c]$. Next, in Appendix A.3, the two tables represent the local and global estimation error variances respectively. In total, 36 people have contributed to the data-set on which these models were fitted.

Similarly, the properties of the minimum stretch regression model are provided in Appendix B. Note that Appendix B does not contain a section with expected prediction error values, since this value is always zero for this particular regression model. In Appendix B.3, we have shown the estimation values of a *plain* estimation model, where no regression is performed. The value printed beneath these two tables represents the global error variance of this estimation model.

5.5.2 The Optimal Calibration Tuple

In the maximum stretch models, the model that is based on the calibration tuple $c = (f_1, f_5)$ yields the most accurate estimations. In the tables in Appendix A.2, we can see that this calibration tuple yields the lowest overall expected prediction errors. As for the overall estimation error variance, we can see from the second table in Appendix A.3 that the general error variance by using $c = (f_1, f_5)$ is the lowest. In Figure 5.4, a normal distribution is plotted where the variance equals the general prediction error variance when using $c = (f_1, f_5)$. The area within the domain $[\mu - 2\sigma, \infty)$ is shaded in pink and covers 97.8% of the total disribution area. This means that in general, if we subtract approximately 2 cm from the estimated maximum distance, then 97.8% of all people will be able to reach this distance. Of course, the variance values in the first table in Appendix A.3 are used to get a more accurate estimation for each specific tuple $i \in I_2$.

Similarly, in Appendix B, we can see that choosing $c = (f_1, f_5)$ yields the most accurate estimation of the minimum stretch distance as well. However, we can see that the prediction error variance does not differ greatly between all possible calibration tuples. The reason for this might be that a maximum stretch value is not strongly co-related to the minimum stretch distances, and therefore has very little influence in the estimation of the minimum stretch. Therefore, we might consider omitting the regression procedure in the estimation of the minimum stretch by performing

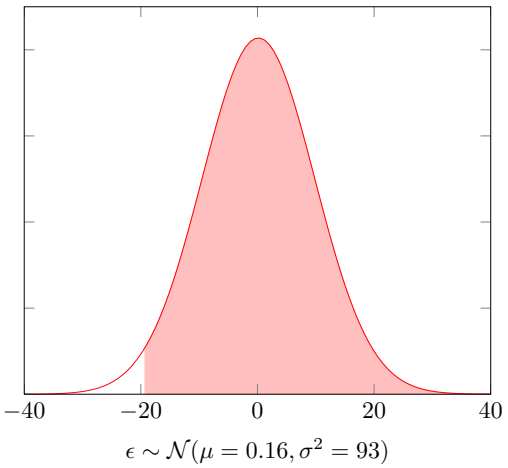


Figure 5.4: The general estimation error distribution of the maximum distance regression model. The area in the domain $[\mu - 2\sigma, \infty)$ is shaded pink. The metrics are in millimeters.

a plain estimation of the minimum distance between each tuple. The values of these estimations are given in the first table in Appendix B.3. Even though performing regression yields lower prediction error variances, we can see that the global variances of the regression models and the global variance using plain estimation differ very little. The difference between the global variance of the regression model using $c = (f_1, f_5)$ and the global variance using plain estimation is 1.5 squared millimeters. We consider this value small enough to choose the plain estimation over the usage of the regression models in the estimation of the minimum stretch, i.e. using the regression model would not yield much more significant accuracy in the prediction of the minimum distance. The general variance when using plain estimation is 13.1 squared millimeters. This means that in general, if we add 7.2 $(= 2 \cdot \sqrt{13.1})$ millimeters to the estimated minimimum stretch, then we can assume that actual minimum stretch will be smaller than this value for 97.8% of the population.

5.5.3 Justification of the Probability Approximations

We will now indicate that we can perform the probability approximations as was discussed in Section 5.3. The table in Appendix B.4 shows the results of running the code in Algorithm 1 for each tuple in I_2. These values were measured in the case that we estimate d_i^- by means of regression (in the left column) and by plain estimation (in the right column). From this table, it follows that the maximum difference between the conditional and the unconditional probability is very small in all cases. Therefore, we consider the approximation of the conditional probability by disregarding its condition to be accurate.

5.6 Applying the Proposed Estimation Model

In this section, we summarize how the proposed model is applied when we want to determine the probability that points can be reached by f_b, given that f_a is located fixed at point p. In order to use the regression models, we first require the calibration value c, which is the maximum distance between the thumb (f_1) and little finger (f_5) of the user on whom the model is fitted. This value is acquired by means of a calibration procedure.

The reachable area of f_b is defined as an annulus, where the inner and outer radius are the estimated minimum distance $d^-_{(a,b)}$ and maximum distance $d^+_{(a,b)}$ respectively. The minimum distance is estimated by means of a plain estimation. The expected minimum distance μ^- and the estimation error variance $(\sigma^-)^2$ are given in Appendix B.3 for each tuple $(f_a, f_b) \in I_2$. Note that the metrics of the standard deviation is in millimeters. Therefore, the metrics of the calibration tuple or the standard deviation must be converted, depending on which metrics are preferred in the output of the model.

To estimate the maximum distance $d^+_{(a,b)}$, we use the regression model that was defined in Section 5.2.1. The coefficients that are used in this estimation are the ones in the fourth row of the table presented in Appendix A.1 (i.e. the row of the calibration tuple $(f_1, f_5) \in I_2$). To estimate the maximum distance between the fingers f_a and f_b, we multiply the calibration value c with the regression coefficient $w_{(a,b)}$. The estimated maximum distance is denoted as μ^+. Since the expected estimation error is a small fraction of a millimeter, we will not take it into account in the estimation. The estimation error variance of this regression is presented in Appendix A.3, and is denoted as $(\sigma^+)^2$.

Using the distribution properties of $d^-_{(a,b)}$ and $d^+_{(a,b)}$, we can now calculate the probability that point q can be reached by finger f_b. First, we let d denote the distance between q and point p (where f_a is located). The probability that q can be reached by f_b is equal to the probability that q is larger than the minimum distance $d^-_{(a,b)}$, and smaller than the maximum distance $d^+_{(a,b)}$. This probability is written as follows:

$$P(d^-_{(a,b)} < d < d^+_{(a,b)})$$

In Section 5.3, it was shown that this simultaneous probability can be approximated as follows:

$$P(d^-_{(a,b)} < d < d^+_{(a,b)}) \approx P(d^-_{(a,b)} < d) \cdot P(d < d^+_{(a,b)})$$

To calculate the probability $P(d^-_{(a,b)} < d)$, we first transform the distance variable d, such that the transformation would make the distribution of $d^-_{(a,b)}$ a standard normal distribution. The transformation is written as follows:

$$d^- = \frac{d - \mu^-}{\sigma^-}$$

If $d^- \geq 0$, then the probability that d is larger than the minimum distance equals $\Phi(d^-)$. Otherwise, this probability equals $1 - \Phi(-d^-)$. Similarly, to calculate the probability $P(d < d^+_{(a,b)})$, we transform the variable d, such that the transformation would make the distribution of $d^+_{(a,b)}$ a standard normal distribution. This transformation is written as follows:

$$d^+ = \frac{d - \mu^+}{\sigma^+}$$

If $d^+ \leq 0$, then the probability that d is smaller than the maximum distance equals $\Phi(-d^+)$. Otherwise, this probability equals $1 - \Phi(d^+)$. The simultaneous probability is approximated by multiplying the two separate probabilities.

6 Multi-Touch Input Model for 3 Fingers

In this section, we wish to create a model from which we can deduce the area that can be reached by finger f_c on a planar surface, given that two other fingers - $(f_a, f_b) \in I_2$ - are already placed at fixed positions on this surface. Since the minimum distance between two fingers cannot be zero, the locations of f_a and f_b are different. Contrary to the model that was described in the previous section, we are not able to rotate the hand freely around any point on the surface, since any rotation would cause either f_a or f_b to move away from it's (fixed) location. This also means that the reachable area is not simply the intersection of the two annuli that are given by the previous model of the tuples (f_a, f_c) and (f_b, f_c), since the previous model was created from the assumption that the hand can move freely around some pivot point.

In Section 6.1, we describe how we estimate the area that can be reached by f_c, and which parameters define this area. This estimated area will be denoted as A. Next, in Section 6.2, a model is defined on how the area A is estimated for finger f_c, given the fixed locations of f_a and f_b. In Section 6.3, we calculate the probability that a point can be reached by f_c according to the proposed model. A method with which we indicate the quality of our model is proposed in Section 6.4, followed by a report on how the hands of several people were measured and what software was written for this purpose in Section 6.5. In Section 6.6 we present the results of the fitted model that is based on these measurements. Finally, in Section 6.7, we give a summary of how the proposed model can be applied.

6.1 Properties of the Placement Area

For each finger f_c and finger tuple $(f_a, f_b) \in I_2$, we want to determine the area A which can be reached by f_c, given that the fingers f_a and f_b are placed on fixed positions. We will disregard whether other fingers of the same hand are also placed on the surface. We have deduced some general characteristics of the reachable area by means of empirical research, in which we let a number of people 'draw' the reachable area with finger f_c on a multi-touch device, while two other fingers f_a and f_b are placed at fixed positions on the screen. These drawings were made for all combinations of fingers and varying distances between the stationary fingers f_a and f_b. One condition of the placement of f_c is that the placement does not become uncomfortable at any point of the drawing process. One observation is that the drawn area A is convex, except for the odd case when the drawn area is close to any of the stationary fingers f_a or f_b. In that case, we disregard the points which are closer to the stationary finger than the minimum distance between f_c and the stationary finger would permit. This makes the area concave, but still most areas that were observed remain convex.

Another observation is that the area A is invariant to any *proper rigid transformation* [2] with relation to the locations of f_a and f_b. Therefore, we will let the location and orientation of the world frame F_w be relative to f_a and f_b. The location of F_w is placed at the mean location of f_a and f_b. The x-axis of F_w points to f_b. The properties of F_w are depicted in Figure 6.1.

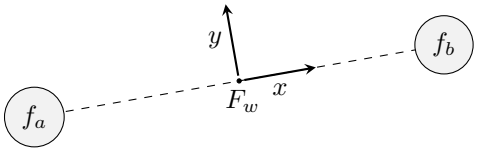


Figure 6.1: The properties of the world frame F_w.

We want to approximate the reachable area with a shape that can be described with a finite number of parameters. These parameters can then be estimated by means of regression, where the distance between f_a and f_b will serve as the input variable. From preliminary observations on the shape of the reachable areas we have gathered that an ellipse approximates the reachable area quite well. An ellipse is defined by 5 parameters. Two parameters, r_x and r_y, define the two axes of the ellipse. We let the variable φ denote the angle of axis r_x relative to the x-axis of the

[2] A proper rigid transformation only involves translations and rotations

world frame. Finally, the location of the ellipse center is denoted by the polar coordinate (θ, d). A depiction of the ellipse, its parameters and its relation to F_w is given in Figure 6.2.

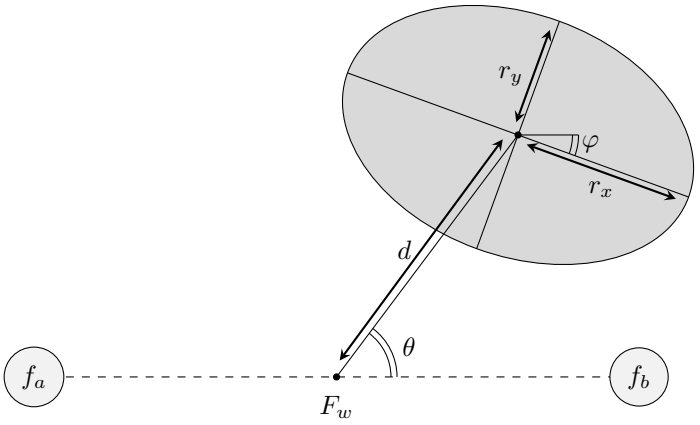


Figure 6.2: The estimated ellipse and its parameters. The distance property and angular properties are relative to F_w.

Note that there are multiple possible values for these parameters that represent an equivalent ellipse. If we increase φ by $\frac{1}{2}\pi$ and swap the axes r_x and r_y, then we still have an equivalent representation of an ellipse. This must be taken into account when regression is performed on the latter three mentioned variables.

The ellipse is fitted on the boundary of the reachable area. The ellipse for which the summed squared error distance to this boundary is minimized is considered to be the 'best-fit'. An approach to fit such an ellipse is given by Harlick et al. [18], where a general conic is fitted to a set of points. A general conic can be described by the following implicit second order polynomial:

$$F(x, y) = ax^2 + bxy + cy^2 + dx + ey + f = 0$$

Since the result of this approach is a general conic, it needs not be an ellipse. An adaptation of this approach forces the coefficients of the latter polynomial to meet the following condition, which ensures that the conic represents an ellipse [17, 12]:

$$4 \cdot ac - b^2 = 1$$

When the implicit representation of the best-fitted ellipse is found, then the coefficients of the polynomial can be converted to the values of the desired ellipse parameters as follows [10]:

$$\begin{aligned}
x &= \frac{cd - be}{b^2 - ac} \\
y &= \frac{ae - bd}{b^2 - ac} \\
r_x &= \sqrt{\frac{2(ae^2 + cd^2 + fb^2 - 2bde - acf)}{(b^2 - ac)[\sqrt{(a-c)^2 + 4b^2} - (a+c)]}} \\
r_y &= \sqrt{\frac{2(ae^2 + cd^2 + fb^2 - 2bde - acf)}{(b^2 - ac)[-\sqrt{(a-c)^2 + 4b^2} - (a+c)]}} \\
\varphi &= \begin{cases}
0 & \text{if } b = 0 \text{ and } a < c \\
\frac{1}{2}\pi & \text{if } b = 0 \text{ and } a > c \\
\frac{1}{2}\cot^{-1}(\frac{a-c}{2b}) & \text{if } b \neq 0 \text{ and } a < c \\
\frac{1}{2}\pi + \frac{1}{2}\cot^{-1}(\frac{a-c}{2b}) & \text{if } b \neq 0 \text{ and } a > c
\end{cases}
\end{aligned}$$

The cartesian coordinate (x, y) can then be converted to the polar coordinate (θ, d).

6.2 Regression of the Ellipse Parameters

In this section, we define a linear regression model for the ellipse parameters. A separate estimation model is built for each parameter of the ellipse. In order to take the proportions of hands into account (e.g. the lengths of fingers), we want to make the input variable of the model proportional to the maximum distance between the stationary fingers f_a and f_b. Therefore, the input variable is the current distance between f_a and f_b, divided by $d^+_{(a,b)}$. Also, the regression of the non-angular parameters (i.e. d, r_x and r_y) will be scaled into proportion by dividing these values by $d^+_{(a,b)}$. The general linear regression model with which we estimate one of the parameters is written as follows:

$$y(t) = w_0 + w_1 \cdot t + w_2 \cdot t^2$$

where t is the input variable. This model contains a constant term (i.e., the intercept w_0), a linear term ($w_1 \cdot t$) and a quadratic term ($w_2 \cdot t^2$). To find the values for the weight coefficients w_i, we first define the following three matrices:

$$\mathbf{X} = \begin{bmatrix} 1 & t_1 & t_1^2 \\ 1 & t_2 & t_2^2 \\ \vdots & \vdots & \vdots \\ 1 & t_N & t_N^2 \end{bmatrix} \quad \mathbf{p} = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_N \end{bmatrix} \quad \mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}$$

In these matrices, N denotes the number of ellipses that were fitted. The value t_n in $\mathbf{X}$ and p_n in $\mathbf{p}$ represent the input variable and the parameter value of the n^{th} fitted ellipse respectively. The resulting matrix of $\mathbf{X} \cdot \mathbf{w}$ contains the estimations of each ellipse parameter value p_n. The weight coefficients for which the sum of squared estimation errors between the ellipse parameters p_n and its estimations is minimized can be computed as follows [13]:

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{p}$$

As was mentioned in the previous section, the variable φ can take multiple values. Any value $\varphi + \frac{\lambda}{2}\pi$ (where $\lambda \in \mathbb{I}$) is an angle between the x-axis of F_w and some axis of the ellipse. An appropriate value for φ has to be found, which fits appropriately in the subsequent order of all the measured angles φ_i. Assume the measurements (t_i, φ_i) are sorted by the input variables t_i, in increasing order. For t_0, we let the angle φ_0 keep its default value. Then, for $i \in 1, 2, ..., N$, we let φ_i take the value $\varphi_i + \frac{\lambda}{2}\pi$ that is closest to φ_{i-1}. As was mentioned in the previous section, we need to swap the values of $r_{x,i}$ and $r_{y,i}$ if the angle φ_i refers to a different axis after its update (i.e. if $|\lambda| \, mod \, 2 \neq 0$). The method with which the quality of this model is measured is proposed in Section 6.4.1.

6.3 Calculating Reachablility

The regression models that were given in the previous sections show us how to find the parameters of the ellipse A that approximates the reachable area for finger f_c. In this section, we show how we can deduce the probability that a point can be reached by finger f_c. An example of such an approach would be to estimate the average proportion of the reachable area that is covered by A. For instance, if the expected amount that is covered equals 70%, then the probability that the point $p \in A$ can be reached would be 0.7. This is quite a crude estimation model, since the probability that points can be reached is equal throughout the entire area A. Also, the points $p \notin A$ are not taken into account. Instead, we wish to gather more *local* information on how much area is covered, so that we can perform more precise estimations.

We first introduce the concept of an *iso-ellipse*, which is a uniformly scaled version of another ellipse. This scale transformation is performed on the axes of the ellipse. For instance, Figure 6.3 shows an ellipse (drawn with a thick stroke) and two scaled versions of this ellipse (drawn dashed). The boundary of the smallest ellipse, which was scaled by 0.5, is fully contained within the reachable area P (shaded in gray), whereas the boundary of the largest ellipse does not intersect P at all. The boundary of the original ellipse partually intersects P. We will refer to the proportion of the ellipse *boundary* that intersects P as the *coverage* of the ellipse. As shown in Figure 6.3, the

coverage of the ellipse may change as the size of the ellipse is increased or diminished. In Section 6.3.1, we show the relationship between the coverage of an iso-ellipse and its scale factor. Next, in Section 6.3.2, a model is created which estimates the coverage, using the scale factor as the input variable.

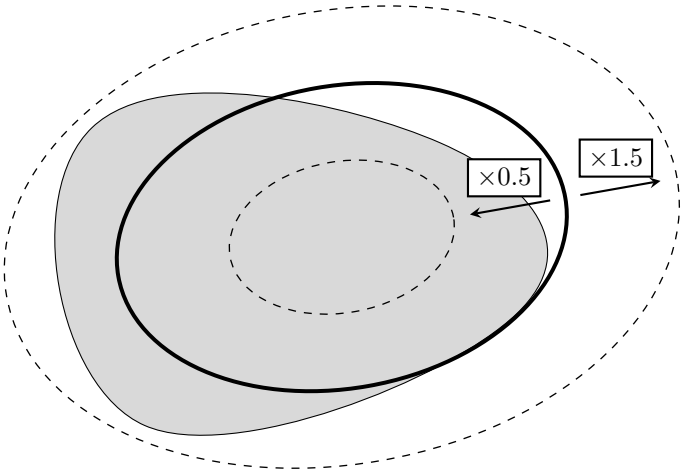


Figure 6.3: A fitted ellipse and two of its iso-ellipses.

6.3.1 The Relation Between Coverage and Scale

Since we seek the *proportion* of the ellipse boundary that intersects P, we are allowed to perform an affine transformation on the space in which the ellipse and P reside. The transformed version of P is denoted as P'. After the transformation, the ratio of the transformed ellipse perimeter inside P' to its total perimeter is preserved [9]. Therefore, we can transform our space, so that the ellipse becomes a unit circle, centered at the origin and aligned with the axes of F_w. The transformation is depicted in Figure 6.4. This is an affine transformation, since it only involves a translation, rotation and scale transformation. We shall refer to this transformed space as the *normalized space*. The iso-ellipse with scale factor λ of the original ellipse, will become a circle with radius λ in the normalized space.

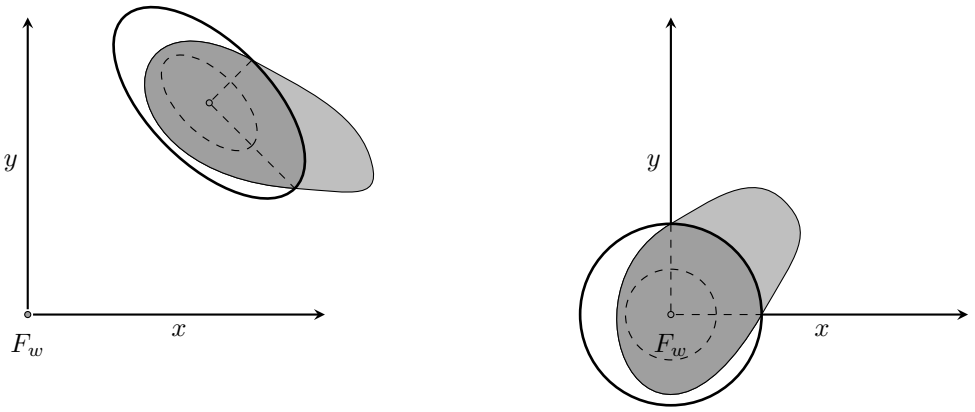


Figure 6.4: Applying the affine transformation.

We want to create a regression model which estimates the coverage, using λ as the input variable. The shape of the regression model will be based on ellipses, which' center point lies strictly within P. Also, in this modelling process, we will assume that P is a convex shape, as was obeserved in Section 6.1. The convex property is invariant to affine transformations [7]. Therefore, the convex property is preserved in the normalized space, so P' is also convex.

In terms of the normalized space, we assume that the center of the circle (i.e. F_w) must be a point in P'. From this, it follows that the coverage of $\lambda = 0$ equals 100%. We now define two other variables, λ^- and λ^+, which denote the *minimum* and *maximum* distance from F_w to the boundary of P' respectively. The coverage of any circle, which' radius is within the range $[0, \lambda^-]$, is equal to 100% (see Figure 6.5a). If the radius of the circle is greater than λ^+, then the coverage must be 0%, since no distance to any point in P' is greater than λ^+, as can be seen in Figure 6.5b.

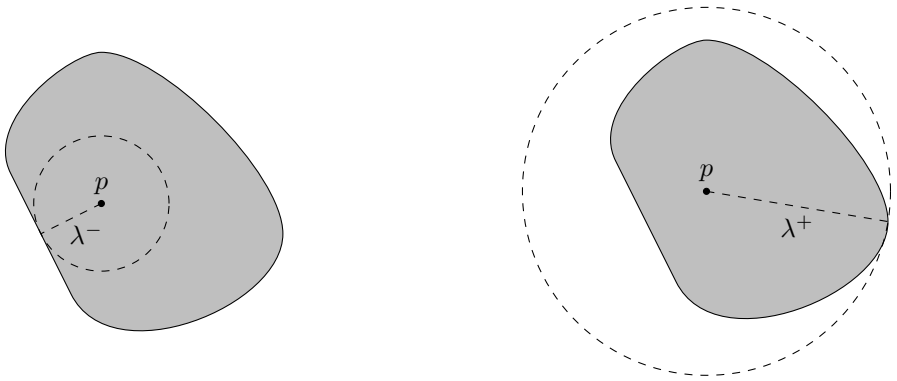


Figure 6.5: The circles with radius λ^- (a) and radius λ^+ (b)

What is left to define is how the coverage progresses as the circle is scaled from λ^- to λ^+. To do so, we will divide the space into segments as follows: a separating line is drawn from the circle center through every intersection point of the circle and the boundary of P'. If a circle segment is aligned with the boundary of P' (i.e. if there is a part with an infinite amount of intersecting points), then we only draw a line through the bounding points of this segment (if there are any). The segmentation is depicted in Figure 6.6.

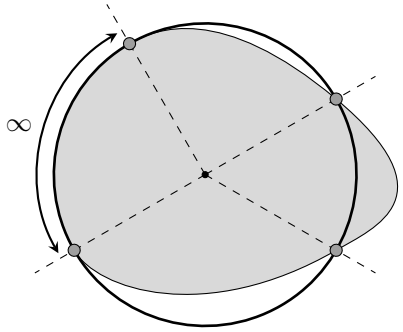


Figure 6.6: The circle is divided in segments. The segment denoted with '∞' has an infinite number of intersecting points.

We now have two sorts of segments:

1. a segment where the circle arc fully intersects P'
2. a segment where the circle arc does not intersect P', except for the intersection points between the circle and the boundary of P'

We will show that the coverage in all these sections deminishes when the circle radius is increased, by showing that the intersecting points of the separating lines and the increased circle boundary cannot be points in P'. We will show this for the intersection point q. We let $\mathbf{t}$ be the tangent vector at q that points to the next clockwise point on the boundary of P', as shown in Figure 6.7.

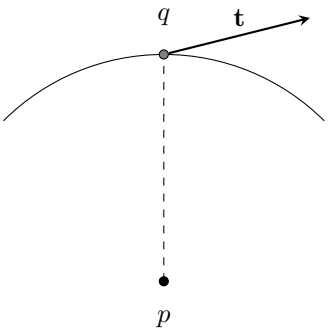


Figure 6.7: The point q is the intersection point of the circle and the boundary of the reachable area. The tangent vector to the next clockwise point on the reachable boundary is denoted as $\mathbf{t}$.

The point p is the center of the circle. Since P' is convex, and $p \in P'$, it follows that p must be oriented to the *right* hand side of $\mathbf{t}$ [3]. Therefore, the following inequality must hold:

$$\mathbf{t} \times (\mathbf{p} - \mathbf{q}) < 0$$

We now increase the radius of the circle with $\Delta\lambda$, as depicted in Figure 6.8.

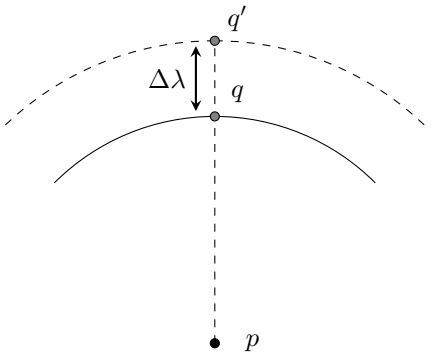


Figure 6.8: The radius of the circle is increased with $\Delta\lambda$.

The intersection-point of the increased circle and the line through p and q is denoted as q'. We want to know if it is possible that q' belongs to P'. To deduce this, we first assume that this is the case. If $q' \in P'$, then q' can *not* be oriented to the *left* hand side of $\mathbf{t}$, i.e., the following condition must hold:

$$\mathbf{t} \times (\mathbf{q}' - \mathbf{q}) \leq 0$$

Note that the vector $(\mathbf{q}' - \mathbf{q})$ is a scaled version of $(\mathbf{p} - \mathbf{q})$, where the scale factor α is less than zero. Therefore, the latter condition can be rewritten as:

$$\mathbf{t} \times \alpha(\mathbf{p} - \mathbf{q}) \leq 0, \text{ where } \alpha < 0$$

$$\Rightarrow \{\mathbf{a} \times \alpha(\mathbf{b}) = \alpha(\mathbf{a} \times \mathbf{b})\}$$

[3] Note that we assume that p lies *strictly* within P', i.e. it cannot be a boundary point of P'

$$\alpha(\mathbf{t} \times (\mathbf{p} - \mathbf{q})) \leq 0, \text{ where } \alpha < 0$$

$\Rightarrow$ { divide both sides with α, since $\alpha < 0$, we change the inequality sign }

$$\mathbf{t} \times (\mathbf{p} - \mathbf{q}) \geq 0$$

The latter expression contradicts the fact that $\mathbf{t} \times (\mathbf{p} - \mathbf{q})$ must be strictly *less* than zero. Therefore, q' cannot be a point in P'. From this, it follows that if λ is increased within the bounds $(\lambda^-, \lambda^+]$, then the coverage diminishes.

6.3.2 The Coverage Estimation Model

In the previous section, we have seen how the coverage relates to the scale factor λ:

- in the range $[0, \lambda^-]$, the coverage equals 100%
- in the range $(\lambda^-, \lambda^+]$, the coverage diminishes if λ increases
- in the range $(\lambda^+, \infty]$, the coverage equals 0%

From the second property, it shows that the preferred model $f(x)$ should be decreasing. From the first and third property, it follows that $f(x)$ should have two horizontal asymptotes, so that:

$$\lim_{x \to -\infty} f(x) = 100\%$$
$$\lim_{x \to \infty} f(x) = 0\%$$

These charactaristics can be found in the *logistic function* (or *logistic curve*). This curve is very common when phenomena are modelled in various fields of research, like physics, economics, chemistry, biology and medicine [8]. A simple logistic curve may be written as follows:

$$f(x) = \frac{1}{1 + e^{-g(x)}}$$

An example plot is shown in Figure 6.9.

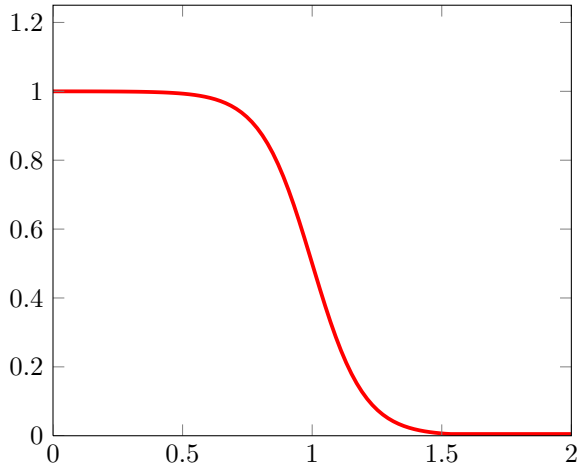


Figure 6.9: An example of the logistic curve, where $g(x) = -10x + 10$. The coverage ranges from 0 to 1.

To determine the probability that point p can be reached, we need to determine the scale value λ of the iso-ellipse on which p is located. From the coverage model, we can deduce the expected coverage using λ, which represents the probability that p is a point in P.

6.4 Measuring the Quality of the Estimation Models

So far, we have discussed two separate regression models with which we test whether points can be reached by finger f_c. With the first model, we estimate an ellipse with which the reachable area is approximated. From the second model, the expected coverages of the ellipse and its scaled iso-ellipses are gathered, so that we can deduce the probability that a point can be reached. In this section, we will show how we measure the quality of these two models. In Section 6.4.1, we show how we measure the quality of the estimated ellipses. Next, in Section 6.4.2, we do the same for the coverage estimation model.

6.4.1 Measuring the Quality of the Ellipse Estimation Model

To indicate the quality of an estimated ellipse, we need to determine how well it serves our purpose of determining whether a point lies in the reachable area P. This quality could be represented by the average probability that a reachable point $p \in P$ is indeed a point in P, according to the estimated ellipse. In the previous section, it was mentioned that the coverage of an iso-ellipse represents the probability that a (random) point on its boundary is a point in P. For each point $p \in P$, we will deduce the iso-ellipse that has point p on its boundary. Of these ellipses, we will calculate the expected coverage. This will serve as the quality indicator of the estimated ellipse. If the expected coverage is high, then for most points in P we are certain that they are indeed points in P, according to the estimated ellipse and its iso-ellipses. Similarly, if the expected coverage is low, then most points in P will be disregarded, since we cannot be certain that they are indeed points in P in most cases.

In terms of the normalized space, an iso-ellipse that was scaled by factor λ becomes a circle with radius λ. The coverage of this circle is denoted as $\texttt{Coverage}(\lambda)$. Note that this is the *actual* coverage of the circle, not the *estimated* coverage from the model that was discussed in the previous section. To determine the expected coverage, we can integrate $\texttt{Coverage}(\lambda)$ over all points $p \in P'$ and divide this by the area of P'. To integrate over all points in P', we will perform a double intergral, where we first integrate over λ, which ranges from 0 to ∞. Next, we integrate over the points on the boundary of the circle with radius λ that are also points in P'. The coverage that is associated with each point on this boundary equals $\texttt{Coverage}(\lambda)$. Therefore, the result of the second integral is the length of the circle-arc inside P' (which equals $2\pi\lambda \cdot \texttt{Coverage}(\lambda)$) times the coverage associated with this circle (which is $\texttt{Coverage}(\lambda)$). The expected coverage can be calculated as follows:

$$\frac{\int_{\lambda=0}^{\infty} \{2\pi\lambda \cdot \texttt{Coverage}(\lambda)^2\}}{\texttt{Area}(P')}$$

6.4.2 Measuring the Quality of the Coverage Estimation Model

Normally, to measure the quality of a fitted logistic regression model, the χ^2 (Chi-squared) test is performed. However, this test is only applicable when the output variable of the regression model is a *categorical* variable [3, 14]. In the coverage regression model, a continuous *numerical* variable is estimated, so therefore we cannot use the χ^2 to indicate the quality of our model. Also, the R^2 quality measure cannot be used for this model, since it only applies on regular linear regression models. There are some pseudo-R^2 tests that could indicate the quality of the logistic regression model. However, the values that are deduced from this test are not intuitive. For instance, some of these tests do not even yield the optimal value (i.e. 1) when the logistic regression model fits the data-set perfectly. These pseudo-R^2 tests can be used to compare different fits of the data, but it does not provide useful information on the model itself. Instead, we will look for the estimation error variance of the fitted coverage estimation models. From these values, we will deduce if the regression models fit the data-points appropriately. We will discuss the estimation error properties in Section 6.6.

6.5 Implementation and Measuring

The application with which the reachable areas are measured was developped for the Microsoft PixelSense, using WPF (.NET 4.0 framework). The Accord.NET framework was used to process the matrices with which the ellipse parameter estimation models are built [1]. This framework also incorporates a module with which a logistic regression model can be fitted on a data-set [4]. The reachable area of each finger $f_c \in I$ was measured for each finger tuple $(f_a, f_b) \in I_2$, where $f_a \neq f_c$ and $f_b \neq f_c$. The distance between the two stationary fingers is chosen ad random from the range of possible distances between these two fingers. For this purpose, we first measure the stretch values between each finger tuple, using the procedure that was described in Section 5.4. The entire session was separated in two (identical) sub-sessions for each person. The procedure of such a sub-session is as follows:

- for each tuple (f_a, f_b) and third finger f_c, we measure the reachable area as follows:
  - we let d be a random distance between f_a and f_b, which is within the range of possible distances between f_a and f_b.
  - the user is asked to place f_a and f_b on the respective button controls that are shown on-screen. The distance between the centers of these two controls is d.
  - next, the user is asked to draw the boundary of the reachable area on the screen, whilst the fingers f_a and f_b remain stationary at their respective locations. The area is drawn using a plastic marker with a byte-tag on its center, which is moved by f_c. The Microsoft PixelSense is able to deduce the center position of the byte-tag when it's placed on the screen. The reason we use the marker instead of direct finger-input is because we want to omit 'false' finger-input that may arise because of the presence of light in our testing environment.
  - once the boundary of the reachable area is drawn, a button is pressed by the observer, so that the next measurement can be performed. Note that the input variable of this measured area is d divided by the maximum measured distance between f_a and f_b. This value will be used lateron when we construct the ellipse estimation models.
- finally, the distance measurements (including the calibration value) and the boundary points of the reachable area are stored, so that we may process them further and construct the regression models.

Since we perform two sub-sessions for each combination of fingers, we get two measurements in total for each combination of fingers for each person. Figure 6.10 shows how the reachable area of f_1 is measured, where the fingers f_2 and f_4 are placed at fixed locations on the screen.

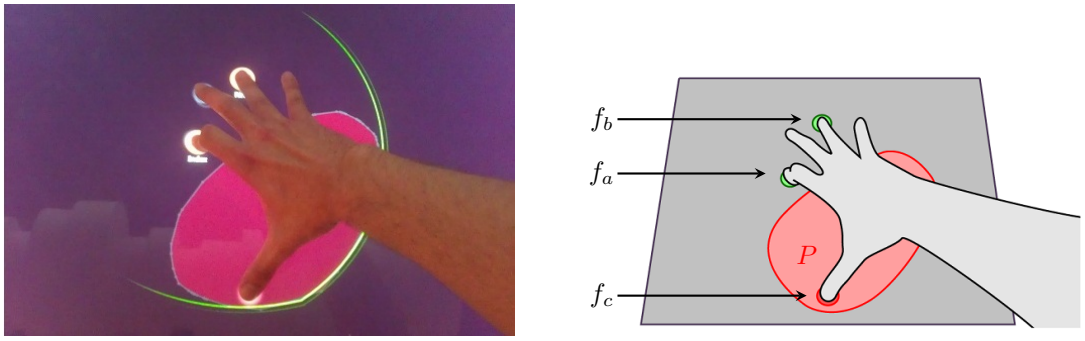


Figure 6.10: Measuring the reachable area of finger f_1. The stationary fingers are f_2 and f_4. The reachable area P is shaded red.

6.6 Results

In this section, we will discuss the results of the fitted models, with which the reachable area is estimated. First, in Section 6.6.1, we will show the regression models that were fitted for the ellipse parameter estimation. Next, in Section 6.6.2, we will show the models that were fitted from which the expected coverage can be calculated for the estimated ellipses. Finally, in Section 6.6.3, we will discuss a degenerate case that may occur when the reachable area is estimated, which affects the coverage estimation model.

6.6.1 Fitted Ellipse Parameter Estimation Models

In Appendix C.1 - at the end of this document - there is a list of tables that include the coefficients with which each separate ellipse parameter is estimated for each tuple of stationary fingers (f_a, f_b) and third finger f_c. The input variable of these regression models is the current distance between the stationary fingers f_a and f_b, divided by the maximum distance between f_a and f_b. For the data-points with which the models were fitted, we have used the *actual* maximum distance between the stationary fingers in the definition of the input parameter. However, if this model is to be used, the model requires the *estimated* distance between the stationary fingers, since in this case the actual distance is not known. Therefore, in the measurement of the quality of the estimated ellipses, we use the *estimated* distance between the stationary fingers as well. The regression model with which we estimate this maximum distance is as given in Section 5.

The tables in Appendix C.2 include the quality measurements of each separate ellipse estimation model. The measurements can be interpreted as the average probability that a point $p \in P$ is indeed considered to be a point in P, according to the estimated ellipse and its iso-ellipses. The overall expected coverage of the ellipses that were fitted over fifteen people is 77.8%. To determine if the model is considered to be a *general* model, we have performed multiple k-fold cross-validation tests. In Figure 6.11, the expected coverage of the training set (of size k) and the test-data (of size $N - k$) is shown for each $k \in [2, N - 1]$. If k is low, e.g. $k = 2$, then the expected coverage of the training data is higher than when $k > 2$. This is because when we use merely two measurements in the construction of the regression model, then the model would be fitted specifically for these two measurements and would not apply to measurements in general. Also, the expected coverage of the test-data when $k = 2$ is low in comparison to the expected coverage when $k > 2$. In this case, the regression model is *over-fitted*. However, as k approaches N, we can see that the difference between the expected coverage of the test-data and the train-data becomes smaller. If k is large, then the model that is generated from the training data would be a more appropriate general model, since it also applies on the data on which the model was not fitted. We consider the model that was generated when $k = 14$ to be an appropriate general model, and will therefore use the entire data-set - which consists of all fifteen measurements - in the modelling of the ellipse estimation model.

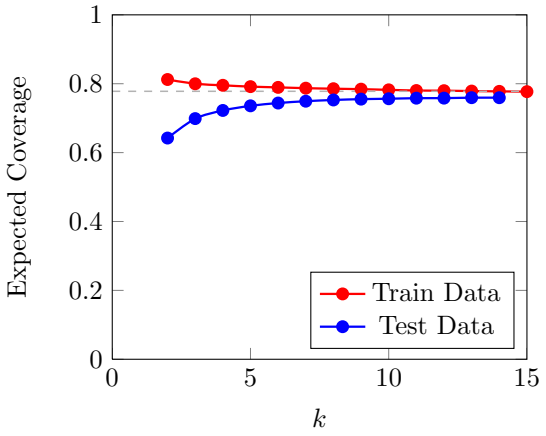
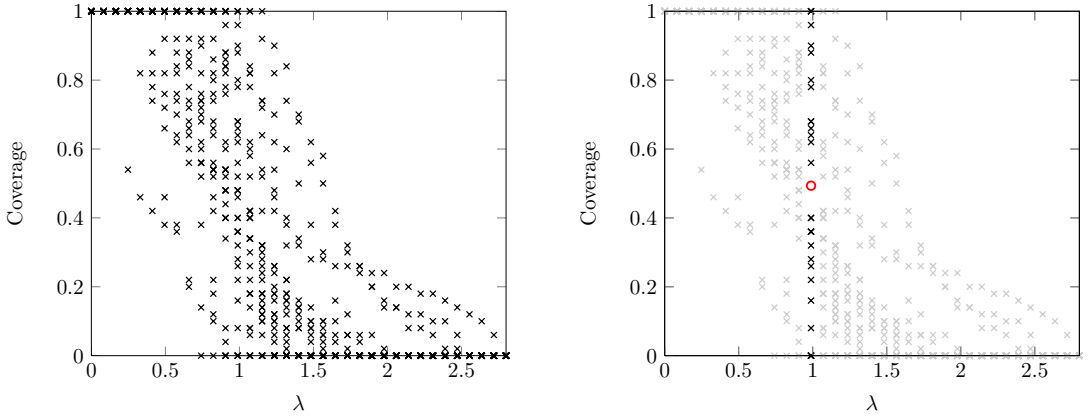


Figure 6.11: The expected coverage of the training data (of size k) and the test data (of size $N - k$), where $N = 15$.

6.6.2 Fitted Estimation Models of the Expected Coverage

In this section, we show how the coverage estimation models are generated from the estimated ellipses. Figure 6.12a contains a discrete set of coverage samples that are associated with the first combination of fingers, i.e. the stationary fingers (f_1, f_2) and third finger f_3. The coverages were sampled over a range of λ values. In Figure 6.12b, the plotted samples at $\lambda = 0.998$ are highlighted. The red circle represents the *mean* of these highlighted points, and represents the probability that a point on the boundary of the iso-ellipse with scale factor $\lambda = 0.998$ lies within the reachable area, given the current combination of fingers.



(a) The coverage samples of all estimated ellipses

(b) A subset of the coverage samples, where $\lambda = 0.998$. The red circle represents the mean coverage value of this subset.

Figure 6.12: The sample plot of the iso-ellipse coverage over scale value λ. The coverage ranges from zero to one.

The mean coverage is calculated for each λ. In Figure 6.13, the red circles represent these mean values for each λ. Note that the coverage samples at the top left and bottom right are relatively dense. The mean coverage values tend to be near these dense locations.

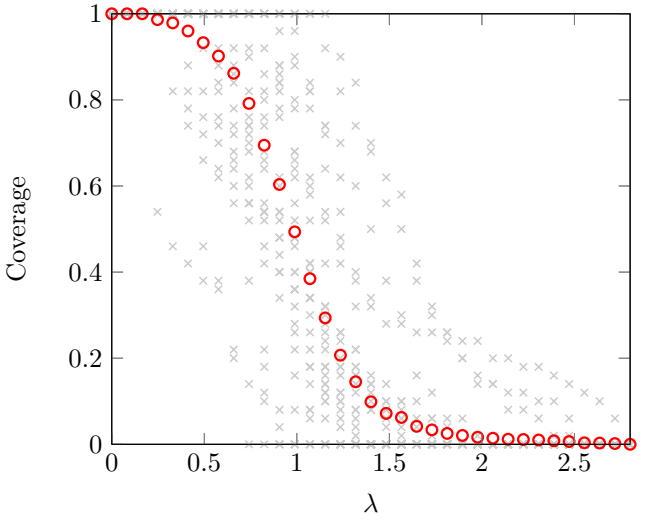
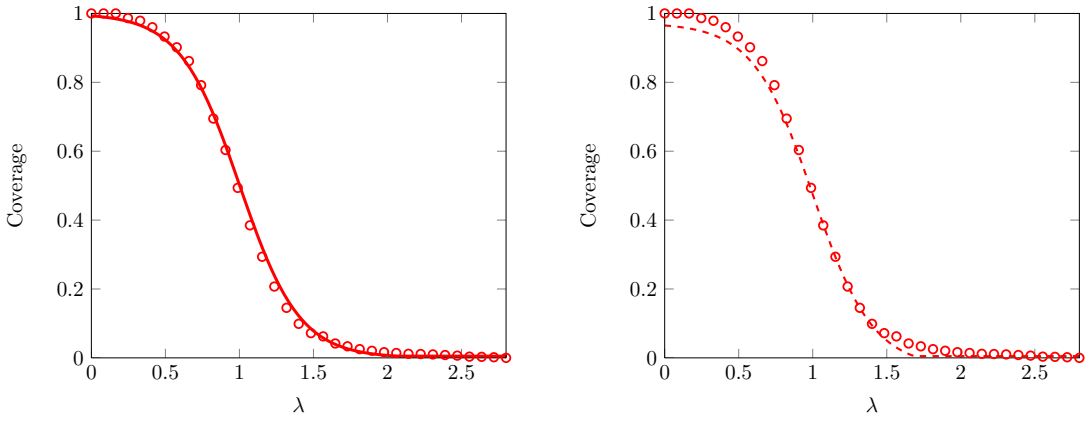


Figure 6.13: The mean expected coverage for each λ.

The logistic regression model is fitted on the mean coverage points. The result of fitting the regression model on these points is shown in Figure 6.14a. The standard deviation (σ) of the expected coverage estimation error for this particular logistic regression is 1.4%. In Figure 6.14b,

the same logistic curve is plotted, of which twice the standard deviation ($2\sigma = 2.8\%$) is subtracted. For any λ, the probability that a point can be reached is *at least* equal to the coverage that is inferred from the regression model, minus $2 \cdot \sigma$, in approximately 98.8% of all cases.



(a) The logistic regression model is fitted through the mean values.

(b) The dashed line represents the fitted logisic curve, minus $2 \cdot \sigma$

Figure 6.14: Fitting the logistic regression model.

Figure 6.15 shows the 'worst' fitted logistic curve (i.e. the curve with the highest standard deviation σ of the estimation error), which is the curve associated with the stationary fingers $(f_3, f_4) \in I_2$ and third finger $f_2 \in I$. The standard deviation of this regression model's estimation error is 2.75%.

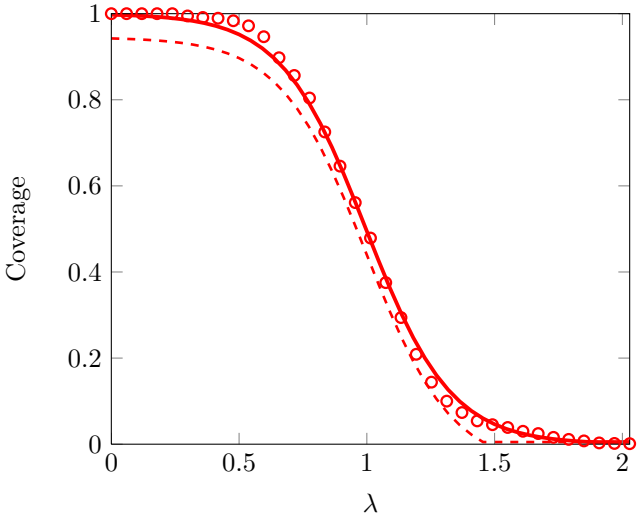
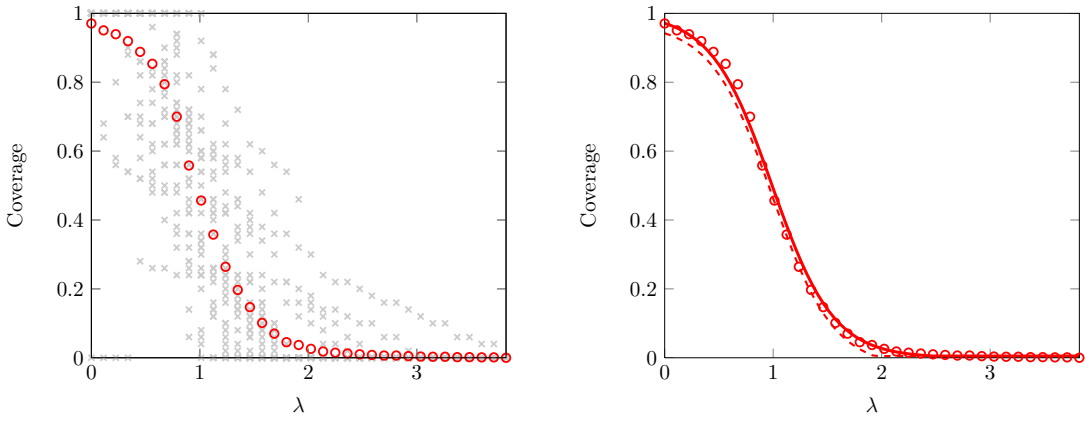


Figure 6.15: The 'worst' fitted logistic regression curve.

6.6.3 Degenerate Case in the Coverage Estimation Model

As was mentioned in Section 6.3, the preferred coverage estimation model was based on the cases in which the center point of the estimated ellipse lies strictly within the actual reachable area. However, in approximately 1.3% of the cases, it happens that the centroid of the estimated ellipse lies *outside* of the reachable area. This degenerate case does not occur for every combination of stationary fingers $(f_a, f_b) \in I_2$ and third finger $f_c \in I$. An example of a combination of fingers for which this does occur are the stationary fingers (f_2, f_3) and third finger f_4. In Figure 6.16a, the sampled coverage values are plotted for this combination of fingers. At $\lambda = 0$, some coverage

samples are zero. Therefore, the mean value at $\lambda = 0$ is less than one. In Figure 6.16b, the logistic regression model is fitted on the mean coverage values. The standard deviation of the estimation error equals 1.45%, which is less than the general standard deviation. It shows that the logistic curve can still yield accurate estimations in this degenerate case.



(a) The mean coverage values. **(b)** The fitted logistic curve.

Figure 6.16: Fitting the logistic regression model in the degenerate case.

6.7 Applying the Proposed Estimation Model

This section summarizes how we can calculate the probability that point p can be reached by finger f_c, given that two other fingers $(f_a, f_b) \in I_2$ are are located at fixed positions on the screen. We place the world frame F_w at the mean location of f_a and f_b. The x-axis of F_w points to f_b.

The reachable area of f_c is approximated with an ellipse. The estimated ellipse is inferred from a regression model. The list of tables that are presented in Appendix C.1 contain the regression coefficients with which each separate parameter of the ellipse is estimated. The input parameter of these regression models is the *current* distance between f_a and f_b, divided by the *estimated* maximum distance between f_a and f_b, which is $d^+_{(a,b)}$. The estimated maximum distance can be calculated with the method that was presented in Section 5.6.1. The non-anglular estimated parameters (i.e. r_x, r_y and d) must be multiplied with $d^+_{(a,b)}$. Note that the angular parameters φ and θ are relative to the x-axis of F_w.

We now seek the scale value λ of the iso-ellipse that has point p on its boundary. An iso-ellipse is a uniformly scaled version of the estimated ellipse, where the scale transformation was applied along the axes of the ellipse. To find λ, we transform the current space to the normalized space, where the estimated ellipse becomes a unit-circle. The scale value λ is the distance from the transformed point p' to the origin of the normalized space (i.e. the radius of the iso-circle that has point p' on its boundary).

The coverage that is associated with the scale value λ can be deduced from the logistic regression model. The coefficients of this regression model are presented in Appendix C.3 for each tuple $(f_a, f_b) \in I_2$ and third finger f_c. The scale value λ serves as the input parameter of this regression model. The coverage that is calculated in this model represents the probability that a point on the respective iso-ellipse is a point in the reachable area. To make sure that the inferred probability holds for 98.8% of all cases, we subtract twice the standard deviation (2σ) of the coverage estimation error from the estimated value. These standard deviations can also be found in Appendix C.3 for each tuple $(f_a, f_b) \in I_2$ and third finger f_c.

Using this model, we can also calculate the reachable area of f_c that consists of all points for which the associated coverage is above some threshold value. This threshold value is denoted as t. The reachable area is the area of the iso-ellipse with scale value λ, for which the estimated expected coverage is equal to t. Note that the coverage estimation function is strictly decreasing. Hence, any iso-ellipse with a scale value smaller than λ will have a greater associated coverage (i.e. the probability that the points within the iso-ellipse is in reach of f_c is at least t).

To determine which value of λ would yield an iso-ellipse with coverage t, we must solve the following equation:

$$\frac{1}{1+e^{-g(\lambda)}} - 2\sigma = t$$

$\Rightarrow$ { Bring -2σ to the right and multiply both sides with $1+e^{-g(\lambda)}$ }

$$1 = (t+2\sigma)\cdot(1+e^{-g(\lambda)})$$

$\Rightarrow$ { Divide both sides with $t+2\sigma$ and subtract 1 from both sides }

$$\frac{1}{t+2\sigma} - 1 = e^{-g(\lambda)}$$

$\Rightarrow$ { $e^x = y \leftrightarrow x = \ln(y)$, $g(\lambda) = w_0 + w_1 \cdot \lambda$ }

$$\ln(\frac{1}{t+2\sigma} - 1) = -w_0 - w_1 \cdot \lambda$$

$\Rightarrow$ { Solve for λ }

$$\lambda = \frac{\ln(\frac{1}{t+2\sigma} - 1) + w_0}{-w_1}$$

The probability that a point within the iso-ellipse with scale value λ can be reached by f_c is at least t. This holds for 98.8% of all points on the boundary of the iso-ellipse, and even a larger percentage of points within the iso-ellipse.

7 Conclusions and Future Work

In the previous sections, we have proposed a model with which we can estimate the reachable area of a right-hand finger on a planar surface, given that either one or two other fingers of the same hand are placed at fixed positions on this surface. The models have been fitted on user-input measurements. A seperate estimation model was defined and fitted for each combination of fingers. In Section 7.1, we will reflect on the estimation models we have obtained and the accuracy of its estimations. Next, in Section 7.2, we will discuss how the proposed models can be extended to serve the general case when there are n stationary fingers placed on the surface. Also, we will discuss some other additions and adjustments to the current model that might yield more accurate estimations.

7.1 Reflection on the Proposed Models

In this section, we will reflect on the models that have been fitted with which we estimate the reachable areas. In Section 7.1.1, we will discuss the model that has been fitted in the case that there is one stationary finger placed on the planar surface. Next, in Section 7.1.2, we will reflect on the model that has been fitted when there are two stationary fingers placed on the surface.

7.1.1 The estimation model for one stationary finger

In this case, the models estimate the reachable area of finger $f_b \in I$, given that some other finger $f_a \in I$ is placed at a fixed position on the screen. The reachable area has the shape of an annulus. In order to estimate the reachable area, we need to estimate the radii of the annulus. To estimate the outer radius, we have defined and fitted a linear regression model on the touch-input measurements of 36 people. This regression model requires an input parameter, which is the maximum distance between the thumb and the little finger. Using this calibration value yields the most accurate estimations from the model. The standard deviation of the estimation error in this regression is approximately 1 cm. Of course, each separate estimation model has its own estimation error properties. Note that the calibration procedure can be performed very quickly, so the usage of this model would not become a bottleneck in the accessibility of applications and games in particular. As for the estimation of the inner radius of the annulus, we have chosen to perform a plain estimation. The standard deviation of this plain estimation error is approximately 3.6 mm.

7.1.2 The estimation model for two stationary fingers

The estimation model for two stationary fingers estimates the reachable area of finger f_c, given that two stationary fingers $(f_a, f_b) \in I_2$ are placed at fixed positions on the screen. We have observed that the reachable area is a convex area, which we approximate with an ellipse. An ellipse was fitted on the touch-input measurements of 15 people. For each combination of stationary fingers and f_c, we have measured two reachable areas per person. Therefore, the data-set consists of 30 measurements per combination of fingers. A regular linear regression model was fitted for each separate parameter of the ellipses. A quality measure was defined for the estimated ellipses, with which we have shown that the ellipse estimation model can be considered a *general* model, since the quality of the reachable area estimations is approximately equal for the training data and the test data. This has been inferred from the k-fold cross validation procedure.

We have introduced the concepts of coverage and iso-ellipses. An iso-ellipse is a uniformly scaled version of an ellipse, where the transformation is performed on the axes of the ellipse. The coverage denotes the proportion of an ellipse boundary that intersects the reachable area P, which is equal to the probability that a point on the ellipse boundary is a point in P. The relation between the coverage and the scale value of iso-ellipses has been measured and fitted in a logistic regression model. From this model, the probability can be deduced that a point p lies within the reachable area, given the scale value of the iso-ellipse that has point p on its boundary. The overall standard deviation of the probability estimation error is approximately 0.018.

The input parameter of the ellipse estimation model requires the estimated maximum distance between the two stationary fingers. This value can be deduced from the estimation model for one

stationary finger. Therefore, this model does not require an additional calibration task and does not compromise the accessibility of the calibration procedure.

7.2 Future Work

In this section, we discuss some ways in which the model that has been presented in this thesis can be adjusted or extended so that it would yield more accurate estimations of the reachable area, or to make the model applicaple in more cases. In Section 7.2.1, we propose how this model can be extended so that is would serve the general case, in which n stationary fingers are placed on the surface. Also, we show how we could take the presence of multiple hands into account. Next, in Section 7.2.2, we discuss how dynamic features can be used in the model so that it adapts its estimations using the input that is provided by the user at run-time.

7.2.1 A General Multi-Touch Input Model for n Fingers

If we want to create a model in the case that four or five stationary fingers are placed on the surface, then we could define a similar model as was presented in Section 6. The world frame F_w could be defined as the frame between the two stationary fingers with the lowest indices in I. The x-axis of F_w could point to the finger with the lowest index. Since the two fingers that define F_w lie on the x-axis, their locations can be denoted with a single distance value from $\mathbb{R}$. However, the third and fourth stationary finger are not necessarily located on the x-axis, so their positions must be sampled from $\mathbb{R}^2$ during the measurement of the reachable area. Therefore, this model requires many more measurements than the one that was presented in Section 6. In the case that there are just two stationary fingers on the screen, we have observed that the reachable area is convex in general. This observation might not hold when there are more than two stationary fingers present on the screen. In that case, we also have to define a new model that estimates the shape of the reachable area appropriately.

Another consideration in the extention of the presented model is to take the presence of multiple hands into account. The proposed model could be applied for each separate hand, but this model only incorporates the intrinsic constaints of the hand anatomy. The physical presence of other hands could constrain the reachable area of a finger. In order to take this constraint into account, we could simulate the situation with 3D hand models and test for possible collisions. However - as was mentioned in Section 2.2 - creating a 3D reconstruction model of the hand can become computationally intensive. Instead, we could restrict the finger placement model so that touch input can only occur in the area that is not close to any other hands that are located on the screen.

7.2.2 A Dynamic Estimation Model

The model that is proposed in this thesis is a static model. We could possibly add a dynamic feature by observing the touch-input that is delivered by the user at run-time. From these observations, the model could learn something about the intrinsic properties of the hand. For instance, if we learn that the maximum distance between two fingers is larger than what the maximum distance model estimates, then we could adjust the model with this new information.

Also, if it is known that the maximum distance between two fingers is larger than the estimated distance, then we could also adjust the model by adressing some known correlating properties of the hand. To illustrate this, we have provided a table in Appendix A.4, which contains the correlation coefficients of the estimation errors (in the estimation of the maximum distance) between each finger tuple . The error correlation between the tuples (f_1, f_2) and (f_1, f_3) is large (i.e.: 0.73). Therefore, if the maximum distance between f_1 and f_2 is larger than the estimated distance, then it is very likely that the maximum distance between f_1 and f_3 is also larger than its estimation. Using this information, we could adjust the maximum distance estimation model, which would then provide more accurate estimations.

Note that the finger-index that is associated with the touch-input points might not be known a priori. However, by adressing the proposed model, we could calculate the probability of which finger-index is associated with a touch-input point. For instance, if the distance between two touch-input points is very large, then it is more likely that the associated fingers are the thumb and pink, rather than the middle- and ring-finger.

References

[1] Accord.net framework project. http://code.google.com/p/accord/. Accessed July 2012.

[2] Capacitive sensing on multi-touch devices. http://electronics.howstuffworks.com/iphone3.htm. Accessed June 2012.

[3] The chi square statistic. http://math.hws.edu/javamath/ryan/ChiSquare.html. Accessed July 2012.

[4] Logistic regression in c#. http://crsouza.blogspot.nl/2010/02/logistic-regression-in-c.html. Accessed July 2012.

[5] Number of applications and games featured in the app store. http://www.apple.com/ipodtouch/from-the-app-store/. Accessed June 2012.

[6] Regression through the origin and its pitfalls. http://courses.washington.edu/qsci483/Lectures/20.pdf. Accessed March 2012.

[7] Wikipedia: Convex functions. http://en.wikipedia.org/wiki/Convex_function. Accessed July 2012.

[8] Wikipedia: the logistic function. http://en.wikipedia.org/wiki/Logistic_function. Accessed July 2012.

[9] Wolfram mathworld: Affine transformations. http://mathworld.wolfram.com/AffineTransformation.htmll. Accessed July 2012.

[10] Wolfram mathworld: Ellipse. http://mathworld.wolfram.com/Ellipse.html. Accessed July 2012.

[11] Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tabless.* Dover Publications, 1972.

[12] Maurizio Pilu Andrew Fitzgibbon and Robert B. Fisher. Direct least square fitting of ellipses. *Pattern Analysis and Machine Intelligence*, 21(5).

[13] Christopher M. Bishop. *Pattern Recognition and Machine Learning.* Springer, 2006.

[14] DeGroot and Schervish. *Probability and Statistics.* Addison Wesley, 2002.

[15] Joseph G. Eisenhauer. Regression through the origin. *Teaching Statistics*, 25(3), 2003.

[16] Radim Halii and Jan Flusser. Modeling the constraints of human hand motion. Technical report, Beckman Institute, University of Illinois at Urbana-Champaign.

[17] Radim Halii and Jan Flusser. Numerically stable direct least squares fitting of ellipses. Technical report, Department of Software Engineering, Charles University.

[18] R. M. Harlick and L. G. Shapiro. *Computer and Robot Vision.* Addison Wesley, 1993.

[19] Sung Uk Lee. Modeling the constraints of human hand motion. Technical report.

[20] Daisaku Arita Ryuji Fujiki and Rin ichiro Taniguchi. Real-time 3d hand shape estimation based on inverse kinematics and physical constraints. Technical report, Department of Intelligent Systems, Kyushu University.

[21] Chuan-Yen Chiang et al. Yen-Lin Chen, Wen-Yew Liang. Vision-based finger detection, tracking, and event identification techniques for multi-touch sensing and display systems. *Sensors*, 11.

Appendices

A Maximum Stretch Regression (2 Fingers, $N = 36$)

A.1 Table of Regression Coefficients

$$w_i = \frac{\sum_{n=1}^{N}(x_{n,i} \cdot x_{n,c})}{\sum_{n=1}^{N}(x_{n,c}^2)}$$

w_i		$i \in I_2$									
		(1,2)	(1,3)	(1,4)	(1,5)	(2,3)	(2,4)	(2,5)	(3,4)	(3,5)	(4,5)
$c \in I_2$	(1,2)	1	1.15	1.18	1.18	0.53	0.71	0.9	0.46	0.73	0.46
	(1,3)	0.86	1	1.02	1.02	0.46	0.61	0.78	0.4	0.63	0.4
	(1,4)	0.83	0.96	1	0.99	0.44	0.6	0.75	0.39	0.61	0.39
	(1,5)	0.84	0.97	1	1	0.45	0.6	0.76	0.39	0.61	0.39
	(2,3)	1.83	2.12	2.18	2.18	1	1.32	1.66	0.86	1.35	0.86
	(2,4)	1.38	1.59	1.64	1.63	0.74	1	1.25	0.65	1.02	0.64
	(2,5)	1.1	1.27	1.31	1.3	0.59	0.79	1	0.51	0.81	0.51
	(3,4)	2.08	2.41	2.48	2.48	1.12	1.5	1.89	1	1.54	0.98
	(3,5)	1.34	1.55	1.6	1.59	0.72	0.96	1.22	0.63	1	0.63
	(4,5)	2.09	2.42	2.49	2.49	1.13	1.51	1.9	0.99	1.56	1

A.2 Tables of Expected Prediction Errors (in millimeters)

$$\epsilon = x_{n,i} - w_i \cdot x_{n,c}$$

$\mathbb{E}[\epsilon\|c,i]$		$i \in I_2$									
		(1,2)	(1,3)	(1,4)	(1,5)	(2,3)	(2,4)	(2,5)	(3,4)	(3,5)	(4,5)
$c \in I_2$	(1,2)	0	0.22	0.27	0.5	0.3	0.34	0.71	0.67	0.69	0.64
	(1,3)	0.04	0	0.07	0.29	0.18	0.29	0.6	0.53	0.53	0.49
	(1,4)	0.16	0.17	0	0.35	0.3	0.28	0.59	0.5	0.48	0.46
	(1,5)	0.07	0.05	0.01	0	0.04	0.11	0.33	0.35	0.26	0.32
	(2,3)	2.26	2.53	2.75	2.5	0	0.87	1.78	0.98	1.5	0.83
	(2,4)	1.68	2.03	1.97	2.01	0.38	0	1.24	0.57	0.76	0.45
	(2,5)	0.33	0.41	0.31	0.31	-0.09	-0.19	0	0.16	-0.04	0.02
	(3,4)	2.64	2.88	2.79	2.75	0.87	0.88	1.83	0	1.06	0.51
	(3,5)	1.51	1.65	1.52	1.5	0.49	0.28	0.86	0.37	0	0.06
	(4,5)	3.36	3.67	3.61	3.6	1.11	1.26	2.27	0.88	1.16	0

	$c \in I_2$									
	(1,2)	(1,3)	(1,4)	(1,5)	(2,3)	(2,4)	(2,5)	(3,4)	(3,5)	(4,5)
$\mathbb{E}[\epsilon\|c]$:	0.43	0.3	0.33	0.16	1.6	1.11	0.12	1.62	0.82	2.09

A.3 Tables of Prediction Error Variance (in squared millimeters)

$var[\epsilon\|c,i]$		$i \in I_2$									
		(1,2)	(1,3)	(1,4)	(1,5)	(2,3)	(2,4)	(2,5)	(3,4)	(3,5)	(4,5)
$c \in I_2$	(1,2)	0	49	90	111	131	178	147	143	212	165
	(1,3)	37	0	48	66	117	179	133	128	186	148
	(1,4)	63	46	0	70	133	168	119	121	167	141
	(1,5)	79	63	71	0	101	153	83	109	141	131
	(2,3)	446	533	641	484	0	158	232	131	253	135
	(2,4)	343	459	459	412	89	0	142	87	123	96
	(2,5)	180	217	207	143	83	91	0	84	78	90
	(3,4)	639	760	761	680	171	201	307	0	191	105
	(3,5)	389	454	433	363	136	117	116	79	0	61
	(4,5)	741	882	894	822	177	223	330	105	149	0

	$c \in I_2$									
	(1,2)	(1,3)	(1,4)	(1,5)	(2,3)	(2,4)	(2,5)	(3,4)	(3,5)	(4,5)
$var[\epsilon\|c]$:	123	104	103	93	302	221	117	383	215	434

A.4 Table of Correlation between Errors ($c = (f_1, f_5)$)

$\rho_{\epsilon_i,\epsilon_j}$		$j \in I_2$									
		(1,2)	(1,3)	(1,4)	(1,5)	(2,3)	(2,4)	(2,5)	(3,4)	(3,5)	(4,5)
$i \in I_2$	(1,2)	1	0.73	0.5	-	-0.07	0.09	0	-0.21	-0.19	-0.2
	(1,3)	0.73	1	0.61	-	-0.03	-0.01	-0.1	-0.13	-0.16	-0.1
	(1,4)	0.5	0.61	1	-	-0.23	0.07	0.03	-0.01	0.01	0
	(1,5)	-	-	-	-	-	-	-	-	-	-
	(2,3)	-0.07	-0.03	-0.23	-	1	0.56	0.45	0.31	0.24	0.41
	(2,4)	0.09	-0.01	0.07	-	0.56	1	0.63	0.52	0.61	0.58
	(2,5)	0	-0.1	0.03	-	0.45	0.63	1	0.46	0.67	0.61
	(3,4)	-0.21	-0.13	-0.01	-	0.31	0.52	0.46	1	0.56	0.6
	(3,5)	-0.19	-0.16	0.01	-	0.24	0.61	0.67	0.56	1	0.8
	(4,5)	-0.2	-0.1	0	-	0.41	0.58	0.61	0.6	0.8	1

B Minimum Stretch Regression (2 Fingers, $N = 20$)

B.1 Table of Regression Coefficients

$$w_{0,i} = \frac{\sum(x_{n,i}) - w_{1,i} \cdot \sum(y_{n,c})}{N}$$

$$w_{1,i} = \frac{N \cdot \sum(x_{n,i} \cdot y_{n,c}) - \sum(x_{n,i}) \cdot \sum(y_{n,c})}{N \cdot \sum(y_{n,c}^2) - (\sum y_{n,c})^2}$$

where $x_{n,i} \in D_{min}$ and $y_{n,c} \in D_{max}$. The summations iterate from $n = 1$ to $n = N$.
Intercept:

w_0		$i \in I_2$									
		(1,2)	(1,3)	(1,4)	(1,5)	(2,3)	(2,4)	(2,5)	(3,4)	(3,5)	(4,5)
$c \in I_2$	(1,2)	35.89	35.85	21.41	3.61	16.95	16.5	27.1	15.39	37.38	28.59
	(1,3)	34.39	35.24	20.21	4.2	18.28	18.11	27.7	13.25	36.61	26.76
	(1,4)	32.76	36.42	21.66	5.83	16.41	18.23	26.79	13.24	32.59	23.41
	(1,5)	37.53	40.84	23.7	8.68	23.45	12.71	31.02	15.77	43.81	30.43
	(2,3)	22.68	29.19	18.2	14.29	22.44	20.41	22.52	17.72	25.95	20.58
	(2,4)	21.73	30.46	18.27	13.26	21.27	22.03	23.21	19.27	26.47	17.65
	(2,5)	27.79	39.77	21.68	10.85	21.92	18.08	24.9	18.91	28.94	21.67
	(3,4)	21.9	25.33	20.41	21.69	25.34	19.98	26.05	18.48	33.14	17.37
	(3,5)	19.05	37.51	21.12	17.5	22.69	24.91	27	16.42	20.28	9.06
	(4,5)	15.57	30.52	14.16	18.66	19.45	23.96	22.11	14.91	18.41	8.89

Slope:

w_1		$i \in I_2$									
		(1,2)	(1,3)	(1,4)	(1,5)	(2,3)	(2,4)	(2,5)	(3,4)	(3,5)	(4,5)
$c \in I_2$	(1,2)	-0.1	-0.11	-0.02	0.07	0	0.01	-0.05	0.01	-0.09	-0.05
	(1,3)	-0.08	-0.09	-0.01	0.06	0	0	-0.05	0.02	-0.07	-0.03
	(1,4)	-0.07	-0.1	-0.02	0.05	0	0	-0.04	0.02	-0.05	-0.01
	(1,5)	-0.09	-0.12	-0.03	0.04	-0.03	0.02	-0.06	0	-0.11	-0.05
	(2,3)	-0.03	-0.13	0	0.02	-0.05	-0.02	-0.04	0	-0.04	-0.01
	(2,4)	-0.02	-0.11	0	0.02	-0.03	-0.03	-0.04	-0.01	-0.04	0.01
	(2,5)	-0.05	-0.15	-0.02	0.03	-0.03	0	-0.04	0	-0.04	-0.01
	(3,4)	-0.03	-0.11	-0.03	-0.06	-0.11	-0.02	-0.1	-0.01	-0.15	0.03
	(3,5)	0	-0.17	-0.02	0	-0.04	-0.05	-0.07	0	0.01	0.09
	(4,5)	0.04	-0.18	0.04	-0.02	-0.02	-0.07	-0.05	0.03	0.04	0.14

B.2 Tables of Prediction Error Variance (in squared millimeters)

$var[\epsilon\|c,i]$		$i \in I_2$									
		(1,2)	(1,3)	(1,4)	(1,5)	(2,3)	(2,4)	(2,5)	(3,4)	(3,5)	(4,5)
$c \in I_2$	(1,2)	25.9	11	8.8	9.7	6	10.6	2.7	4.5	24.5	13.4
	(1,3)	26.5	11.3	8.9	10	6	10.7	2.6	4.3	24.9	13.7
	(1,4)	26.5	9.9	8.8	10.1	6	10.7	2.6	4.3	25.6	14
	(1,5)	26.1	10.1	8.7	11.1	5.7	10.4	2.2	4.5	23.1	13.3
	(2,3)	28.5	10.2	9	11.4	5.1	10.5	3	4.5	26.6	14.2
	(2,4)	28.8	10.4	9	11.3	5.6	10.3	3	4.4	26.6	14.1
	(2,5)	28.3	10.8	8.9	11.4	5.8	10.7	3.2	4.5	26.8	14.2
	(3,4)	28.7	13	8.8	10.9	4.2	10.6	1.9	4.5	23.7	14.1
	(3,5)	28.9	11.6	8.9	11.6	5.8	10.3	3	4.5	27.2	13.4
	(4,5)	28.7	11.7	8.8	11.5	5.9	10.1	3.3	4.4	27	12.2

	$c \in I_2$									
	(1,2)	(1,3)	(1,4)	(1,5)	(2,3)	(2,4)	(2,5)	(3,4)	(3,5)	(4,5)
$var[\epsilon\|c]$:	11.7	11.9	11.8	11.5	12.3	12.4	12.5	12	12.5	12.4

B.3 Plain Estimation and Variance (No Regression)

	$i \in I_2$									
	(1,2)	(1,3)	(1,4)	(1,5)	(2,3)	(2,4)	(2,5)	(3,4)	(3,5)	(4,5)
$\mathbb{E}[d_i^-]$:	19.1	16.9	17.7	16.6	17.2	18.1	18.1	17.5	21.6	19.6

	$i \in I_2$									
	(1,2)	(1,3)	(1,4)	(1,5)	(2,3)	(2,4)	(2,5)	(3,4)	(3,5)	(4,5)
$var[\epsilon \mid i]$:	28.9	14.9	9	11.6	6	10.7	3.6	4.5	27.2	14.2

$$var[\epsilon] = 13.1$$

B.4 Maximum Difference Between $P(d_i^- < d \mid d < d_i^+)$ and $P(d_i^- < d)$

The following table contains the maximum difference between $P(d_i^- < d \mid d < d_i^+)$ and $P(d_i^- < d)$, given that $d \geq \texttt{Min}[d_i^+]$. These probabilities were calculated using the Φ-function, as defined in section 5.3. The maximum approximation error of the Φ function is added to the entries in this table.

		Estimation Procedure	
		regression:	plain estimation:
$i \in I_2$	(1,2)	$1.5 \cdot 10^{-7}$	$1.5 \cdot 10^{-7}$
	(1,3)	$1.5 \cdot 10^{-7}$	$1.5 \cdot 10^{-7}$
	(1,4)	$1.5 \cdot 10^{-7}$	$1.5 \cdot 10^{-7}$
	(1,5)	$1.5 \cdot 10^{-7}$	$1.5 \cdot 10^{-7}$
	(2,3)	$1.5 \cdot 10^{-7}$	$1.5 \cdot 10^{-7}$
	(2,4)	$1.5 \cdot 10^{-7}$	$1.5 \cdot 10^{-7}$
	(2,5)	$1.5 \cdot 10^{-7}$	$1.5 \cdot 10^{-7}$
	(3,4)	$3.5 \cdot 10^{-7}$	$2.7 \cdot 10^{-7}$
	(3,5)	$2.1 \cdot 10^{-7}$	$2.0 \cdot 10^{-7}$
	(4,5)	$5.0 \cdot 10^{-4}$	$3.2 \cdot 10^{-4}$

C Reachable Area Regression (3 Fingers, $N = 15$)

C.1 Tables of Ellipse Parameter Estimation Coefficients

The following tables contain the coefficients for each separate ellipse parameter regression model. Such a model can be written as follows:

$$y(x) = w_0 + w_1 \cdot x + w_2 \cdot x^2$$

The input variable of this regression model is the distance between the stationary fingers (f_a,f_b), divided by their maximum (estimated) distance $d^+_{(a,b)}$. A table is given for each combination of two stationary fingers and third dynamic finger f_c. The table that belongs to these three fingers is indexed by 'f_a f_b f_c'.

f_1 f_2 f_3 :

	w_0	w_1	w_2
θ	1.12	-1.39	0.94
d	0.35	0.38	-0.16
φ	-0.54	0.46	0.19
r_x	0.14	0.69	-0.69
r_y	0.17	0.15	-0.13

f_1 f_2 f_4 :

	w_0	w_1	w_2
θ	0.92	-0.12	0.1
d	0.46	0.16	-0.08
φ	-0.15	-1.11	1.59
r_x	0.11	1.12	-1
r_y	0.15	0.37	-0.26

f_1 f_2 f_5 :

	w_0	w_1	w_2
θ	1.24	-0.52	0.41
d	0.52	0.18	-0.13
φ	0.19	-2.21	2.62
r_x	0.56	-0.23	-0.09
r_y	0.33	-0.15	0.13

f_1 f_3 f_2 :

	w_0	w_1	w_2
θ	-2.14	2.65	-1.3
d	0.24	0.17	-0.07
φ	-0.67	1.64	-2.26
r_x	0.18	0.25	-0.28
r_y	0.16	0.14	-0.17

f_1 f_3 f_4 :

	w_0	w_1	w_2
θ	1.15	-1.16	0.53
d	0.21	0.36	-0.09
φ	-0.45	0.68	-0.43
r_x	0.06	0.64	-0.58
r_y	0.07	0.23	-0.18

f_1 f_3 f_5 :

	w_0	w_1	w_2
θ	1.09	-0.13	-0.16
d	0.37	0.22	-0.15
φ	2.34	-10.07	9.61
r_x	0.17	0.53	-0.55
r_y	0.29	-0.21	0.1

f_1 f_4 f_2 :

	w_0	w_1	w_2
θ	-1.65	1.2	-0.64
d	0.41	-0.15	0.08
φ	0.43	5.16	-6.76
r_x	0.24	0.06	-0.12
r_y	0.3	0.26	-0.41

f_1 f_4 f_3 :

	w_0	w_1	w_2
θ	-1.53	1.97	-1.04
d	0.15	0.42	-0.15
φ	-1.9	8.51	-8.58
r_x	0.21	0.25	-0.32
r_y	0.12	0.08	-0.06

f_1 f_4 f_5 :

	w_0	w_1	w_2
θ	1.67	-2.82	1.75
d	0.11	0.67	-0.34
φ	-1.26	2.61	4.08
r_x	0.18	0.09	-0.18
r_y	0.07	0.22	-0.2

f_1 f_5 f_2 :

	w_0	w_1	w_2
θ	-1.87	1.11	-0.64
d	0.45	-0.14	0.07
φ	-1.15	6.71	-8.34
r_x	0.16	0.77	-0.77
r_y	0.17	0.26	-0.2

f_1 f_5 f_3 :

	w_0	w_1	w_2
θ	-1.5	0.84	-0.25
d	0.33	0.18	-0.07
φ	-0.41	1.37	-1.63
r_x	0.17	0.69	-0.71
r_y	0.14	0.24	-0.24

f_1 f_5 f_4 :

	w_0	w_1	w_2
θ	-1.37	1.53	-0.77
d	0.2	0.27	-0.02
φ	0.04	-0.02	-0.19
r_x	0.16	0.35	-0.39
r_y	0.08	0.19	-0.18

f_2 f_3 f_1 :

	w_0	w_1	w_2
θ	1.75	0.82	-0.73
d	1.06	0.17	-0.17
φ	-0.16	1.6	-1.32
r_x	1.4	-0.22	-0.08
r_y	0.62	0.5	-0.43

f_2 f_3 f_4 :

	w_0	w_1	w_2
θ	1.01	-0.6	0.48
d	0.6	0.1	-0.05
φ	1.49	-7.87	5.97
r_x	0.48	0.15	-0.39
r_y	0.15	0.67	-0.54

f_2 f_3 f_5 :

	w_0	w_1	w_2
θ	1.09	-0.34	0.26
d	0.81	0.12	-0.06
φ	1.15	-10.58	8.31
r_x	0.79	-1.07	0.68
r_y	0.27	1.35	-1.25

$f_2\ f_4\ f_1$:

	w_0	w_1	w_2
θ	1.84	0.48	-0.43
d	0.76	0.27	-0.21
φ	1.75	-8.36	5.44
r_x	1.43	-2.21	1.35
r_y	0.18	2.18	-1.58

$f_2\ f_4\ f_3$:

	w_0	w_1	w_2
θ	-1.62	0.49	-0.22
d	0.43	-0.03	-0.11
φ	-0.82	6.6	-11.41
r_x	0.51	-0.56	0.25
r_y	0.19	0.19	-0.2

$f_2\ f_4\ f_5$:

	w_0	w_1	w_2
θ	0.85	-0.53	0.42
d	0.35	0.6	-0.33
φ	-2.25	6.73	-0.74
r_x	0.36	-0.31	0.13
r_y	0.05	0.76	-0.68

$f_2\ f_5\ f_1$:

	w_0	w_1	w_2
θ	2	0.09	-0.27
d	0.76	-0.04	-0.1
φ	-0.19	3.35	-4.83
r_x	0.51	0.92	-1.11
r_y	0.26	0.55	-0.24

$f_2\ f_5\ f_3$:

	w_0	w_1	w_2
θ	-1.69	-0.54	0.35
d	0.41	0.09	-0.19
φ	-0.09	-0.58	0.75
r_x	0.19	0.14	-0.2
r_y	0.18	0.03	-0.03

$f_2\ f_5\ f_4$:

	w_0	w_1	w_2
θ	-1.45	0.6	-0.11
d	0.27	0.22	-0.17
φ	0.31	-0.75	-0.27
r_x	0.11	0.44	-0.45
r_y	0.08	0.27	-0.27

$f_3\ f_4\ f_1$:

	w_0	w_1	w_2
θ	1.52	0.96	-0.74
d	1.25	0.3	-0.09
φ	-0.17	1.31	-0.97
r_x	1.91	-0.08	-0.3
r_y	1.01	0.23	-0.29

$f_3\ f_4\ f_2$:

	w_0	w_1	w_2
θ	1.8	1.23	-1.05
d	0.83	-0.17	0.11
φ	-0.95	6.19	-6.68
r_x	0.7	0.84	-1.17
r_y	0.5	-0.27	0.41

$f_3\ f_4\ f_5$:

	w_0	w_1	w_2
θ	0.51	0.59	-0.46
d	0.56	0.6	-0.3
φ	-4.07	14.69	-8.88
r_x	0.45	-0.09	-0.11
r_y	0.09	1	-0.89

$f_3\ f_5\ f_1$:

	w_0	w_1	w_2
θ	1.97	-0.55	0.3
d	1.06	-0.36	0.23
φ	0.31	-0.1	-0.06
r_x	1.19	-0.55	0.33
r_y	0.63	0.01	-0.02

$f_3\ f_5\ f_2$:

	w_0	w_1	w_2
θ	1.83	1.45	-1.05
d	0.46	0.38	-0.21
φ	-1.38	8.39	-4.91
r_x	0.87	-1.48	0.9
r_y	0.09	1.1	-0.83

$f_3\ f_5\ f_4$:

	w_0	w_1	w_2
θ	-1.37	-0.56	0.37
d	0.32	-0.07	0.02
φ	2.05	-4.53	1.5
r_x	0.11	0.09	-0.1
r_y	0.1	0.13	-0.16

$f_4\ f_5\ f_1$:

	w_0	w_1	w_2
θ	1.71	0.15	-0.2
d	1.41	0.18	-0.13
φ	-0.08	0.98	-0.73
r_x	2.19	-1.03	0.52
r_y	1.13	-0.19	0.09

$f_4\ f_5\ f_2$:

	w_0	w_1	w_2
θ	2	0.38	-0.33
d	1.08	-0.08	-0.02
φ	-0.3	3.14	-2.97
r_x	1.06	0.36	-0.67
r_y	0.75	-0.48	0.42

$f_4\ f_5\ f_3$:

	w_0	w_1	w_2
θ	2.2	0.69	-0.55
d	0.71	0.32	-0.21
φ	0.61	0.27	-0.18
r_x	0.5	0.47	-0.58
r_y	0.29	0.26	-0.26

C.2 Expected Coverage of the Estimated Ellipses

In the following tables, the combination of two stationary fingers $(f_a, f_b) \in I_2$ and third finger f_c is indexed by (f_a, f_b, f_c). In terms of the normalized space, the expected coverage is calculated as follows:

$$\frac{\int_{\lambda=0}^{\infty} \{2\pi\lambda \cdot \texttt{Coverage}(\lambda)^2\}}{\texttt{Area}(P')}$$

	Expected Coverage:		Expected Coverage:
(f_1, f_2, f_3):	77.0%	(f_2, f_4, f_1):	84.2%
(f_1, f_2, f_4):	78.9%	(f_2, f_4, f_3):	79.7%
(f_1, f_2, f_5):	79.7%	(f_2, f_4, f_5):	68.2%
(f_1, f_3, f_2):	81.6%	(f_2, f_5, f_1):	82.4%
(f_1, f_3, f_4):	72.1%	(f_2, f_5, f_3):	75.2%
(f_1, f_3, f_5):	73.4%	(f_2, f_5, f_4):	71.9%
(f_1, f_4, f_2):	83.7%	(f_3, f_4, f_1):	87.3%
(f_1, f_4, f_3):	78.4%	(f_3, f_4, f_2):	79.2%
(f_1, f_4, f_5):	71.2%	(f_3, f_4, f_5):	65.3%
(f_1, f_5, f_2):	82.3%	(f_3, f_5, f_1):	85.3%
(f_1, f_5, f_3):	79.7%	(f_3, f_5, f_2):	77.5%
(f_1, f_5, f_4):	74.4%	(f_3, f_5, f_4):	59.4%
(f_2, f_3, f_1):	86.1%	(f_3, f_5, f_1):	88.5%
(f_2, f_3, f_4):	67.9%	(f_3, f_5, f_2):	81.5%
(f_2, f_3, f_5):	75.2%	(f_3, f_5, f_3):	75.7%

Overall expected coverage: 77.8%

C.3 Table of Coverage Estimation Coefficients and Error Properties

In the following tables, the combination of two stationary fingers $(f_a, f_b) \in I_2$ and third finger f_c is indexed by (f_a, f_b, f_c). Note that the expected prediction error ϵ is zero when the logistic model is constructed using the minimum squared error condition. The values in the fourth column are the variances of the estimation error, where the coverage ranges from zero to one. In the fifth column, the standard deviations (times two) are given as percentages of the coverage. The estimation model is written as follows:

$$Coverage(\lambda) = \frac{1}{1 + e^{-g(\lambda)}}, \qquad \text{where: } g(\lambda) = w_0 + w_1 \cdot \lambda$$

	w_0	w_1	$var[\epsilon \mid f_a, f_b, f_c]$	$2 \cdot \sigma$
(f_1, f_2, f_3):	4.964	-4.954	0.0002	2.8%
(f_1, f_2, f_4):	6.677	-6.704	0.00012	2.2%
(f_1, f_2, f_5):	6.602	-6.716	0.00057	4.8%
(f_1, f_3, f_2):	6.001	-6.183	0.00057	4.8%
(f_1, f_3, f_4):	5.045	-5.265	0.00008	1.8%
(f_1, f_3, f_5):	5.68	-5.935	0.00011	2.1%
(f_1, f_4, f_2):	9.27	-9.383	0.00003	1.1%
(f_1, f_4, f_3):	6.977	-7.134	0.0001	2%
(f_1, f_4, f_5):	4.228	-4.58	0.00019	2.7%
(f_1, f_5, f_2):	8.472	-8.779	0.00006	1.5%
(f_1, f_5, f_3):	6.785	-7.083	0.00014	2.4%
(f_1, f_5, f_4):	4.412	-4.756	0.00026	3.2%
(f_2, f_3, f_1):	9.953	-9.642	0.00005	1.4%
(f_2, f_3, f_4):	3.517	-3.55	0.00021	2.9%
(f_2, f_3, f_5):	4.724	-4.726	0.00047	4.3%
(f_2, f_4, f_1):	9.371	-9.555	0.00008	1.8%
(f_2, f_4, f_3):	4.857	-4.997	0.00063	5%
(f_2, f_4, f_5):	4.512	-4.758	0.00051	4.5%
(f_2, f_5, f_1):	8.171	-8.635	0.00012	2.2%
(f_2, f_5, f_3):	4.431	-4.864	0.00057	4.8%
(f_2, f_5, f_4):	4.677	-5.131	0.00043	4.2%
(f_3, f_4, f_1):	11.035	-11.214	0.0003	3.5%
(f_3, f_4, f_2):	5.986	-6.007	0.00076	5.5%
(f_3, f_4, f_5):	2.757	-2.867	0.00073	5.4%
(f_3, f_5, f_1):	8.121	-8.971	0.00059	4.8%
(f_3, f_5, f_2):	6.92	-7.817	0.00027	3.3%
(f_3, f_5, f_4):	1.751	-2.433	0.00022	3%
(f_4, f_5, f_1):	10.451	-11.098	0.00059	4.9%
(f_4, f_5, f_2):	6.829	-7.391	0.00051	4.5%
(f_4, f_5, f_3):	4.807	-5.278	0.00051	4.5%

The general estimation error properties:

$$var[\epsilon] = 0.00033$$
$$2 \cdot \sigma = 3.6\%$$

C.4 Several Results of the Estimated Reachable Regions

Each of the following figures contains three images. The first image represents the area that was measured for some combination of stationary fingers and third finger (of which the reachable area is measured and estimated). The second image shows the ellipse that is fitted on the reachable area. Finally, the third finger contains the estimated ellipse of the reachable area, which is deduced from the model that is based on all the fitted ellipses of the respective finger combination. The red gradient represents the coverage that is deduced from the respective coverage estimation model, which ranges from red (full coverage) to transparent (zero coverage). In the gradient, the standard deviation is not subtracted from the coverage estimation model.

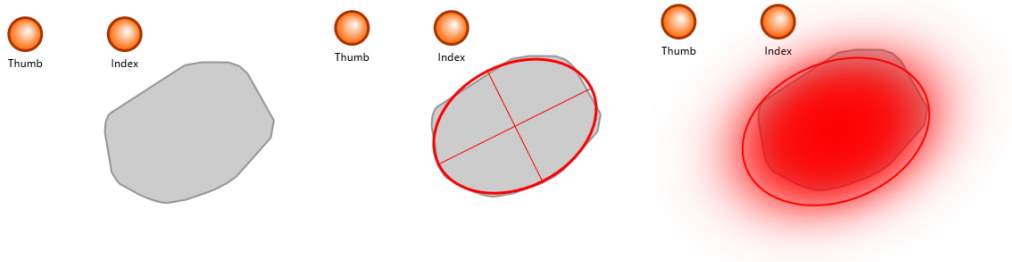


Figure C.1: The estimated region of some measurement of the finger combination f_1, f_2, f_3

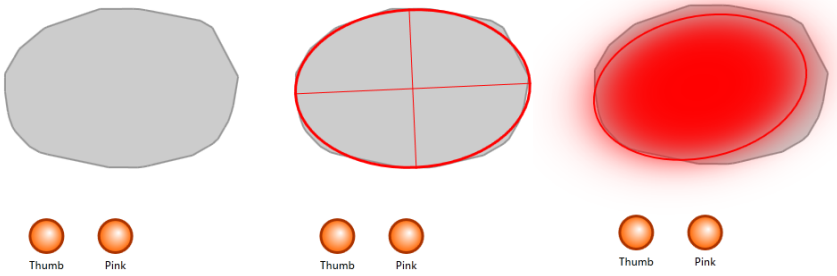


Figure C.2: The estimated region of some measurement of the finger combination f_1, f_5, f_3

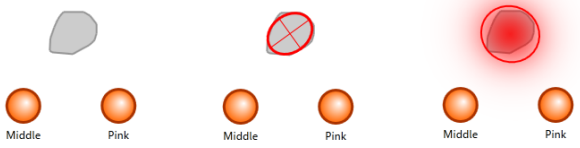


Figure C.3: The estimated region of some measurement of the finger combination f_3, f_5, f_4