

University of Alabama in Huntsville

**LOUIS**

---

Theses

UAH Electronic Theses and Dissertations

---

2016

## Designing natural hand gestures for prolonged human computer interaction

Vaidyanath Areyur Shanthakumar

Follow this and additional works at: <https://louis.uah.edu/uah-theses>

---

### Recommended Citation

Areyur Shanthakumar, Vaidyanath, "Designing natural hand gestures for prolonged human computer interaction" (2016). *Theses*. 638.

<https://louis.uah.edu/uah-theses/638>

This Thesis is brought to you for free and open access by the UAH Electronic Theses and Dissertations at LOUIS. It has been accepted for inclusion in Theses by an authorized administrator of LOUIS.

**DESIGNING NATURAL HAND GESTURES FOR  
PROLONGED HUMAN COMPUTER INTERACTION**

by

**VAIDYANATH AREYUR SHANTHAKUMAR**


**A THESIS**

Submitted in partial fulfillment of the requirements  
for the degree of Master of Science  
in  
The Department of Computer Science  
to  
The School of Graduate Studies  
of  
The University of Alabama in Huntsville

**HUNTSVILLE, ALABAMA**

**2016**

In presenting this thesis in partial fulfillment of the requirements for a master's degree from The University of Alabama in Huntsville, I agree that the Library of this University shall make it freely available for inspection. I further agree that permission for extensive copying for scholarly purposes may be granted by my advisor or, in his/her absence, by the Chair of the Department or the Dean of the School of Graduate Studies. It is also understood that due recognition shall be given to me and to The University of Alabama in Huntsville in any scholarly use which may be made of any material in this thesis.

  
\_\_\_\_\_  
Vaidyanath Areyur Shanthakumar

4/4/2017  
(date)



## ABSTRACT

School of Graduate Studies  
The University of Alabama in Huntsville

Degree Master of Science College/Dept. Science/Computer Science

Name of Candidate Vaidyanath Areyur Shanthakumar

Title Designing Natural Hand Gestures for Prolonged Human Computer  
Interaction

The use of hand gestures is a natural and intuitive way to interact with data. Hand gestures have been proven to be a promising input modality in human computer interaction. However, there are two major limitations in traditional gesture recognition systems: (1) gesture vocabularies usually contain unnatural hand gestures, and (2) a high level of physical exertion usually occurs when interacting with the system for a prolonged amount of time. In this thesis, we design six natural hand gestures that are generic in their use and propose an adaptive hand model that uses a vector based angular velocity approach to recognize the six hand gestures.

We report two user studies. The first study involves participants playing a video game using two of the six hand gestures with their hands in a natural supported position, with their hands in a mid-air position, and using a keyboard for 30 minutes each. The endurance time and perceived level of physical exertion in the game are considered. The results show that the hand gestures performed in a resting position show exertion levels similar to that of the keyboard. The exertion levels when gestures are performed in mid-air are higher compared to that of the keyboard. The second study

involves participants playing a video game consisting of three mini-games requiring the use of subsets of the six gestures in different combinations. The accuracy and reliability of hand gestures are considered based on a questionnaire and study of a video recording. The results show that the gestures have different accuracy levels, with a conflict between gestures likely leading to low accuracy levels for recognizing those gestures.

Abstract Approval: Committee Chair



*Dr. Chao Peng*

Department Chair



*Dr. Heggere S Ranganath*

Graduate Dean



*Dr. David K Berkowitz*

## ACKNOWLEDGMENTS

First and foremost, I would like to express my sincere gratitude to my advisor, mentor, Dr. Chao Peng for his continuous support and guidance. I would like to thank him endlessly for always being there when I needed help, mentoring me, motivating me at the right times, encouraging me to strive for the best, pushing my limits higher, making me believe that I can do better and pushing me to do better. I could not have imagined of having a better person to mentor me like how Dr. Peng did.

I would like to thank Dr. Ranganath S Heggere for guiding me in the right direction and helping me take the right decisions.

Also, I would like to sincerely thank Dr. Jeffrey Hansberger, for being a great supervisor, again a person who was always there when I needed him, for his constant efforts and valuable insights with which the research was conducted without any hurdles, hitches or hindrance. I would like to thank the US Army Research Lab (ARL) for completely funding and supporting this research. Also, I would like to thank Lizhou Cao for his substantial help in developing the video game that was used in the study. I would like to thank Sarah and Victoria for their valuable help in data collection and analysis. My sincere thanks to Dr. Shannon Mathis, department of Kinesiology for her precious contribution towards the research. I would also like to thank Dr. Timothy Newman for his valuable feedback that helped me make this a better thesis.

I would like to thank my friend Garima for all her help and support. Last but not the least, I wish to thank my parents for their unconditional love and support with which this became a lot easier and also my friends Suhas, Balaji, Kiran, Subin, Suggu, Sindhu, Manikanta, Shabu, Ayush, Kyle, Durgi, Sowmya and Srishti for their moral support and encouragement at different times.



## TABLE OF CONTENTS

List of Figures	x
List of Tables	xii
Chapter	
<b>1 Introduction</b>	<b>1</b>
<b>2 Literature Review</b>	<b>5</b>
<b>3 Hand model and Gesture Vocabulary</b>	<b>12</b>
3.1 Adaptive Hand Model . . . . .	12
3.1.1 Degree of Finger Independence . . . . .	14
3.1.2 Articulation . . . . .	15
3.2 Gesture Vocabulary . . . . .	18
3.2.1 Swipe . . . . .	20
3.2.2 Stop . . . . .	24
3.2.3 Come . . . . .	25
3.2.4 Go . . . . .	28
3.2.5 Pinch . . . . .	29
3.2.6 Drop in and trash out . . . . .	31
3.3 How Our Approach Is Better? . . . . .	33

<b>4</b>	<b>User Experiment To Study The Gorilla Arm Effect</b>	<b>35</b>
4.1	Overview . . . . .	35
4.2	Mid-air gestures . . . . .	37
4.3	Experiment . . . . .	40
4.4	Apparatus . . . . .	41
4.5	Procedure . . . . .	43
4.6	Results . . . . .	44
4.7	Discussion . . . . .	49
<b>5</b>	<b>The reliability of hand gesture enabled input in video games: A study</b>	<b>52</b>
5.1	Game . . . . .	53
5.1.1	Spring . . . . .	54
5.1.2	Winter . . . . .	55
5.1.3	Fall . . . . .	56
5.1.4	Summer . . . . .	59
5.2	Experiment . . . . .	60
5.3	Apparatus . . . . .	61
5.4	Procedure . . . . .	62
5.5	Results . . . . .	64
5.6	Discussion . . . . .	68
<b>6</b>	<b>Conclusions</b>	<b>72</b>

## LIST OF FIGURES

FIGURE	PAGE
3.1 The full articulation of a human right hand . . . . .	13
3.2 The upper body and hand articulations. . . . .	16
3.3 The first row images (a-d) show static gestures. In (b) and (d), their hands and fingers have very similar postures, but the meanings are different. The second row images (e-h) show dynamic gestures. The meanings are presented by the motions of hands and fingers. Hand gesture examples are excerpted from Internet. The sources are: (a) [1], (b) [2], (c) [3], (d) [4], (e) [5], (f) [6], (g) [7], (h) [8] . . . . .	19
3.4 The swipe gesture. . . . .	21
3.5 The stop gesture. . . . .	24
3.6 The come gesture. . . . .	26
3.7 The go gesture. . . . .	28
3.8 The pinch posture. . . . .	30
3.9 The drop-in and trash-out gestures with the hand in a pinch posture. . . . .	32
4.1 Example postures for a mid-air gesture (a) and a supported gesture (b). The posture in (a) is a common arm position for vertical touchscreen. The posture (b) is the arm position for gestures during face-to-face communications. . . . .	37
4.2 A screenshot of the mini-game for the gorilla arm experiment . . . . .	43
4.3 Mean time spent out of a possible 30 minutes for each condition. Error bars represent standard deviation. . . . .	45
4.4 The mean subjective Borg ratings for physical exertion across conditions (0=Nothing at all, 10=Impossible). Error bars represent standard error of the mean. . . . .	48

5.1	Screenshot of the spring minigame. . . . .	55
5.2	Screenshot of the winter mini-game . . . . .	57
5.3	Screenshot of the fall minigame. . . . .	58
5.4	Screenshot of the summer minigame. . . . .	60
5.5	An example frame of the user study showing the user performs the stop gesture in the winter game to make the ball slide under the ice block. . . . .	64
5.6	Gesture accuracy scores for each gesture according to accurate hits, misses, and errors. . . . .	66
5.7	The mean UEQ scores for each scale across all 3 games and compared with UEQ scores from other studies. . . . .	67
5.8	Mean UEQ scores for each scale for each game type, Summer, Fall, and Winter. Scales are in the range of [-3,3], scores bigger than 0.8 represent a positive evaluation, scores smaller than -0.8 represent a negative evaluation, and scores between -0.8 and 0.8 represent a neutral evaluation. . . . .	67
A.1	Institutional Review Board approval letter for the user experiment to study the gorilla arm effect. . . . .	76
A.2	Institutional Review Board approval letter for the reliability of hand gesture enabled input in video games: A study. . . . .	77
A.3	The UEQ Analysis Report obtained from the results of the reliability of hand gesture enabled input in video games: A study . . . . .	78

## LIST OF TABLES

TABLE	PAGE
3.1 The thresholds of the parameters used in the swipe gesture. . . . .	22
3.2 Parameters and threshold ranges used in the stop gesture. . . . .	25
3.3 Parameters and threshold ranges used in the come gesture. . . . .	26
3.4 Parameters and threshold ranges used in the go gesture. . . . .	29
3.5 Parameters and threshold ranges used in the pinch gesture. . . . .	31
3.6 Parameters and threshold ranges used in the drop-in and trash-out gestures. . . . .	32
4.1 Parameter estimates from the GEE model for the perceived exertion across the interaction conditions. . . . .	48
4.2 Multiple comparisons and mean differences in perceived exertion by interaction types. . . . .	49

## CHAPTER 1

### INTRODUCTION

In modern life, advanced computing technology has made many positive impacts. People use interactive products everyday to communicate, learn, work, and entertain. Products including video games, web applications, and data analytics tools, all require natural user interfaces. Although traditional control devices like keyboards, mice, joysticks, and touch screens provide convenient access to information, they do not offer interactions as natural as interactions people use in the real world.

Human interaction with the real world is multimodal [9], and one important part of that is the use of hands in everyday body language (e.g., extending fingers to express numbers from one to five). Development of natural human interactions with computing devices has increased quickly in recent years. Human computer interaction relies upon intuitiveness (the power or faculty of attaining to direct knowledge or cognition without evident rational thought or inference) of the users [10]. A user with no physical disabilities may find it intuitively easy to be able to manipulate virtual data the way a person interacts with physical objects. For example, doctors may wave hands in the air to navigate through patients' medical records on the computer

screen just like flipping papers. *Motion control*, in the context of human-centered interactive systems, means the use of a user's hands, arms, or body to command computers. Motion-controlled systems offer an intuitive way to play, interact, learn, etc. Naturalness in controlling a video game environment has been shown to increase immersion and enjoyment [11].

In terms of kinesics, hands are a very expressive part of our anatomy and can express many body language signs [12]. Compared to full-body gesture interactions (e.g., [13,14]), hand gestures cause less physical exertion and avoid injuries by limiting unusual body movements. Hands are usually the preference for prolonged interactions. If pushing buttons was the way of interaction in the past and finger-tapping on touchscreens is the present, we believe the future will be using hand gestures to manipulate virtual worlds and to control immersive display devices. Decoupling hands from a touch-sensing surface will increase the gesture-language variety and therefore may increase functionality. This may also avoid the imprecision problem caused by *fat finger* [15] or *occlusion* [16] observed with restricted touch spaces, such as the screen of a smartwatch [17].

In this thesis, we propose a simple and adaptive hand model with respect to hand biomechanics. Our hand model can be used for both motion-matching and posture-matching based recognition algorithms. We also propose a vocabulary of natural gestures that are implemented with angular-velocity methods. The hand model and gesture vocabulary are implemented using the commercial motion capture system Perception Neuron. It includes a wearable suit with networked IMU (Inertial Measurement Unit) sensors, which return accurate positions for articulated joints

on the chest, arms, hands, and fingers. Our gesture framework is generic and can be easily applied to various types of interactive applications such as manual image categorization, data visual analytics, and 3D navigations.

Real-world tasks on modern computing systems can be complex and involve interactions with a large quantity of data. They usually require frequent repetition of a few gestures or their combinations. Some tasks also require the user to interact with the system for a prolonged amount of time. For example, an analyst at a security agency might work all day manipulating and sorting images from security cameras.

The discomfort in the arms caused by performing hand gestures for a prolonged amount of time is often referred to as the *gorilla arm syndrome* [18]. We also report an experimental study to estimate the physical exertion (discomfort) associated with performing hand gestures for a prolonged amount of time by comparing the exertion (discomfort) levels across keyboard, mid-air hand gestures, and our supported natural hand gestures (we call our newly developed hand gestures “supported gestures” as they can be comfortably performed by resting one’s hands on the arm-rest of a chair or one’s own lap). We also evaluate the accuracy of the gesture recognition and the users’ level of engagement during their interactions. Our experimental studies are set in a gaming environment with gamification for the gesture vocabulary.

The rest of this thesis is organized as follows. Chapter 2 reviews existing work related to hand gesture based interactions. Chapter 3 presents the adaptive hand model, gesture vocabulary, and describes our gesture recognition methods. Chapter 4 presents the details of an experiment conducted to evaluate the exertion levels from playing a video game using three different input conditions, namely the keyboard,



mid-air gestures and the supported gestures. Chapter 5 presents the gamification method and reliability as well as the engagement evaluations of the hand gestures within the gaming environment. Chapter 6 presents the conclusion.

## CHAPTER 2

### LITERATURE REVIEW

In this chapter, related work will be discussed.

When designing user interfaces for gaming or interactive systems, a natural way to interact with these systems is preferred. Hand gestures are natural interactions. In a hand gesture based interface, to support complex interactions, the design of hand gestures should be ergonomic [19–21], bimanual (performed with both hands), and pantomimic (meaningful without speech) [22]. Additionally, the design of hand gestures should be natural and intuitive [19], and the preferences of the users who would use the interface should also be considered while designing natural hand gestures [20, 21].

Performing hand gestures to manipulate data or play video games can be compared to gestures performed by sign language interpreters. Sign language interpreters generally use an extensive set of hand gestures to translate an oral speech or other kind of material to people who are deaf or suffer from hearing impairment. As Rempel et al. [23] indicated in their paper, sign language interpreters are experts in forming hand gestures and they often have to perform gestures continuously for a long period of time. Hence, most of them have experienced hand and arm fatigue during and

after performing gestures for long hours [24, 25]. Rempel et al. [23] presented that the most likely position to induce the least fatigue while performing hand gestures is with the arms at the lower chest level. The position that would most likely induce the greatest fatigue is with the arms stretched straight in front of the body. Ramos et al. [26] developed a biomechanical model and used it to measure the fatigue when the arms were held in four different positions. A Microsoft Kinect (a vision based motion capture device) was used to obtain the arm positions. Rohmert's equation [27], which measures endurance as a function of the applied force and the maximum force of the muscle, was used to estimate the impact of fatigue on endurance when the arms were held in a particular position. Ramos et al. found that the position that induced the least fatigue while performing hand gestures was when the arm formed a 90 degree angle with the forearm. The most fatiguing position found was when the arms were stretched straight and perpendicular to the torso.

The literature relevant to our work was found in two categories: (1) research in designing hand gestures for human computer interaction (i.e., [22, 23]), and (2) research in developing gesture recognition systems (i.e., [28, 29]). Gesture recognition systems usually contain vision based systems (i.e., [30, 31]) or wearable sensor based systems (i.e., [32, 33]).

An advantage of using a vision based motion capture device for gesture recognition is that there is no need to wear any form of sensors, unlike wearable motion capture devices (such as gloves). A survey on vision based gesture recognition approaches [30] mentioned that, in general, gesture recognition using a vision based motion capture device has three phases: detection, tracking, and recognition. The

gesture recognition system detects and segments hand movements, tracks positions and orientations of the hand, and then recognizes a particular action as a meaningful gesture or an unintentional movement.

Here, we discuss a few recent gesture recognition approaches that use vision based MoCap (Motion Capture) systems. Song et al. [34] detected gestures using a single stereo camera to reconstruct body postures in 3D and then build features that were fed into a classifier algorithm. Their approach recognized both static (isolated) and dynamic (continuous) gestures. An accuracy of 93.7% was achieved for static gesture recognition, as where an accuracy of 88.37% was achieved for dynamic gesture recognition. Wang et al. [29] proposed a gesture recognition approach in which hand shapes and their corresponding textures and depths were captured using a Microsoft Kinect and represented using superpixels. They used this superpixel representation to derive a distance metric that measured the dissimilarity between the hand gestures. Their approach was tested using three datasets: the public NTU (Nanyang Technological University's) hand digit dataset, the public ASL (American Sign Language) finger spelling dataset, and their own dataset with 10 gestures. The approach achieved accuracies of 99.6%, 75.8%, and 99.1%, respectively. In yet another study, Liu et al. [35] applied a rule-based approach on data combined from stereo images to recognize gestures. This system recognized seven motional hand gestures with an accuracy of 92% and 6 finger spelling hand gestures (showing numbers 0 through 5) with an accuracy of 93%. Lastly, Marin et al. [31] used a classifier algorithm for gesture recognition, which used data from a depth sensor along with a Leap Motion

device to capture images and motion data. Their approach was tested using a set of 10 ASL hand gestures and achieved an accuracy of over 90%.

Although high accuracies were seen in the approaches described above, an underlying disadvantage of vision based systems is that they may fail to provide accurate recognition results in the absence of stable lighting conditions or in cluttered room backgrounds [36]. Additionally, using cameras may also raise privacy issues [32].

Another set of gesture recognition systems involve the use of wearable sensor based motion capture devices. Some of the approaches found in the literature are as follows. Murakami et al. [37] fed hand and finger motion data captured using a VPLDataGlove into a recurrent neural network (RNN) to recognize Japanese sign language gestures, including hate, skilled, unskilled, brother, and sister. Xu [28] used a feed forward neural network approach to recognize hand gestures using data collected from a “Data Glove” (commercially available MoCap glove) where the gestures were adopted in a driving training system. Neto et al. [38] developed a gesture recognition system based on a machine learning algorithm that utilized joint angle measurement data obtained from a Cyber Glove II (also a commercially available MoCap glove). The system recognized 10 artificial hand gestures (e.g., linear motion in all axes and striking a shown pose) with an accuracy of 99.8%, and 30 natural hand gestures (e.g., thumbs up, thumbs down, and grasp) with an accuracy of 96.3%. Lu et al. [39] developed a framework that utilized a Bayes linear classifier and a dynamic time-warping algorithm for recognizing hand gestures using data from a custom forearm wearable device. The approach recognized 19 hand gestures for mobile device operations such as pick, swipe, and draw numbers from 0 through 9. Romaszewski et al. [40] used

a machine learning algorithm based approach to recognize natural hand gestures by re-sampling and interpolating data obtained from a DG5VHand glove and a Cyber Glove. The system recognized 22 hand gestures, including thumbs up, thumbs down, and knocking. Xu et al. [41] detected gestures using a smart watch. Some of the gestures include gestures that are performed using fingers, such as showing numbers from one to five, thumbs up, and pointing, as well as a few arm gestures such as pushing left/right and drawing a circle. Luzhnica et al. [32] presented a sliding window based approach for gesture recognition with the help of a custom glove. The motion data from the glove was collected and fed into different classification algorithms. The performance of each algorithm was evaluated to identify the best algorithm for each gesture. The corresponding best algorithms recognized a set of 31 natural hand gestures with an accuracy of 98.2%. Alavi et al. [33] proposed the use of a support vector machine algorithm or an artificial neural network (depending on the level of training of the user) for gesture recognition based on motion data collected using IMU sensors. Both the approaches were able to detect 6 different gestures (jab, uppercut, throw, lift block, sway) with an average accuracy of 98%.

Based on the literature reviewed in this chapter, we can infer that, firstly, there is a need for designing hand gestures that are both natural and intuitive, and can be performed with hands in positions that result in low physical exertion. Rempel et al. [23] and Ramos et al. [26] recommended “hands at chest level” to be the optimal position for minimizing fatigue while performing hand gestures.

Secondly, we disregard the use of vision based motion capture devices as they may provide inaccurate motion data due to varied lighting conditions or cluttered

backgrounds. It has also been observed that the vision based systems have poor performance in gesture recognition and are less reliable for applications that require recognition of human hand gestures [42, 43]. Vision based systems require the user's hands and arms to have direct line of sight with the cameras in order to maintain the accuracy and fidelity of the detection. Otherwise, systems such as the Leap Motion controller are unable to return the correct positions of finger joints or can even lose track of fingers completely, if the hand is in a position such that the palm is perpendicular to the field of view of the cameras present in the controller [42]. This means users may not have the freedom to keep their hands in a natural position when performing gestures. This could potentially lead to the loss of "naturalness" in human computer interaction. On the contrary, inertial measurement units (IMU) based MoCap devices can provide accurate and efficient hand and finger tracking capabilities [44]. These can be used for gesture recognition without having the drawbacks that vision based systems suffer.

Finally, most of the existing approaches for gesture recognition involve data collection, manual annotation, and feeding data into learning algorithms which may be laborious and cumbersome tasks. For example, Luzhnica et al. [32] used a procedure in which the participant was guided by a video tutorial on the right way how to perform the gesture, followed by instructions on getting ready to perform the gesture. A progress bar then started on the screen indicating the participant to perform the gesture in that time frame in which the data was recorded. This process took about 30-40 seconds per trial per gesture. As shown by this time in data collection for developing any learning based gesture recognition system, in order to train the

system effectively there is a need to collect data from greater number of trials from a particular participant. The same data collection procedure was to be repeated for other participants as well. If we assumed that data is to be collected from 30 participants each performing 20 trials of one gesture, then it would take a total of 5 hours for data collection. This data would require manual annotation, and to facilitate this, the data would have to be observed again (usually by playing the recorded video), and annotated. For the number of participants and their corresponding numbers of trial we had, it would take another 5 hours in the best case. Thus, we would require 10 hours of data collection and annotation, ignoring the time to feed this data into a learning algorithm for training to recognize one particular gesture. This makes it a laborious and cumbersome task. Taking all these details in to consideration, we propose an approach that uses a heuristic rule-based angular velocity method to recognize a set of six hand gestures that are designed to mimic natural hand gestures. This proposed approach does not require any machine learning algorithms to train the system. It is easy to understand and developer-friendly.



## CHAPTER 3

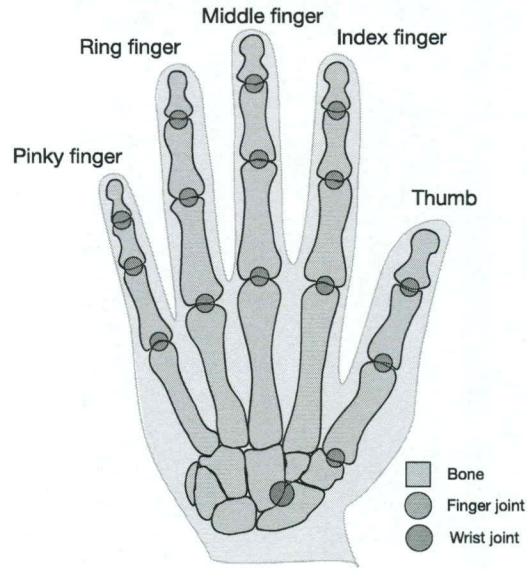
### HAND MODEL AND GESTURE VOCABULARY

In this chapter, we present the design of the adaptive hand model. We also describe the gesture vocabulary and the methods for hand gesture recognition using this adaptive hand model. The adaptive hand model is constructed with a subset of hand joints and a minimum number of vectors (bones formed by adjacent joints), and can be used with any wearable MoCap device without any modification. This adaptive hand model is used to implement a system that recognizes a variety of commonly used gestures, which are performed using both hands and arms.

Section 3.1 describes the design of the adaptive hand model. Section 3.2 defines a gesture vocabulary including a set of gestures that our approach recognizes in real time and also explains the process of extraction of angular velocities for a limited number of joints/sensors from the MoCap data.

#### **3.1 Adaptive Hand Model**

A human skeletal hand consists of 27 bones with proportional lengths to each other. The full articulation constructed from the skeletal hand is usually configured with 16 major joints including 15 joints on fingers and one joint at the wrist, as shown



**Figure 3.1:** The full articulation of a human right hand

in Figure 3.1. A joint is the junction between two bones. There are three joints on each finger. The five fingers are named: *thumb*, *index finger*, *middle finger*, *ring finger*, and *pinky finger*. However, many hand gestures do not require joint movements from all fingers. Also, some fingers have low degree of independence. For example, the movement of the pinky and ring fingers is usually affected by the movement of the middle finger.

We discuss the finger independence issues and then propose an adaptive hand model, which is a simpler hand articulation constructed with relatively independent fingers.

### 3.1.1 Degree of Finger Independence

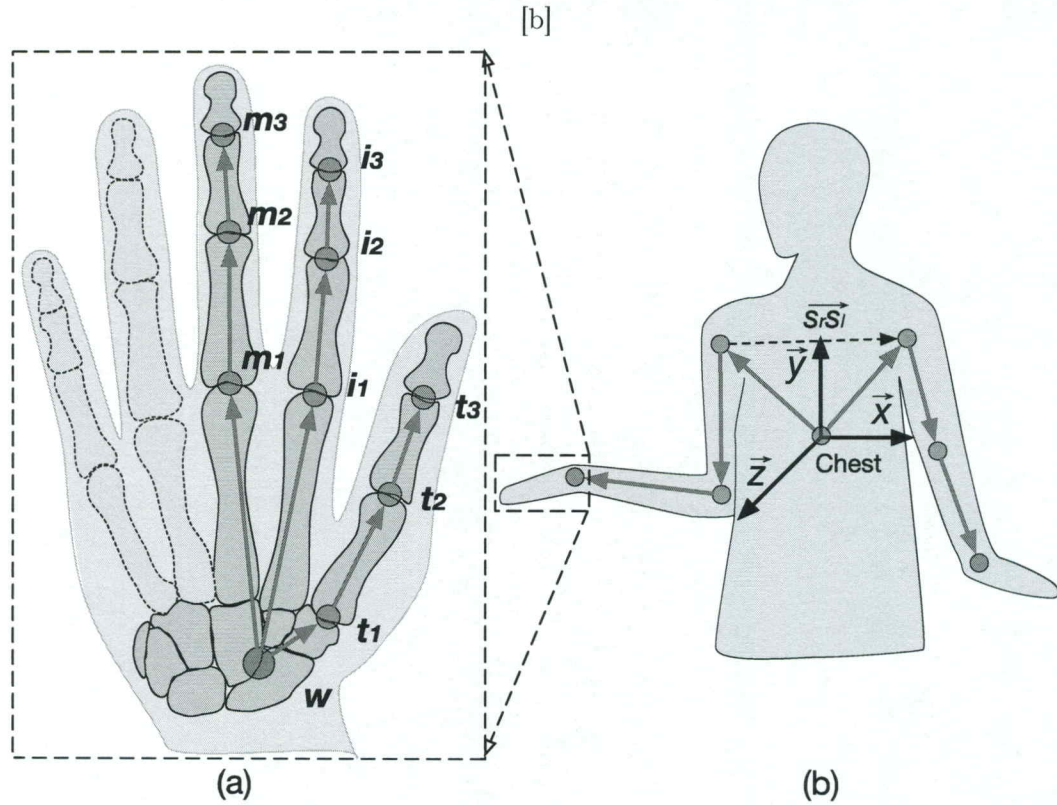
The *degree of finger independence* refers to the amount of muscle-controlled movement versus the amount of passive movement, coupled with the movement of other fingers [45]. While people can make various types of hand gestures to communicate, the kinematic structure of a finger results in limited motion capability. Horton et al. [46] studied the stiffness of one finger when it is being flexed or extended, and observed its effect on stretching other fingers due to the inter-tendon connections. The results of the study showed that the fingers of the hand are not equally independent. The thumb has a greater degree of independence than the other fingers. The index finger has the second most degree of independence. The middle finger has the third most degree of independence. The index, middle, ring, and pinky fingers move according to one degree of freedom with restricted independence. Flexing a finger may cause a significant passive movement on the finger next to it. The effect of such passive movement on other fingers decreases with the distance from it.

The thumb, index, and middle fingers are usually preferred by people to perform gestures because they have a higher degree of independence than the ring and pinky fingers. For example, when people use fingers to count numbers, they naturally start from the thumb or the index finger. The index finger is also the finger preferred for a pointing gesture and the thumbs-up is a natural gesture to show approval. Another example is the victory or peace sign gesture, which is usually performed by extending the index and middle finger from a closed fist. Unlike the index and middle fingers, the ring and pinky fingers are seldom used in natural gestures. This is

because they have a much lower degree of independence than the others. For example, the ring finger can only be extended to a limited degree when the other fingers form a fist. Hence the ring and pinky fingers are only used to make hand gestures in special circumstances. For instance, sign language interpreters may choose the ring and index fingers in conjunction with other fingers in order to deliver meanings and knowledge [23].

### 3.1.2 Articulation

As seen in Section 3.1.1, since the thumb, index finger and middle finger are the fingers with higher degree of independence, our hand model consists of the thumb, index finger, and middle finger only. Our hand model is adaptive to different hand proportions. Its articulation contains a set of joints denoted as  $J = \{w, t_j, i_j, m_j\}$ , where  $j$  is an integer index in the range of  $[1, 3]$ . As illustrated in Figure 3.2 (a),  $w$  represents the wrist joint,  $t_j$  represents the three joints of the thumb,  $i_j$  represents the three joints of the index finger, and  $m_j$  represents the three joints of the middle finger. We construct vectors from the joints. Some of the vectors created between joints can be said to represent the bones of the hand. In the Figure 3.2 (a),  $\vec{ab}$  is denoted as a vector from the joint  $a$  to the joint  $b$ . For example, the vector created between the proximal phalanx (the phalanx that is closest to the hand) and the intermediate phalanx (the phalanx that is second closest to the hand) of the index finger can be represented by the vector  $\vec{i_1 i_2}$ . Similarly, the vectors created between the wrist and the proximal phalanges of the index finger and the middle finger can be represented by  $\vec{w m_1}$  and  $\vec{w i_1}$ . These vectors define the plane of the palm.



**Figure 3.2:** The upper body and hand articulations.

We assume that all vectors are already normalized. We also assume that the articulated joints are defined in the world (global) coordinate system rather than based on a bone relationship within the skeletal hierarchy, so they are not configured for relative movement within the skeleton. In other words, our hand model does not use the rotation matrices given by Euler angles to identify postures or motions; thus, the potential gimbal lock [47] issue can be avoided. This assumption makes the hand model generalized for any MoCap device that captures position information in space. We used a system called *Perception Neuron* for implementing our gesture recognition system. This system consists of IMU based sensors. These sensors are present in

places of major joints. They export the joint positions. All the motion capture data is passed into a hub which in turn transmits the data to a package (that serves as a plug-in for the Unity Game Engine) from Perception Neuron. The data is then transferred to the Unity Game Engine where this data drives a human skeleton to imitate the movements performed by the user. Our gesture recognition approach was built based on this skeleton. Section 4.4 gives a more detailed description on the use of the Perception Neuron MoCap system.

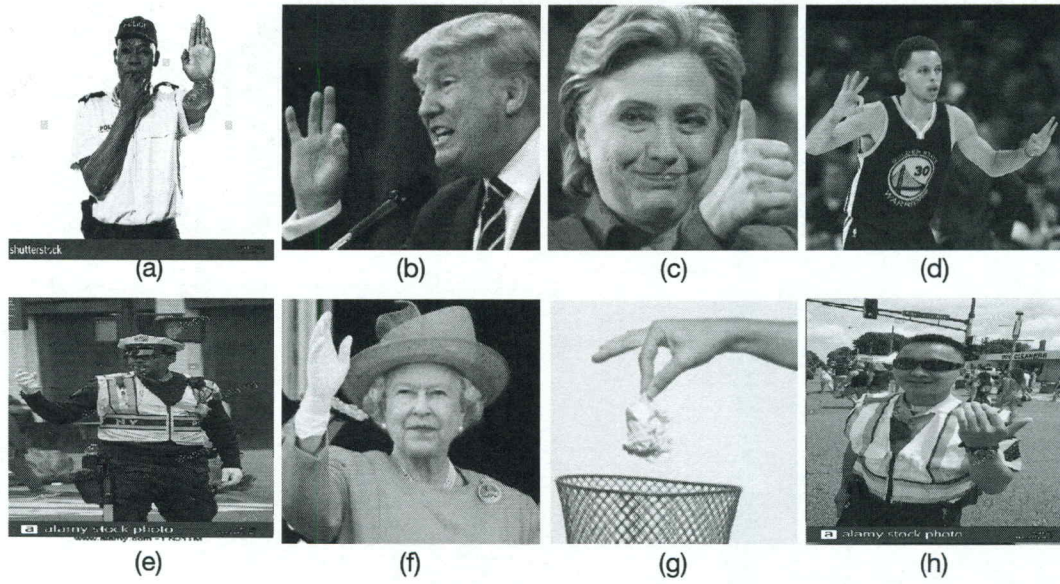
We define a local coordinate system rooted at the chest joint of the avatar (as shown in Figure 3.2 (b)). The acceptance of initial hand postures are evaluated relative to the direction of the axes of the local coordinate system as explained in Section 3.2. Our system's y-axis vector  $\vec{y}$  is parallel to the world y-axis vector. Our z-axis vector  $\vec{z}$  direction corresponds to the direction the avatar faces. To obtain the vector  $\vec{z}$ , we first compute an assisting vector  $\overrightarrow{s_r s_l}$  using the shoulder joints. We denote the left shoulder joint as  $s_l$  and the right shoulder joint as  $s_r$ . Equation 3.1 shows the metric for computing the axes.

$$\begin{aligned}
\vec{y} &= \langle 0, 1, 0 \rangle; \\
\overrightarrow{s_r s_l} &= \frac{s_l - s_r}{\|s_l - s_r\|}; \\
\vec{z} &= \frac{\overrightarrow{s_r s_l} \times \vec{y}}{\|\overrightarrow{s_r s_l} \times \vec{y}\|}; \\
\vec{x} &= \vec{y} \times \vec{z};
\end{aligned} \tag{3.1}$$

With our local coordinate system, users are allowed to adjust their body while performing hand gestures ( i.e., they may bend forward, turn, walk, or lay back) which are habitual actions we perform daily to relax our bodies. We believe, having no constraint on the position and/or posture of the users' body while interacting with applications has the potential to increase the user engagement, especially during a prolonged interaction. For example, when playing a video game, no matter how often one has been playing it, there are moments of readjustments for one's physical comfort. According to the study by Wachslar et al. [48], people usually focus on the comfort of their backs and buttocks while they sit or lean the body against a supporting object. The neck and shoulders play a secondary role in regards to the influence on comfort, and the comfort of the thighs and legs has almost no impact on the overall comfort. In our described articulation approach, the only restriction is that the local coordinate system and our recognition techniques require the hands in front of the body at all times.

### **3.2 Gesture Vocabulary**

Natural hand gestures can be classified into two categories: static hand gestures and dynamic hand gestures [49,50]. Figure 3.3 shows a few examples of natural static and dynamic hand gestures. Static hand gestures are defined by the posture of the hand where the hand and fingers do not move within a certain period of time. Dynamic hand gestures are meaningful actions that involve physical movements of the hand and fingers in two or three dimensions. The position of the hand and/or fingers changes temporally. Technically, a dynamic gesture is recognized as an action



**Figure 3.3:** The first row images (a-d) show static gestures. In (b) and (d), their hands and fingers have very similar postures, but the meanings are different. The second row images (e-h) show dynamic gestures. The meanings are presented by the motions of hands and fingers. Hand gesture examples are excerpted from Internet. The sources are: (a) [1], (b) [2], (c) [3], (d) [4], (e) [5], (f) [6], (g) [7], (h) [8]

that is performed within a period of time. It usually has three stages: *start*, *update*, and *end* [51]. The start stage detects a hand posture that satisfies the criteria to start the tracker. The update stage detects in-between changes within the sequence of satisfactory postures. The end stage detects an exclusive posture that makes the hand movement not satisfy the criteria any more.

According to the design heuristics of usability engineering [19, 52], we define our gesture vocabulary with the goal of eliminating the burden on users to learn, remember, or recall the gestures. Also, we make the gestures closely related to users' natural hand motions. There are four considerations for designing the gestures: (1)

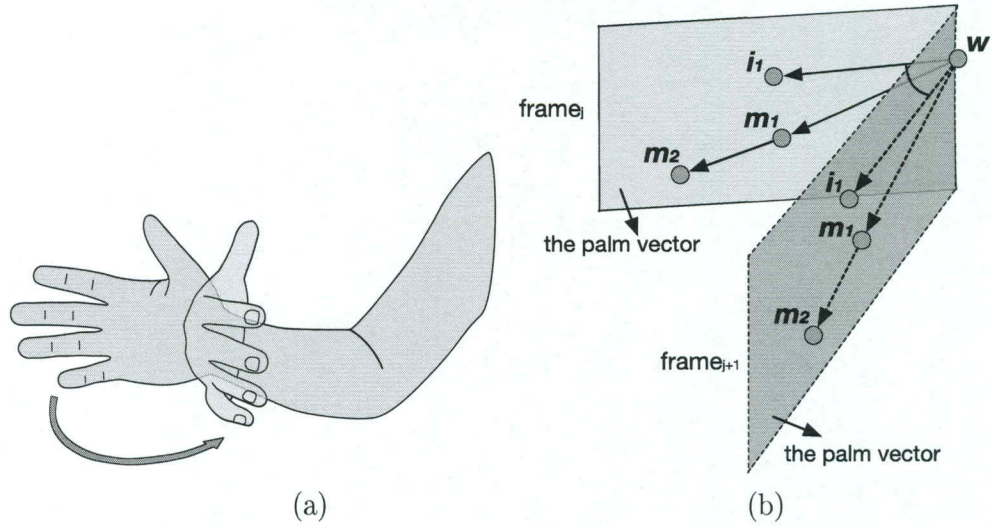


they are conscious gestures [53] that have meaning without speech, as opposed to spontaneous gestures having meaning only within the context of speech; (2) they are supported gestures which are not necessarily performed by having hands in the air but can be placed on an armrest to avoid excessive physical exertion and thus be performed over a long period of time; (3) they deliver signs, where a dynamic gesture can be categorized clearly into a start posture, possibly a series of update postures and a stop posture. This is opposed to those gestures without any clear categorization of start or stop postures when performed in a conversational setting [54]; and (4) they are authoritative gestures [55] that give commands or orders in gesture form (e.g., Figure 3.3 (a, e, g, h)), as opposed to descriptive gestures that are used to reinforce ideas or encode meaning to facilitate social transactions (e.g., Figure 3.3 (b, c, d, f))

The gesture vocabulary contains 6 basic hand gestures. The following subsections describes the applicable situations and technical approaches of implementation for each of those gestures.

### **3.2.1 Swipe**

Swipe is considered to be a dynamic gesture. As shown in Figure 3.4 (a), a natural swipe gesture is characterized by the palm of the hand facing sideways and rotating the wrist so that the hand moves in the direction of the palm. Swipe is a common gesture used in various actions in daily life such as while switching between pages while using personal gadgets such as tablets or Kindles. Oftentimes, people use a swipe gesture to ask someone in the distance to move aside. We develop a swipe



**Figure 3.4:** The swipe gesture.

gesture action that people see or use in their daily lives, so that it eliminates the need for users to learn or remember how to perform the gesture. This swipe gesture can enable interactions with virtual environments, including maneuvering 3D objects to different paths or directions in a game, flipping pages of a digital document, or browsing through images in a digital image gallery.

Since swipe is a dynamic gesture, it has three stages mentioned earlier, namely the start stage, the update stage and the end stage. The start stage of the swipe gesture consists of the recognition of the palm (or palms) being held fairly straight, facing inward, and fingers pointing away from oneself. We use  $\overrightarrow{wm_1}$  and  $\overrightarrow{m_1m_2}$  to check if the hand is straight. Figure 3.4 (b) illustrates the vector representation of the gesture. It uses the *Swipe Palm Angle*, defined as the angle between the two vectors. The hand would be in a perfectly straight posture if the Swipe Palm Angle is equal to 180 degrees. To check the initial orientation of the hand for correctness

**Table 3.1:** The thresholds of the parameters used in the swipe gesture.

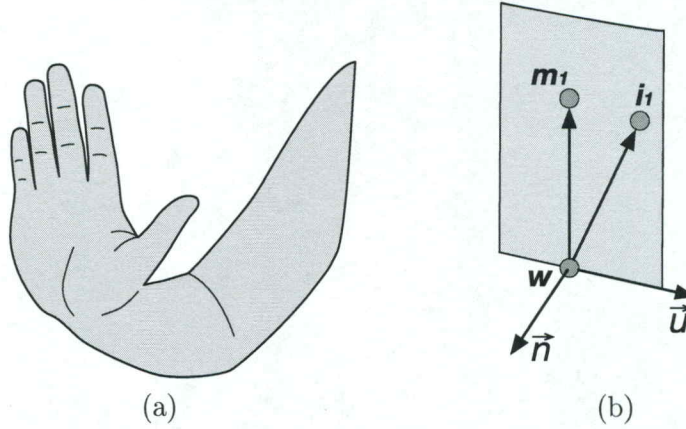
Parameters	Threshold Ranges
<i>Swipe Palm Angle</i> ( $\text{acos}(\overline{wm}_1 \cdot \overline{m}_1 \overline{m}_2)$ )	[120°, 180°]
<i>Swipe Posture Angle</i> ( $\text{acos}(\vec{n} \cdot \vec{y})$ )	[65°, 115°]
<i>Swipe Angular Velocity</i> ( $\text{acos}(\overline{wm}_1^{f_j} \cdot \overline{wm}_1^{f_{j+1}})/\Delta t$ )	[900°/second, +∞)
<i>Swipe Direction</i> ( $(\overline{wm}_1^{f_j} \times \overline{wm}_1^{f_{j+1}} \cdot \vec{y})$ )	$[-\infty, 0)$ - left hand $(0, +\infty]$ - right hand
<i>Swipe Performing Time</i>	[0.016 seconds, +∞)
<i>Swipe Break Time</i>	[0.1 seconds, +∞)

(straight, palms facing inward and fingers pointing away from the body), we compute the palm vector, denoted as  $\vec{n}$ , equal to the normalized cross product of  $\overline{wi}_1$  and  $\overline{wm}_1$ . For the left hand,  $\vec{n} = \frac{\overline{wi}_1 \times \overline{wm}_1}{\|\overline{wi}_1 \times \overline{wm}_1\|}$ , and for the right hand,  $\vec{n} = \frac{\overline{wi}_1 \times \overline{wm}_1}{\|\overline{wi}_1 \times \overline{wm}_1\|}$ . The *Swipe Posture Angle* is defined as the angle between  $\vec{n}$  and the  $\vec{y}$  axis. The palm would be at a perfect sideway posture if the *Swipe Posture Angle* is equal to 90 degrees. Both the *Swipe Palm* and the *Swipe Posture* angles are obtained by taking the dot products of their respective constituent vectors.

The update stage follows the satisfaction of the conditions defined in the start stage. This stage uses the *Swipe Angular Velocity* of the rotation at the wrist joint. We define threshold ranges for all the parameters used in a particular gesture, which define the range of values the respective parameters could assume at the desired time for performing a particular gesture successfully. This stage also requires the values of the *Swipe Palm Angle* and *Swipe Posture Angle* to be in their threshold ranges. The *Angular Velocity* is defined as the angle between the vectors  $\overline{wm}_1^{f_j}$  and  $\overline{wm}_1^{f_{j+1}}$ , where  $\overline{wm}_1^{f_j}$  is the  $\overline{wm}_1$  at the frame  $j$ , and  $\overline{wm}_1^{f_{j+1}}$  is the  $\overline{wm}_1$  at

the next consecutive frame. In our implementation, the left hand can only perform the swipe to the right, and the right hand can only perform the swipe to the left. To make sure the direction of the movement of the hand is correct, we check the *Swipe Direction*, which is determined by the dot product between the  $\vec{y}$  axis and the vector resulting from the cross product of  $\overline{wm}_1^{f_j}$  and  $\overline{wm}_1^{f_{j+1}}$ . We also define *Swipe Performing Time*, which is the duration for which the conditions for performing the swipe gesture are met. Table 3.1 lists the parameters used in the swipe gesture and their threshold ranges. The Swipe Performing Time value begins accumulating after the start stage, and stops after reaching the end stage. The end stage is triggered when any of the first four parameters defined in Table 3.1 pass the threshold range. At the end stage, if the Swipe Performing Time is within its threshold range, a swipe gesture is detected.

As observed in real human hand swipe motions, the speed of the hand may not always be uniform or increasing. Sometimes for a very small amount of time (a fraction of a second), it is possible that the Swipe Angular Velocity may briefly be below the threshold, before quickly exceeding it. Humans will not perceive this as two separate swipes, but the method presented above would do so. To avoid this anomaly, we define a parameter called *Swipe Break Time*, which makes sure that a minimum amount of time has passed before recognizing another swipe gesture. The value of Swipe Break Time begins accumulating from zero if the end stage has detected a swipe gesture; otherwise, it is set to be a number in the range. The process to recognize another swipe gesture will start after the Swipe Break Time has reached its threshold range.



**Figure 3.5:** The stop gesture.

### 3.2.2 Stop

Stop is a static gesture and is performed as shown in Figure 3.5 (a). It is characterized by an open palm facing outwards from the front of the body with the four fingers extended straight up and together. A stop gesture is usually used to end an activity. For example, a security guard may use a stop gesture to stop a crowd from coming towards him from a distance. It may also be used as an attempt to stop someone from doing certain things. In the field of HCI, a stop gesture may be used to stop the execution that the computing system is performing.

In this gesture, we assume that the direction of  $\overrightarrow{wm_1}$  represents the hand pointing direction. There are two vectors used in recognizing this stop gesture, as illustrated in Figure 3.5 (b): one is the palm vector  $\vec{n}$  (as shown in Section 3.2.1 for its definition); the other is the vector pointing to the side of the palm denoted as  $\vec{u} = \overrightarrow{wm_1} \times \vec{n}$ . The *Stop Palm Angle* of the stop gesture is defined as the angle between the vector  $\vec{n}$  and the z-axis, and the *Stop Posture Angle* is defined as the

**Table 3.2:** Parameters and threshold ranges used in the stop gesture.

Parameters	Thresholds or Threshold Ranges
<i>Stop Palm Angle</i> ( $\text{acos}(\vec{n} \cdot \vec{z})$ )	$[0^\circ, 25^\circ]$
<i>Stop Posture Angle</i> ( $\text{acos}(\vec{u} \cdot \vec{x})$ )	$[0^\circ, 25^\circ]$
<i>Stop Performing Time</i>	0.016 seconds

angle between the vector  $\vec{u}$  and the x-axis. The two angles are obtained from the dot products of the vectors  $\vec{n}$  with  $\vec{z}$  and  $\vec{u}$  with  $\vec{x}$ , respectively. When the first stop posture is detected, the *Stop Performing Time* parameter starts recording the elapsed time. If the value of Stop Performing Time reaches the threshold, a stop gesture is detected, and its value is reset to zero. The elapsed time will then be accumulated as long as the hand is at the stop posture. Table 3.2 lists the parameters used in the stop gesture and their threshold ranges.

### 3.2.3 Come

Come is a dynamic gesture. As shown in Figure 3.6 (a), it starts with a posture where the palm faces upward and the fingers are together, straight, and pointing away from the body. Following this, the four finger tips (primarily the middle finger tip) move towards the hollow of the palm. A come gesture has been used in human-to-human communication in instances such as gesturing a person standing at a distance to move closer. This gesture has also been used in computer interfaces to move objects, zoom in to images, and in racing video games to drive a car in reverse.

The start stage requires the palm to be facing up. To verify this, we define the *Come Palm Angle*, which is nothing but the angle between the vector  $\vec{n}$  and

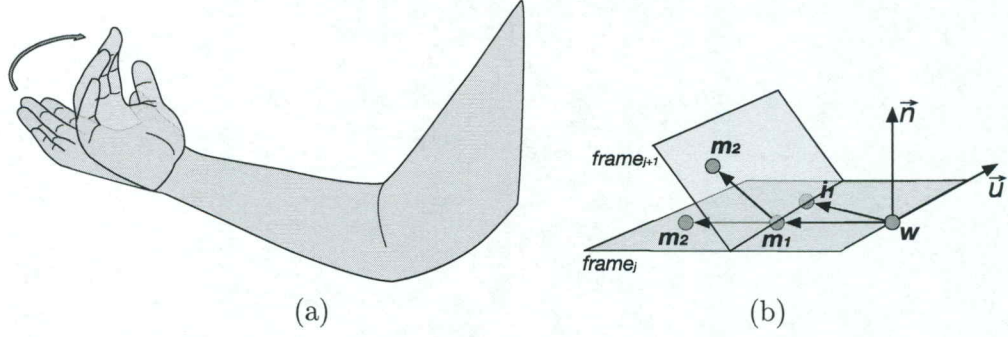


Figure 3.6: The come gesture.

Table 3.3: Parameters and threshold ranges used in the come gesture.

Parameters	Threshold Ranges
<i>Come Palm Angle</i> ( $\text{acos}(\vec{n} \cdot \vec{y})$ )	$[0^\circ, 30^\circ]$
<i>Come Posture Angle</i> ( $\text{acos}(\vec{u} \cdot -\vec{x})$ )	$[0^\circ, 30^\circ]$
<i>Come Angle Velocity</i> $((\text{acos}(\overline{wm_1}^{f_j} \cdot \overline{m_1m_2}^{f_j}) - \text{acos}(\overline{wm_1}^{f_{j+1}} \cdot \overline{m_1m_2}^{f_{j+1}})) / \Delta t)$	$[100^\circ/\text{second}, +\infty)$
<i>Come Performing Time</i>	$[0.016 \text{ seconds}, +\infty)$
<i>Come Break Time</i>	$[0.1 \text{ seconds}, +\infty)$

the  $\vec{y}$  axis. Ideally this angle should be 90 degrees, but people may not be able to comfortably hold the palm at this ideal angle due to their natural instincts. Thus, a threshold range is set for the Palm Angle. The *Come Posture Angle* is defined as the angle between the vector  $\vec{u}$  and the  $-\vec{x}$  axis. Ideally this angle should be 0 degrees. But, there is a threshold range set for the Come Posture Angle as well. Table 3.3 lists the parameters used in the come gesture and their threshold ranges.

The update stage follows the satisfaction of the conditions defined in the start stage. In the update stage, we check the *Come Angle Velocity* of the rotation at the first and second joint of the middle finger. We note that the requirements of the start

stage must be satisfied during the update stage. The Come Angle Velocity is defined as the difference between the angle formed by  $\overrightarrow{wm_1^{f_j}}$  and  $\overrightarrow{m_1m_2^{f_j}}$  and the angle formed by  $\overrightarrow{wm_1^{f_{j+1}}}$  and  $\overrightarrow{m_1m_2^{f_{j+1}}}$  (as shown in Table 3.3). Figure 3.6 (b) illustrates the vectors used for detecting a come gesture. We note that the Come Angle Velocity is a positive value when the middle finger moves towards the palm, and it is a negative value when moving away from the palm. Once the velocity value is in the threshold range, the *Come Performing Time* increases until the process reaches the end stage. The process reaches the end stage when one or more of the first three parameters in Table 3.3 are out of the threshold range. In the end stage, a come gesture will be detected if the Come Performing Time reaches a value within its threshold range. The value of Come Performing Time is set back to zero before processing the next gesture.

An anomaly of unexpected multi-recognition, like the one in the swipe gesture, may occur if the value of Come Angle Velocity falls below the threshold and then exceeds it during a very short period of time. To avoid this anomaly, the *Come Break Time* is used to make sure that a minimum amount of time has passed before recognizing another come gesture. The value of the Come Break Time starts increasing after the end stage detects a come gesture. When its value has surpassed the threshold, as specified in Table 3.3, the process to detect another come gesture can start.



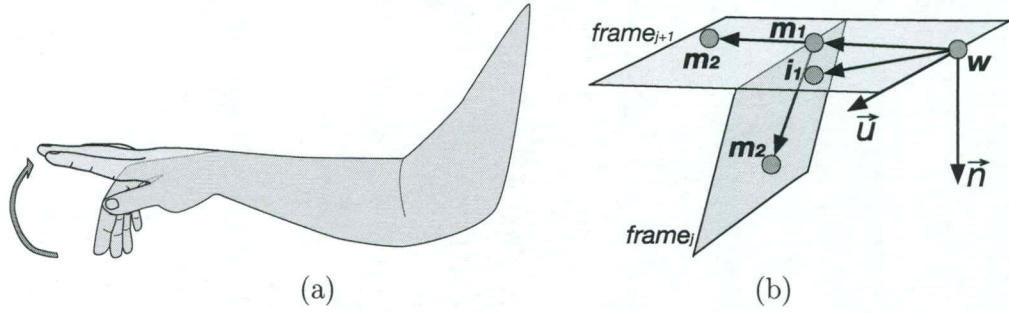


Figure 3.7: The go gesture.

### 3.2.4 Go

Go is a dynamic gesture. As shown in Figure 3.7 (a), it is characterized by the palm of the hand facing downwards and having the fingers bent towards the palm as the starting posture of the gesture. The four fingers then move away from the outstretched palm so as to make the palm fairly straight. The go gesture is the inverse command to the come gesture. It can be used to ask somebody to go away. In a virtual reality application, it can be used to push an object away.

The start stage requires the palm to be facing downwards. This is determined by the *Go Palm Angle*, the angle between the vector  $\vec{n}$  and  $-\vec{y}$  axis. It also requires the use of the *Go Posture Angle* which is the angle between  $\vec{u}$  and  $\vec{x}$  axis. Their threshold ranges are shown in Table 3.4.

In the update stage, we check the *Go Angle Velocity* which is defined as the difference between the angle formed by  $\overline{wm_1}^{f_{j+1}}$  and  $\overline{m_1m_2}^{f_{j+1}}$  and the angle formed by  $\overline{wm_1}^{f_j}$  and  $\overline{m_1m_2}^{f_j}$  (as shown in Table 3.4). Figure 3.6 (b) illustrates the vectors used for detecting a go gesture. We note that the Go Angle Velocity is a positive

**Table 3.4:** Parameters and threshold ranges used in the go gesture.

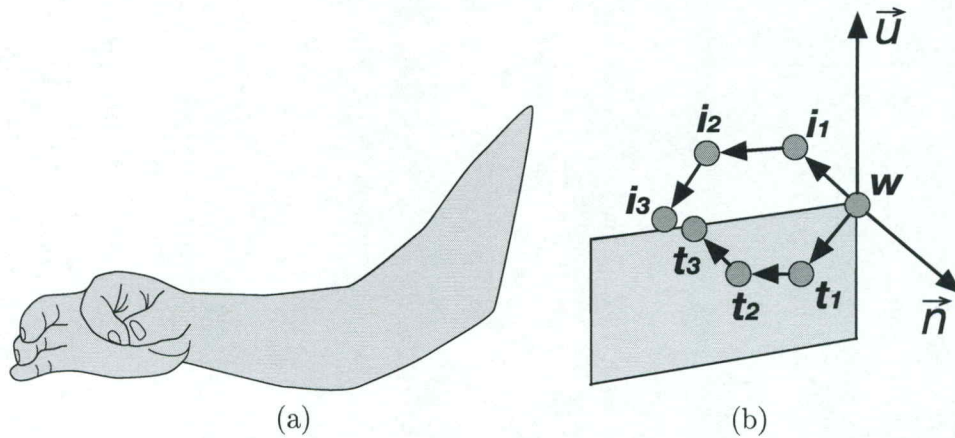
Parameters	Threshold Ranges
<i>Go Palm Angle</i> ( $\text{acos}(\vec{n} \cdot -\vec{y})$ )	$[0^\circ, 35^\circ]$
<i>Go Posture Angle</i> ( $\text{acos}(\vec{u} \cdot -\vec{x})$ )	$[0^\circ, 30^\circ]$
<i>Go Angle Velocity</i> $((\text{acos}(\overline{w}\vec{m}_1^{f_{j+1}} \cdot \overline{m}_1\vec{m}_2^{f_{j+1}}) - \text{acos}(\overline{w}\vec{m}_1^{f_j} \cdot \overline{m}_1\vec{m}_2^{f_j}))/\Delta t)$	$[200^\circ/\text{second}, +\infty)$
<i>Go Performing Time</i>	$[0.016 \text{ seconds}, +\infty)$
<i>Go Break Time</i>	$[0.1 \text{ seconds}, +\infty)$

value when the middle finger moves away from the palm, and it is a negative value when moving towards the palm. The value of *Go Performing Time* increases while the velocity value is within the threshold range. The process moves into the end stage when one or more of the first three parameters in Table 3.4 are out of the threshold ranges. In the end stage, a go gesture will be detected if the Go Performing Time reaches a value in its threshold range. The value of Go Performing Time is set back to zero before processing the next gesture.

Similar to the other dynamic gesture algorithms, we use the *Go Break Time* parameter to avoid the multi-recognition anomaly. The Go Break Time ensures that a minimum amount of time has passed before the next Go gesture can be recognized.

### 3.2.5 Pinch

Pinch is a static gesture. As shown in Figure 3.8, a pinch gesture requires the palm of the hand facing sideways to the other hand, and involves the tip of the thumb touching the tip of the index finger. Pinch is a gesture performed using the thumb and the index finger by placing them together to hold an object, including, picking



**Figure 3.8:** The pinch posture.

up a pin or holding salt grains between the thumb and the index finger. It has been used in virtual reality environments for users to select an object. Though pinch is a simple gesture, it can be used in conjunction with other gestures. For example, people can pick an object, and move it with arm or hand movements and throw it away with a wrist rotation.

In the process of pinch gesture recognition, both the hands are treated independently. It uses the *Pinch Palm Angle* which is the angle between the vector  $\vec{u}$  and the  $\vec{y}$  axis. The wrist therefore has the freedom to rotate about the  $\vec{y}$  axis. The palm would be at a perfect posture if the value of Pinch Palm Angle is zero. It also uses the *Pinch Posture Angle* which is the angle between  $\vec{n}$  and the  $\vec{z}$  axis. The ideal angle value is 90 degrees. We find the distance between the tip of the thumb and the tip of the index finger and record that distance as the *Pinch Distance*. Table 3.5 lists the parameters and their threshold ranges. A pinch gesture is detected when the Pinch Distance is a value within its threshold range.

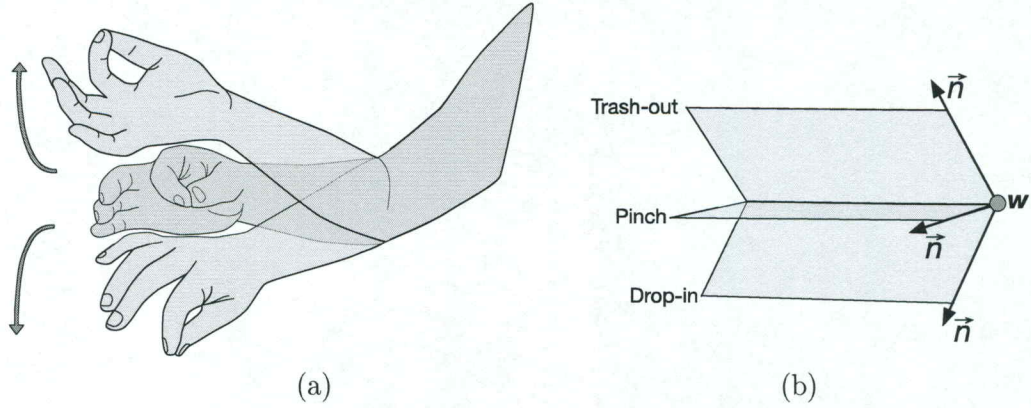
**Table 3.5:** Parameters and threshold ranges used in the pinch gesture.

Parameters	Threshold Ranges
<i>Pinch Palm Angle</i> ( $\text{acos}(\vec{u} \cdot \vec{y})$ )	$[0^\circ, 35^\circ]$
<i>Pinch Posture Angle</i> ( $\text{acos}(\vec{n} \cdot \vec{z})$ )	$[60^\circ, 120^\circ]$
<i>Pinch Distance</i> ( $\ i_3 - t_3\ $ )	$[0, 1\text{cm}]$

### 3.2.6 Drop in and trash out

The Figure 3.9 (a) illustrates the two gestures of drop-in and trash-out. Drop-in and trash-out are a pair of gestures used to give commands like “put it in” and “throw it away.” They also can be used to indicate the direction in which a person intends to move an object. Both gestures are usually combined with a pinch gesture to indicate the need of holding an object first. The drop-in gesture performs a pinch gesture first, then turns the hand (1) to make the palm move from facing sideways to facing down and (2) to stop the pinch. In a data sorting application, the drop-in gesture can be used to select data elements such as digital images and then put them in a folder. The trash-out gesture is defined as performing a pinch gesture first, and then changing the orientation of the palm from facing sideway to facing up to stop the pinch. The trash-out gesture can be used to pick up an object and then toss it away to demonstrate getting rid of it. In a gesture-based application, it can be used to delete elements such as files, folders or emails. In a video game, it provides commands to destroy game objects or eliminate obstacles from the path.

Both drop-in and trash-out gestures are dynamic gestures. The start stage involves the detection of a pinch gesture, and the end stage involves checking the



**Figure 3.9:** The drop-in and trash-out gestures with the hand in a pinch posture.

**Table 3.6:** Parameters and threshold ranges used in the drop-in and trash-out gestures.

Parameters	Threshold Ranges	Gesture Types
<i>Throw Posture Angle</i> ( $\text{acos}(\vec{n} \cdot \vec{y})$ )	$[0^\circ, 35^\circ)$	Trash-out
	$[55^\circ, 125^\circ]$	Pinch
	$(145^\circ, 180^\circ]$	Drop-in

action of the gesture. We use the approach introduced in Section 3.2.5 to detect if a pinch gesture is performed. While the update stage requires the pinch gesture to be active, drop-in and trash-out gestures are performed in different ways. We define the *Throw Posture Angle* as the angle between the vector  $\vec{n}$  and  $\vec{y}$  axis. If the angle is a value corresponding to the posture of the palm facing down, the user is intending to perform a drop-in gesture. Otherwise, the intention is to perform a trash-out gesture. Table 3.6 shows the threshold ranges of the Throw Posture Angle. The pinch gesture deactivates when a drop-in or trash-out gesture is detected.

### 3.3 How Our Approach Is Better?

There are different approaches used along with different devices to develop a system that recognizes gestures. The state of the art gesture recognition systems or research in the development of gesture detection algorithms mostly make use of machine learning algorithms. Such approaches that use machine learning algorithms require a large amount of training data. Such data collection is a time-consuming task. It also requires manual annotation of data as ground truth in order to train accurate gesture recognition models [56] [32], which is a cumbersome and laborious task. In addition, a substantial amount of these studies consider only static hand gestures and not the dynamic ones [57] [58], which raises questions on how accurately the systems would recognize dynamic gestures.

In the recent past, many affordable motion capture systems that have hit the commercial market have improved motion capture accuracy. They have opened a new dimension for research in developing an efficient gesture recognition system without the use of machine learning algorithms, which can recognize gestures accurately.

Our approach eliminates the use of machine learning algorithms and collecting and annotating datasets. We take advantage of the state-of-art motion capture systems and their software development kits to develop a novel approach that can be used for recognizing natural hand gestures. The basic set of natural gestures include gestures for navigation, selecting, de-selecting, translating, scaling, etc. [22]. For this reason, we have defined natural gestures that closely match the tasks as mentioned here: Swiping for navigation and translation, come and go gestures for scaling (or

zooming), a pick gesture for selecting, a drop gesture for de-selecting, and a drop-in and trash-out gesture for data manipulation tasks such as saving a file (i.e., drop-in gesture) and deleting a file (i.e., trash-out gesture).

## CHAPTER 4

### USER EXPERIMENT TO STUDY THE GORILLA ARM EFFECT

#### 4.1 Overview

The hand gestures described in Chapter 3 can be seamlessly integrated into user interfaces as an input modality. Past research has studied hand gestures as a type of input for interfaces. For example, Bolt performed a study called “Put-That-There” [59] that used hand gestures within a graphical user interface. Another example is the gesture interface in the movie “Minority Report” which was a creative work based on the research by Underkoffler [60]. In the movie, the lead actor Tom Cruise used a three-fingered glove to perform various gestures to interact with data. These gesture modalities however are not yet considered as the primary mode of input for human computer interaction. One major reason is the *gorilla arm syndrome*. Howe [61] further described the arm discomfort at a mid-air interaction (gorilla arm) as “... *humans aren’t designed to hold their arms in front of their faces making small motions. After more than a very few selections, the arm begins to feel sore, cramped, and oversized - the operator looks like a gorilla ...*”.

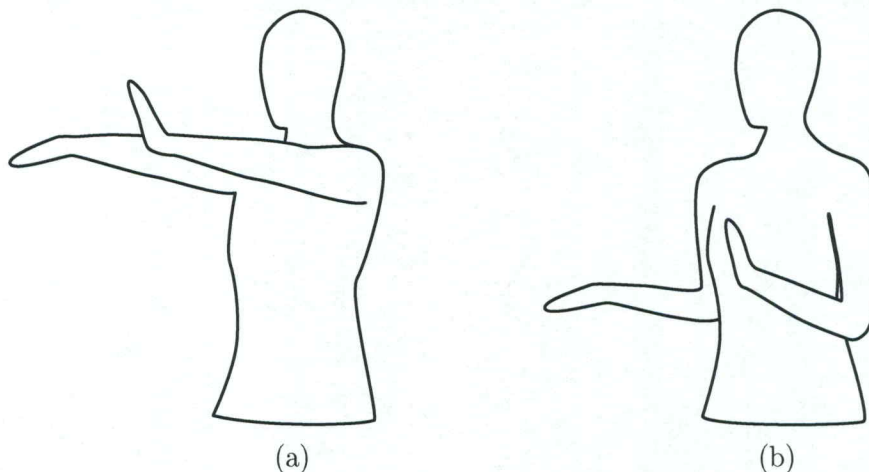
Often gorilla arm effects are found in sign language interpreters who perform various hand gestures over an extended period of time, especially when translating



the words of a speaker to a deaf crowd in real time. A study by Rempel et al. [23] that observed 24 experienced sign language interpreters showed that they experienced fatigue and discomfort while performing hand gestures for long speeches, with the pain after the session lasting half a day. The results of this study motivate our work, though the subjects are different. We focus on the gorilla arm effects caused by prolonged interaction with computer interfaces.

In this chapter, we present a game-based experiment to compare the exertion levels from the use of keyboard, mid-air gestures, and the hand gestures as described in Chapter 3. Since our hand gestures can be performed in a resting position on an armrest of a chair or one’s own lap, we denote these as *supported gestures*. Supported gestures are those proposed in our gesture vocabulary, as shown in Section 3.2, where the arm is at a supported position commonly used in face-to-face communications. Mid-air gestures require the arm to stay at an extended position, which is the common position for vertical touchscreen use or sign language communication. For each type of input, namely keyboard inputs, mid-air gestures, and supported gestures, a participant plays the game for an equal amount of time. Then, the exertion levels experienced in each input type are rated by the participant. A higher exertion level indicates a more significant gorilla arm effect (gorilla arm syndrome). Figure 4.1 shows the example postures for both mid-air and supported gestures.

Sections in this chapter are organized as follows. Section 4.2 describes the design of mid-air gestures in correspondence to those supported gestures for this study. Section 4.3 presents the experimental design. Section 4.4 describes the devices used



**Figure 4.1:** Example postures for a mid-air gesture (a) and a supported gesture (b). The posture in (a) is a common arm position for vertical touchscreen. The posture (b) is the arm position for gestures during face-to-face communications.

in the experiment. Section 4.5 explains the procedure of the user study. Section 4.6 presents the discussion of results.

## 4.2 Mid-air gestures

Different from the supported gestures defined in Section 3.2, the mid-air gestures are the conventional gestures performed with arms stretched out straight in front of the body as shown in Figure 4.1 (a), which is the position of hands that Rempel et al. [23] and Ramos et al. [26] claimed the most fatigue inducing position. We designed a mid-air gesture vocabulary that contains the same gesture types as the supported gesture vocabulary defined in Section 3.2. The same angular-velocity method used to implement the supported gestures as described in Chapter 3 was used to implement these mid-air gestures but with constraints that forces users to have

their arms stretched straight. For this purpose, two vectors calculated from the arm joints are added in addition to the 16 joints in the hand model. We denote the two vectors as  $\vec{se}$  and  $\vec{ew}$ , where  $s$  is the shoulder joint,  $e$  is the elbow joint, and  $w$  is the wrist joint. We check the angle between  $\vec{se}$  and  $\vec{ew}$ . Ideally, this angle should be at  $180^\circ$ . However, the natural posture of human arms may vary from person to person where keeping the arms perfectly straight is not feasible and unnatural since the anatomic structure of the arms requires a small degree of flexion at the elbows, even with the intention to keep the arms straight. In practice, we define a lower bound on the angle between  $\vec{se}$  and  $\vec{ew}$  as the *arm posture threshold*, which is set to be  $150^\circ$ . For all mid-air gestures, the hand postures are the same as those in the corresponding supported gestures. Before calling a gesture recognition function, the arm posture is checked first to determine whether or not the arm is straight. Besides this additional requirement, there are two modifications made in the mid-air gesture recognition functions, as itemized below:

- The swipe gesture is performed by rotating the forearm with the pivot at the elbow joint, and the forearm moves towards the direction that the palm faces. Here the vector  $\vec{ew}$  is used to calculate the angular velocity since the swipe gesture requires users to rotate the forearm at the elbow instead of rotating the hand at the wrist, as described in Section 3.2.1.
- To perform the come gesture, users position the palm to face upwards and then perform a brisk flexion of the forearm towards the body. Similar, to performing the go gesture, users position the palm to face downwards and perform a brisk

flexion of the forearm away from the body. Here the only implementation difference from the corresponding supported gestures is that we use the angular velocity between  $\vec{s}\vec{e}$  and  $\vec{e}\vec{w}$  to decide whether or not the user is performing the gesture, rather than using the angular velocity between  $\vec{w}_1\vec{m}_1$  and  $\vec{m}_1\vec{m}_2$ , as described in Section 3.2.3 and Section 3.2.4. The threshold value for this angular velocity is set to 100 degrees per second.

While all the gestures described in Chapter 3 have been implemented as mid-air gestures as well, only mid-air swipe and mid-air stop gestures were used in this experiment. We use these gestures to compare the exertion levels of performing gestures with hands in mid-air versus performing gestures with hands in a resting position.

One reason we tested our swipe gesture is because the touchscreen swipe gesture is widely used across almost all applications for different purposes, including switching between apps in mobile smart phones' basic interfaces, in games like *Temple Run* to maneuver player characters, to switch between screens, and turn pages of an e-book, etc. The frequency of using a swipe gesture is much higher than the frequency or possibility of using a pinch gesture. This is only a general case and the frequency and possibility of usage depends on personal needs as well.

One reason we tested the stop gesture is because our game-based environment needed this gesture.

In most user interfaces that support hand gestures as input, there is a limited set of gestures that can be used. Also, the use of these gestures is limited to a

particular scenario. For example, in a general touchscreen user interface of a mobile phone, a touchscreen pinch gesture is used to zoom-in to objects. The objects that can be zoomed into are limited to a few such as pictures, pages of an e-book, and web pages in certain browsers.

There are also gestures that are rarely used, and where the scenarios in which that gesture is used occurs less frequently. A gesture that is more difficult to perform is better required to be performed in scenarios that are rare where the necessity to perform those kind of gestures is less [23]. Hence to accommodate the need of more gestures in an interface but still avoid the exertion caused in the arms (gorilla arm effect), these kind of arm gestures can be implemented easily with our approach as described in Chapter 3, just by tweaking actual gestures or by creating new ones and can be used in scenarios or for tasks that are rarely encountered.

Having described the various gestures that were used in the experiment, we now delve into the intricate details of the experiment itself. Sections 4.3 - 4.5 describe these details followed by the results, analysis and the discussion of these results.

### **4.3 Experiment**

Our experiment estimated the physical exertion when a user utilizes different modes of input to interact with an interface. The participants played a video game using all the three modes of input (keyboard, mid-air gestures, and supported gestures) for 30 minutes each. The study was conducted with 16 participants, out of which 10 were male and 6 were female. The participants were sampled from university classrooms and participated on a first come first served basis. The sampling selection

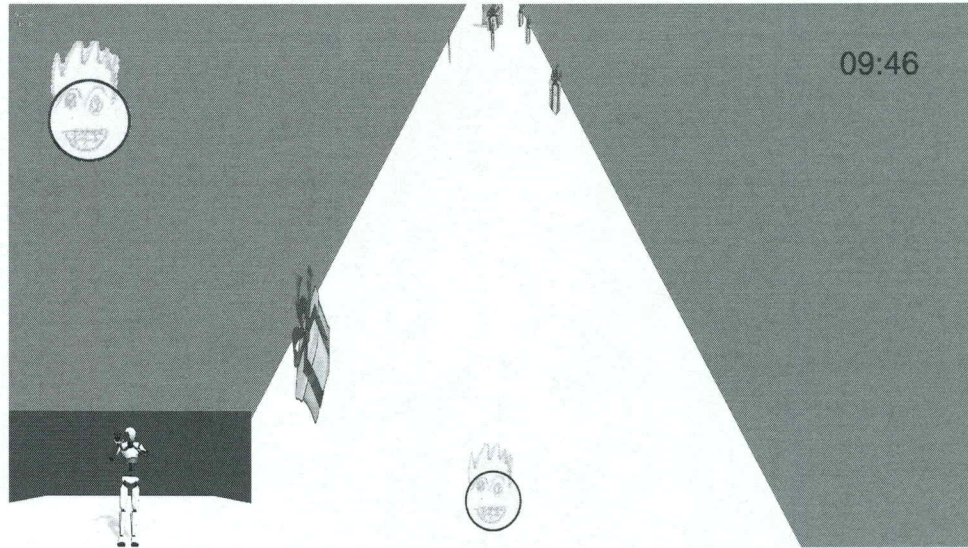
was random. The age of the participants ranged from 20 to 28, with a mean age of 23. The video game was developed to test the viability of the gestures we developed.

#### 4.4 Apparatus

**Motion Capture system:** The gesture recognition systems require capturing the position of users' hands and arms in real-time. The IMU based motion capture suit from Perception Neuron was used. It was easy to use, put on and take off. As the gestures are only concerned with hand and arm, only two gloves and the torso strap were used. The head strap and other lower body straps were not needed which made it easier to handle the suit. There were a total of 23 sensors used. Nine sensors were present on each glove. The configuration for those nine sensors was as follows: two sensors each on the thumb, index finger, and middle finger, one each on the ring finger and pinky finger, and one on the wrist. This was because the thumb, index finger, and middle finger are three fingers that have a good degree of independence and are mostly used while performing gestures providing better accuracy than the other two fingers. There was also a sensor on each arm that tracks the position of the arm. Three sensors were present on the torso strap where one sensor tracked each shoulder. The third sensor was for the hip which was the parent sensor that ultimately got the data from the rest of the sensors and passed the data on to a hub to which it was connected to. The hub passed the data onto the computer through a wifi connection that drove the avatar on the screen to replicate the movements performed by the user.

**Perceived exertion:** Perceived exertion is the subjective measurement of physical exertion. The participants were asked to rate their current level of exertion using a Borg CR10 scale. This is a 12-point scale that starts with a 0, goes on to 0.5, then scales from 1-10 with descriptions along each point to rate the amount of exertion starting from “Nothing at all” to “Impossible.” The participants would rate their exertion after playing through each input condition and their responses were recorded on a feedback form they completed at the end of each session.

**Game:** The winter mini-game was used from the “Happy ball” game for this experiment. This mini-game consists of a bouncing ball traveling through a certain path with three lanes the ball can switch between, in the winter season setting. The aim of the ball is to survive as long as possible. The ball can collect Christmas gifts which are the reward objects in this case. By doing this, the ball accumulates points in one point increments. The two obstacles that can hurt the ball and cost the ball a life are the stone blocks on the ground and the ice blocks that are positioned above the ground but within the height range of the ball as it bounces. The ball has to avoid these obstacles by switching lanes and ducking the obstacles, respectively. Once the control for ducking is performed, the ball decelerates and comes to a complete stop as long as the gesture is being performed. Once the gestures’ action has stopped or changed, the ball accelerates and is on its way again. The winter mini-game requires the use of three gestures to play it, namely the swipe left, swipe right and stop gestures. It is also played using the keyboard controls of ‘a’ and ‘d’ keys (‘a’ to move left and ‘d’ to move right) and the spacebar to duck under the ice blocks.



**Figure 4.2:** A screenshot of the mini-game for the gorilla arm experiment

The participant plays this mini-game across all the three conditions of input controls namely supported gestures, mid-air gestures and keyboard controls.

#### **4.5 Procedure**

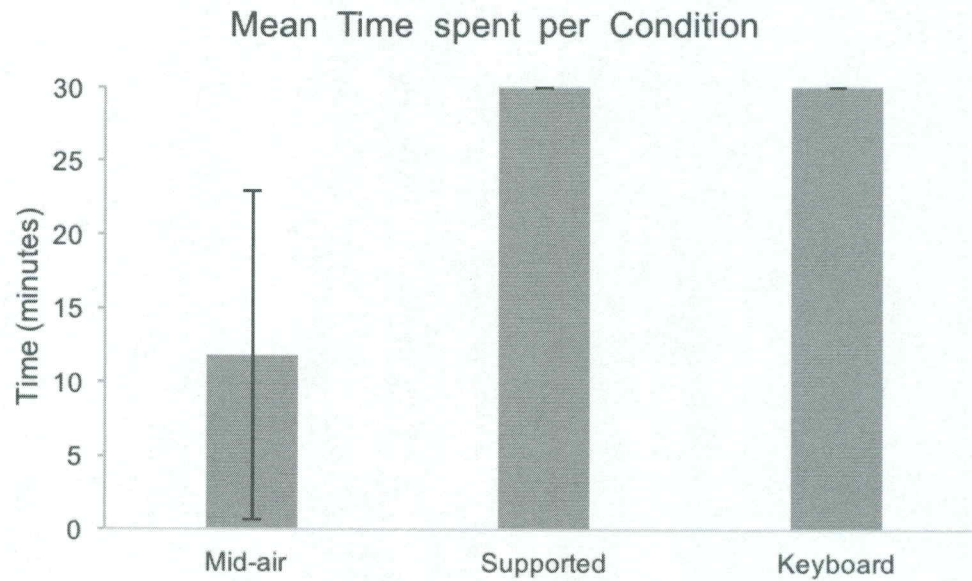
The time duration of the experiment lasted for 2 hours for each participant. The participant placed the motion capture suit on and was subjected to necessary calibration with the motion capture suit, which was approximately a 2 minute process. The participant was given instructions on the different gestures used to play the game and how to perform them. Once this was done, the participant played the game across all the three conditions for 30 minutes each. After each condition, a rest period of 5 to 10 minutes was given so that the participant could get suitable rest for their arms. During that time, the participant was asked to provide their level of perceived exertion



from the condition in which they played the game. Then the game play with the next condition was started and this process was repeated. If a participant felt unbearable discomfort on their arms or hands while playing the game in any of the condition, they would signal the experimenter that they wanted to stop the game play for this particular condition. The experiment ended when the participant played the game using all three conditions and gave their feedback on the level of physical exertion they perceived. The motion capture suit was then removed at the conclusion. This user study “Gorilla Arms” was approved by the University of Alabama in Huntsville’s Institutional Review Board (IRB) of Human Subjects Committee. The approval letter can be found in the Appendix, Figure A.1.

#### **4.6 Results**

The research hypothesis was that the position of the hands and arms with which the supported gestures were performed would cause significantly less physical exertion than the position of the arms and hands in which traditional mid-air gestures were performed. Since each participant went through 30 minutes of game-play in each condition, time can be used as a measure to test our hypothesis. The more time the participants endured in a condition, the more comfortable that input mode is to use. The supported gesture input and keyboard input condition was played for the entire 30 minute session by all the participants. However, only 27% of the participants could complete the whole 30 minute period in the mid-air gesture condition. Figure 4.3 displays the mean time spent across each condition. A significant difference in game-play time was shown when a paired samples t-test was conducted to compare the time



**Figure 4.3:** Mean time spent out of a possible 30 minutes for each condition. Error bars represent standard deviation.

spent in supported gesture condition and mid-air gesture condition. The following results were obtained: supported gesture condition ( $M=30.0$ ,  $SD=0.0$ ) and mid-air gesture condition ( $M=11.85$ ,  $SD=11.13$ );  $t(15) = -6.52$ ,  $p=0.00$ . The results were the same for the supported gesture condition and keyboard. Results were obtained by conducting a paired samples t-test through SPSS software, which is a software tool capable of analyzing data and provides tables and graphs based on its analyses.

Perceived exertion: A Borg CR10 scale was used to measure the perceived exertion. Participants rated their level of perceived exertion after playing the video game in each of the keyboard, mid-air hand gestures, and supported gestures conditions using a Borg CR10 scale. This data was later analyzed. In our study, the outcome variable (Borg CR10 scale rating) was collected (measured) multiple times

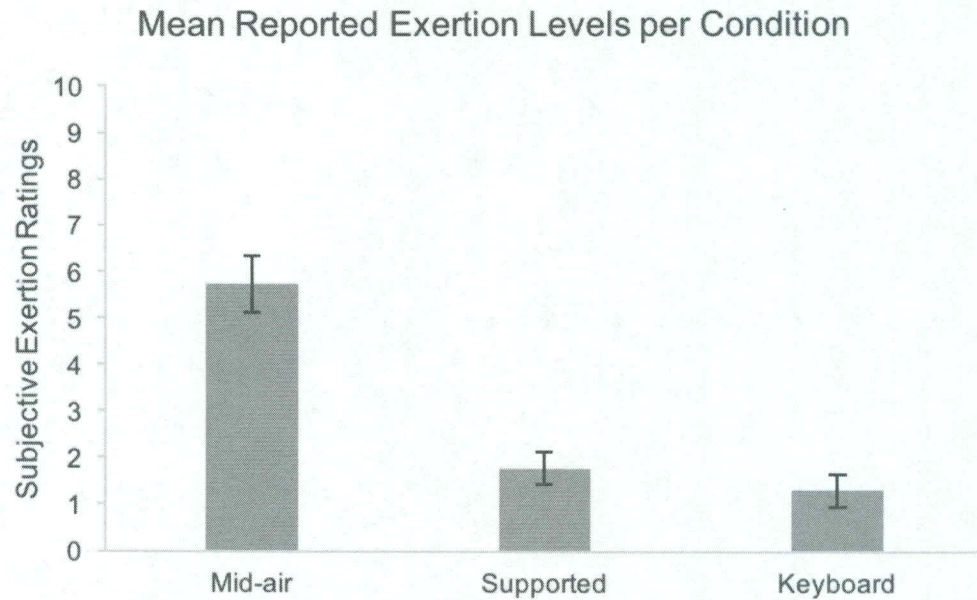
for each individual and this data was correlated within the observations of an individual. This is called the longitudinal data and requires special statistical techniques that take into account the correlation while processing the data.

There are a good number of statistical analyses to analyze longitudinal data with their ability to investigate the development of only one continuous variable over time and compare it between different groups (e.g., MANOVA [62]). However, these methods are not useful when we have to investigate and compare the developments of more than one continuous variable over time or analyze the relationship between one continuous outcome variable and several predictor variables [63]. These methods provide results on response data that is normally distributed. Most of the time, the response data is scattered and these methods result in inefficient estimation of regression parameters. The researchers are forced to omit the response data that do not fall within the normal distribution that makes these methods biased [64].

The generalized estimating equations (GEE) technique is a powerful analytical tool that accounts for the correlation of responses and is also flexible enough for responses that aren't normally distributed [65]. Another advantage of GEE includes analyzing the relationships between the variables at different time points simultaneously. The GEE uses the quasi-likelihood procedure to estimate regression coefficients, following an iterative approach [66]. The GEE hosts different kinds of correlation matrices. A few examples are independent correlation matrix where repeated measurements are correlated, M-dependent correlation matrix where a common correlation coefficient is shared between consecutive measurements, exchangeable correlation

matrix where elements have homogeneous correlations between them, unstructured correlation matrix which is a completely general correlation matrix, etc. [66].

A GEE model with an unstructured co-relation matrix was used was used to analyze the observations in perceived exertion across each of the conditions. Here the outcome variable is the rating of the perceived exertion obtained from the participants. A significant difference in results were observed across the conditions (Wald chi-square = 71.58,  $p \ll 0.001$ ,  $N=47$ ). Parameter estimates are found in Table 4.1. A Wald chi-square test is the test that is conducted to check if any variables that are supposed to be independent are significant (i.e., if such variables are contributing to the model in any way). If not, such variables can be removed. Since we do not have any such variable in our study, this value can be ignored. The differences in estimated marginal means were studied using pairwise comparison. The results are: supported gestures ( $M=1.70$ ,  $SE=0.34$ ), mid-air gestures ( $M=5.69$ ,  $SE=0.57$ ) and keyboard ( $M=1.3$ ,  $SE=0.34$ ) (Table 4.2). There is a significant difference between the supported gesture condition and the mid-air gesture condition with a mean difference of -3.99. There was also a significant difference between mid-air gestures condition and keyboard condition with a mean difference of 4.44. There was a difference between the supported gestures and the keyboard but it was very small when compared to the other two comparisons. The mean difference that was observed was 0.46. The Borg CR10 scale label that represented the mean supported gesture condition was “very easy” and that for the keyboard condition was “very very easy.” But, the mid-air gesture condition’s mean equated to the label “hard” on the Borg CR10 scale. Figure 4.4 represents the mean subjective Borg ratings for physical exertion across



**Figure 4.4:** The mean subjective Borg ratings for physical exertion across conditions (0=Nothing at all, 10=Impossible). Error bars represent standard error of the mean.

**Table 4.1:** Parameter estimates from the GEE model for the perceived exertion across the interaction conditions.

Models	$\beta$	SE	Wald Chi-Square	<i>p</i> -value
Intercept	5.69	0.57	98.66	$\ll 0.001$
Keyboard & Mid-air	-4.44	0.52	71.57	$\ll 0.001$
Supported & Mid-air	-3.99	0.52	59.73	$\ll 0.001$

conditions. Finally, the perceived exertion results confirm that the traditional gestures caused a lot more exertion when compared to that of the keyboard condition, where as the exertion caused by supported gestures was very close to the exertion caused by keyboard condition.

**Table 4.2:** Multiple comparisons and mean differences in perceived exertion by interaction types.

Comparison	Mean Difference	SE	<i>p</i> -value
Supported vs. Mid-air	-3.99	0.52	≪0.001
Supported vs. Keyboard	0.46	0.21	0.03
Mid-air vs. Keyboard	4.44	0.52	≪0.001

## 4.7 Discussion

Present-day digital data varies from simple plain text documents to a large collection of digital images, high-definition videos, or complex mathematical data. While the amount of data continues to grow, the types of inputs used to interact with the data are still confined to traditional ways such as using keyboards, mice, or touchscreens. Using these traditional input modalities to manipulate large amounts of data could be inefficient. Also, the growing virtual reality (VR) and augmented reality (AR) industries call for a new input modality as the traditional input modalities are ruled out as options for interacting with objects/data in VR and AR environments. This is because, the user needs the advantage of vision to use a keyboard, mouse, or a touchscreen, whereas the users have no visual access to their external surroundings while using a head-mounted display (HMD). This calls for an input modality that does not require the user to constantly have visual access to some device. Hand gestures can provide a means for solving this problem because the user is consciously aware of the actions being performed by his hands without having to view the actions. While gestures have been used as a type of input for interacting with data in the past, the gorilla arm syndrome is a potential issue that is holding back the technology com-

munity from developing systems that use hand gestures as a primary input modality for prolonged interaction. This is because the common position for both vertical touchscreen use and mid-air gesture input is with the arms being extended in front of the user at around shoulder height [26]. This was found to be the most fatiguing physical position out of the four positions that were investigated by Ramos et al. [26]. Adding weight to this, in our study, the mid-air gesture condition that required 30 minutes of game-play was completed by less than 25% of the participants. This also provides evidence that people are unable to use these gestures for longer periods of time. In the case of desk workers who interact with a system for long hours, it would be highly unlikely to continually use gestures in positions that cause high amounts of physical exertion as demanded. Furthermore, the evidence from recorded perceived physical exertion on the BorgCR 10 scale also shows that it is extremely difficult to use mid-air gestures for extended periods. Here, an extended period of time can be considered as a normal work shift period of 8 hours where a desk worker interacts with an interface using mid-air hand gestures. Our study aimed at developing a method to collectively address the issues discussed above. The results of the study helped provide evidence that supported gesture interactions induce significantly less amount of physical exertion compared to mid-air gesture interactions. The exertion induced due to supported gesture interactions was similar to the exertion induced while using a keyboard. This can be explained by the fact that the hand positions in performing supported gestures were similar to the hand positions normally used while working with a keyboard. This provides strong evidence that supported gestures can be used for prolonged periods of time. Hence, if there is any better input method that can be

a viable alternative to the traditional input methods and could also be used in VR and AR environments, then it is supported gestures.



## CHAPTER 5

### THE RELIABILITY OF HAND GESTURE ENABLED INPUT IN VIDEO GAMES: A STUDY

Video games are the choice of entertainment for many children and even young adults. Most video games are played for fun and provide recreation. They also help in fostering certain qualities in individuals such as competitiveness, a sense of pride in accomplishing goals, co-operativeness [67] (in multi-player games especially), learning lessons from mistakes, concentration, etc. According to a report by the Entertainment Software Association [68], as of 2015, 155 million people in America play video games regularly and 4 out of 5 American households have a device to play video games.

Input devices that are used to play video games have been evolving over time. Some of them include conventional methods like keyboards, mice, joysticks, game controllers, etc. However, motion capture devices are not far behind in being used as input devices to play video games. Microsoft Kinect is a vision based motion capture device that is quite famous for being used to play video games on Xbox game consoles. Other similar devices include Leap Motion controller, Wii remote controller, etc. However, due to the limitations of these devices in recognizing fine gestures, tracking finger movements, and supporting natural gestures, the applications or games from

these devices are not very popular among users, as they do not support long hours of game play and thus do not give a satisfactory experience to users. Hence the reliability of these devices is low [42,43], and when the reliability is low, we cannot expect games played using such devices to keep the users engaged or provide a pleasant experience.

In this chapter, we present games that take advantage of supported, low-fatigue hand gestures. The experiment presented in this chapter aims at studying the reliability of our gestures and the level of engagement of the users when playing a video game using these gestures. A total of 15 participants played variations of a video game using gestures and answered demographic questions along with a User Experience Questionnaire (UEQ). A video analysis of the participants was examined to study the reliability of the gestures in the video game environment.

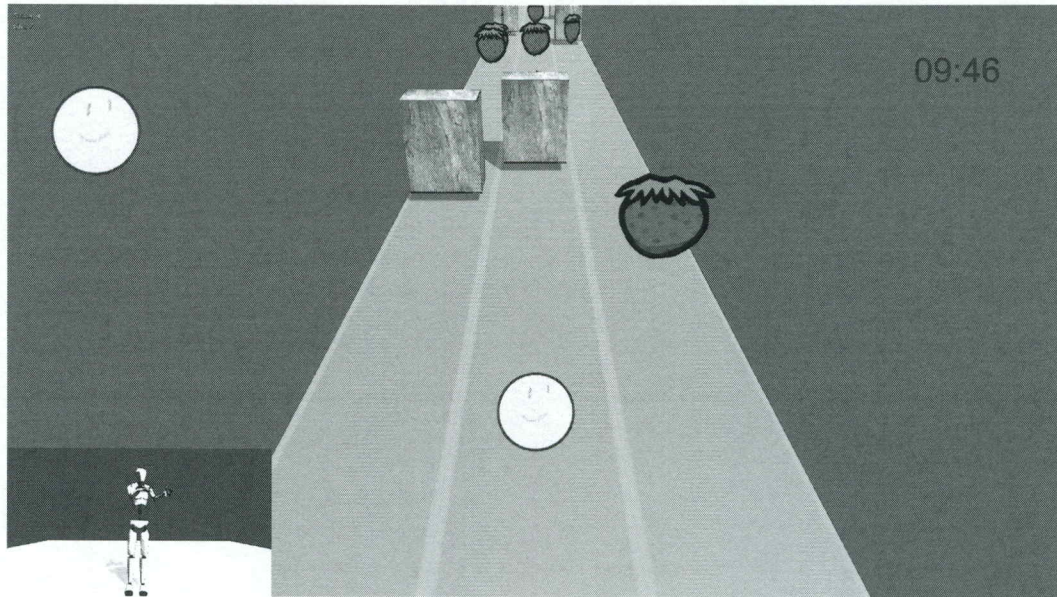
## 5.1 Game

A video game which comprised of four mini-games was developed. The game was titled “Happy ball” as the primary motto of the game was to keep the ball happy. The ball was the central character of the game. It was portrayed to be a joyful, elated and jovial ball that was to travel through all 4 seasons of a year without reaching a “crying” state. Each season of the year was played as a minigame. Each mini-game was finite. The primary purpose of the game was to survive as long as possible and score as many points as possible. The ball traveled through an infinite path that had both reward objects and obstacles. The score increased in increments of 1 when the ball collected reward objects. The ball had to avoid obstacles. If the ball collided with an obstacle or an enemy object, its emotion went from positive to

the next negative level. The ball had ten different emotions ranging from elated to crying. The highest level of emotion of the ball was when it was elated. The order of the ball's emotions then went to joyful, happy, satisfied, neutral, unhappy, sad, depressed, and crying, with crying being the lowest emotion and the most negative. When the ball began crying, the game ended and started from the beginning again with a score of 0. Hence the aim of the game was to keep the ball as happy as possible by avoiding obstacles and scoring as many reward objects as possible. The four mini-games depicted the four seasons that the ball traveled through in a year and were totally independent of each other. The mini-games were also named after four seasons: Spring, Fall, Winter and Summer. Each of these seasons incorporated a subset of gestures from the set of supported gestures that have been developed as the input to control the ball and to keep the ball's emotion from entering the "crying" state. Different subsets of supported gestures were used in each mini-game based on how the game was designed. Each mini-game also had a corresponding set of keyboard controls. The following sub-sections explain in detail the four mini-games.

### **5.1.1 Spring**

Spring is the season of rejuvenation, regrowth and positivity. The ball is always bouncing and travels advancing through the path in this season. The travel path through the spring season has three lanes between which the ball can switch anytime but is in one lane at any point of time and can switch only one lane per input action. The ball has to eat strawberries to score points and has to avoid stone blocks to protect itself from getting hurt which results in the ball losing the current



**Figure 5.1:** Screenshot of the spring minigame

emotional state and going the next negative emotional state. The strawberries are the reward objects that are the motivation for the ball to switch lanes. Also, the ball has to switch lanes to avoid obstacles. Here the only input control required is to switch lanes i.e., to move left and to move right. The input set used in each condition is: Supported gestures - swipe (swipe left and swipe right), Mid-air gestures - mid-air swipe (mid-air swipe left and mid-air swipe right) , Keyboard - 'a' and 'd' keys ('a' to move left and 'd' to move right).

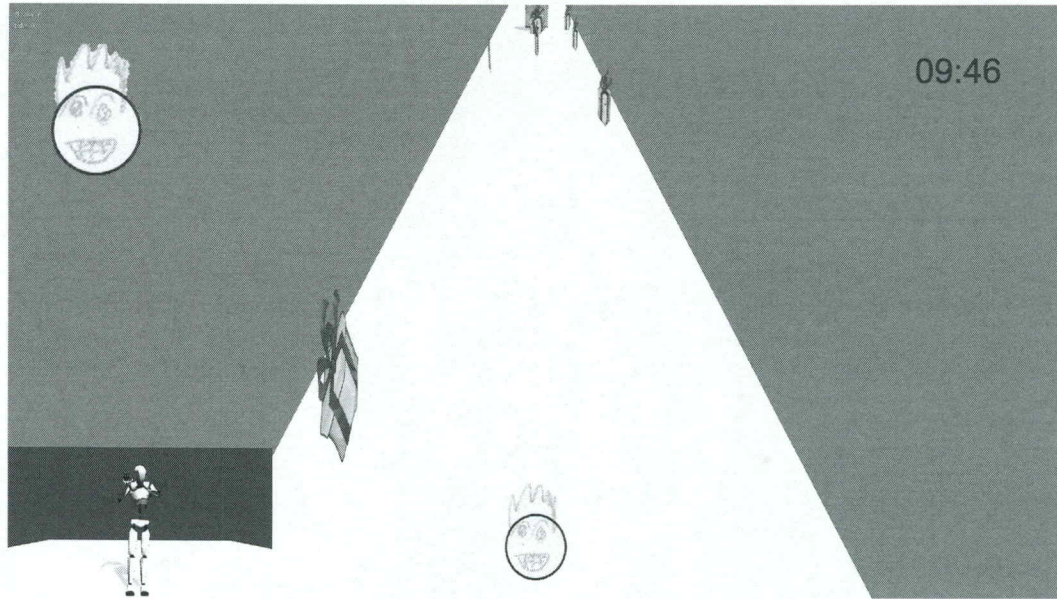
### 5.1.2 Winter

Winter is the season of freezing temperatures, snow and the holiday season. The ball travels by bouncing through the path in this season too. The travel path of the winter season also consists of three lanes between which the ball has to switch.

The reward objects in the winter season are Christmas gifts. Collecting Christmas gifts accumulates points. In this case, an obstacle of ice blocks in addition to the stone blocks is included. The ice blocks are placed horizontally across the path and they cover a length of two lanes and are above the path leaving space for the ball to duck in order to avoid them. The game is played by switching lanes to collect Christmas gifts and to avoid stone blocks and ice blocks if possible. The ice blocks can be avoided by switching to a lane where there is no ice block or by ducking below the ice block. The input set used for the supported gestures condition includes “swipe” (left and right) to switch lanes and the “stop” gesture to duck under the ice blocks. The input set for the mid-air gestures includes - “mid-air swipe” (left and right) and “mid-air stop” to duck under the ice blocks. The input set for the keyboard condition includes ‘a’ and ‘d’ keys (a to move left and d to move right) and the spacebar to duck under the ice blocks. When the input action for the ball to duck is performed, the ball decelerates and the bouncing slows for around 2 seconds before coming to a complete stop. As long as the input action is being performed, the ball will be at complete rest (i.e., it will be still). Once the input action is withdrawn, the ball accelerates back into motion. It is important to time the input action for ducking (i.e., stopping) the ball so that it does not hit the ice block and stoop lower on the emotioanl scale.

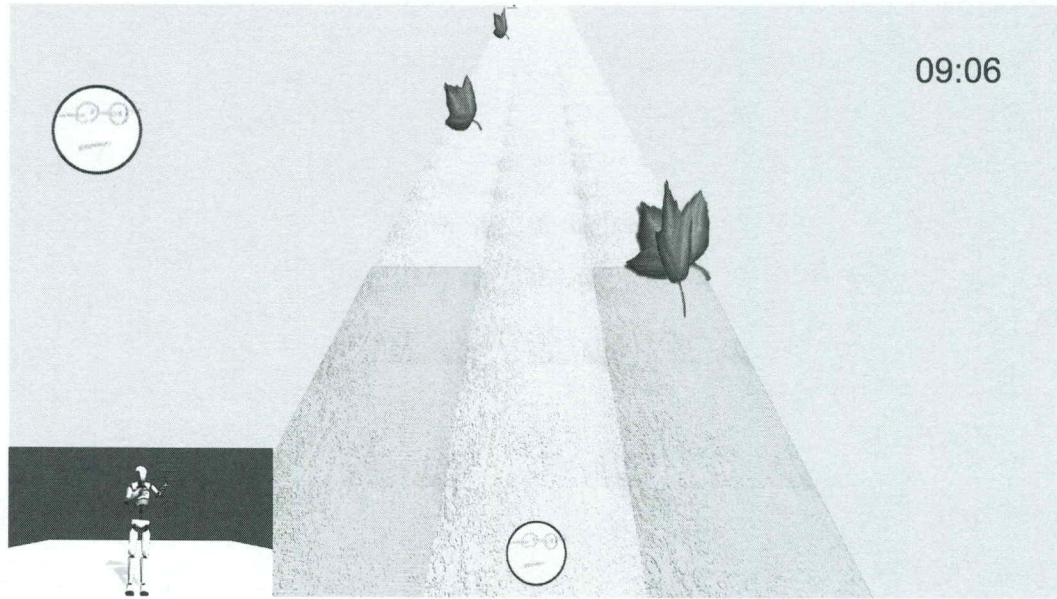
### **5.1.3 Fall**

Fall is the season of leaves falling everywhere and cold starting to creep in. In this mini-game, unlike the other two explained above, the ball travels by sliding through only one lane without bouncing. The fall season consists of dry leaves as



**Figure 5.2:** Screenshot of the Winter minigame

reward objects and stone blocks as the obstacle objects. The leaves and stone blocks appear on both sides of the ball's lane. The goal of this game is to keep the ball happy for the maximum time and not letting the ball go into the "crying" emotion state. The ball collects the leaves by picking them and dropping them on itself. The ball picks up the stone blocks and throws them out of the travel path to avoid becoming sad. Every leaf picked and dropped in results in a one point increase in the score. Every stone obstacle missed or dropped in towards the ball instead of thrown-out has a penalty of the ball's emotion moving to the next negative state. The game-play uses an input action of picking to drop in the leaves towards the ball and to trash out the stone blocks away from the travel path. The input set used in the supported gestures condition includes the "pick" gesture to pick objects, the "drop in" gesture to drop objects towards the ball, and "trash out" to throw away the stone blocks out



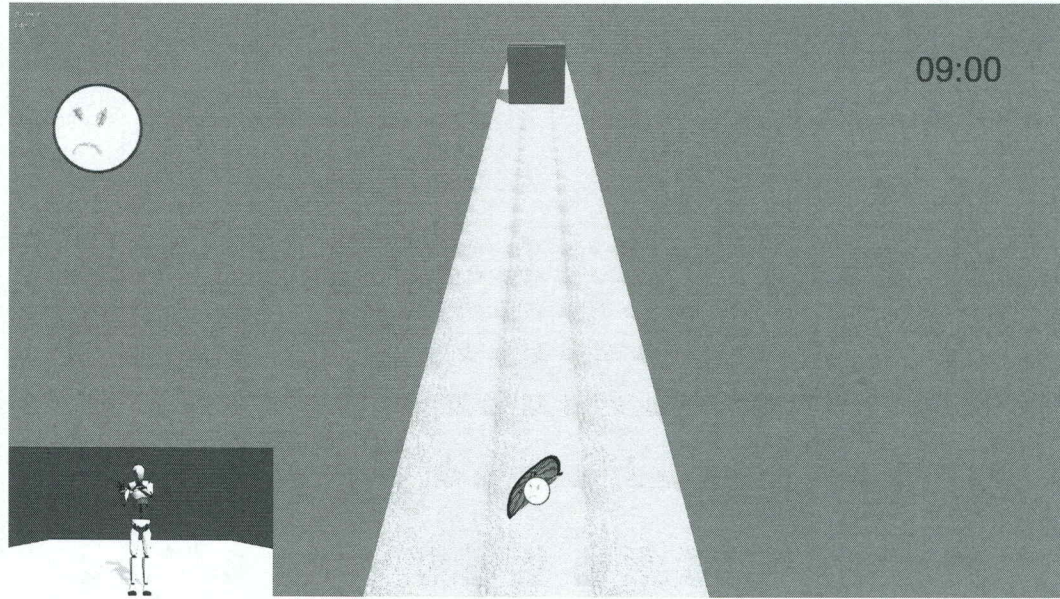
**Figure 5.3:** Screenshot of the fall minigame

of the travel path. The input set used in the mid-air gestures condition includes the “mid-air pick” gesture to pick objects, the “mid-air drop in” gesture to drop objects towards the ball, and “mid-air trash out” to throw away the stone blocks out of the travel path. The input set used in the keyboard condition includes ‘s’ key to pick up objects on the left hand side of the path, the ‘a’ key to trash objects out and the ‘d’ key to drop objects in. For the right side of the path, the ‘k’ key is used to pick up objects, the ‘l’ key is used to trash objects out and the ‘j’ key is used to drop objects in. The gestures performed by the left hand will only affect objects on the left side of the ball’s travel path and the gestures performed by the right hand will only affect the objects on the right side of the ball’s travel path. In other words, a single hand is able to manipulate objects only on one side of the travel path, but both hands can manipulate objects in the same way achieving same results.

#### 5.1.4 Summer

Summer is the season of heat, vacation and fun. The ball plays a fun game, but in this fun game, the ball can get sad. The game is similar to the game “Red Light - Green Light”. In this mini-game, the ball is initially at rest at a starting point and there is a traffic light at a distance in front of the ball. The traffic light is basically a block with a red light on one side and a green light on the other side. The block rotates and randomly stops with either the green light side or the red light side facing the ball. Near the traffic light is a watermelon. The goal is to go fetch the watermelon and bring it back to the starting point. The ball can be moved forward or backward by user’s actions. The goal of the mini-game is to move the ball forward whenever there is green light or when the block is rotating and to stop before the red light fully faces the ball. Once the red face begins turning, the ball can move forward again until it collects the watermelon. If there is a green light or the block is rotating, the block can be pulled back along with the watermelon. The ball should be stopped before the red light flashes. If the ball is in motion when the red light flashes or if the ball is moved while the red light is flashing, the ball loses its current emotion state and moves to the next negative emotional state. There are three input actions in this mini-game. One to move forward, one to pull backward and one to stop. The input set used for the supported gestures includes the “Go” gesture to move forward toward the watermelon, the “Come” gesture to pull the ball backward and the “Stop” gesture to stop the ball from movement. The input set used for the mid-air gestures includes the “mid-air go” gesture to move forward towards the watermelon, the “mid-air come”





**Figure 5.4:** Screenshot of the summer minigame.

gesture to pull the ball backward and the “mid-air stop” gesture to stop the ball from movement. The input set for the keyboard condition includes ‘w’ and ‘s’ keys (‘w’ to move forward and ‘s’ to move backward) and the spacebar to stop. For every watermelon collected and brought back to the starting point, one point is scored. For every movement made or attempted when the red light is flashing, the ball loses its current emotional state and goes down to the next negative emotional state.

## 5.2 Experiment

An experiment was conducted to study the accuracy and assess the reliability of supported gestures in a video game environment using gestures. A total of 15 participants were recruited from The University of Alabama in Huntsville. There were no inclusion or exclusion criteria. The participants were allowed to participate

in random on voluntary consent. The participants played the winter, fall and summer mini-games for 5 minutes each. The spring mini-game was excluded from the study as the winter mini-game includes the same swipe gestures. The games were played with the newly created supported hand gestures. The user experience questionnaire (UEQ) [69] was administered after each mini-game and the answers given by the participants were examined and evaluated.

### 5.3 Apparatus

Motion Capture system: The Perception Neuron Mocap system with IMU sensors was used in this experiment. An upper body strap with four sensors, an arm strap with one sensor on each hand and a glove on each hand with nine sensors each were used for motion tracking. The glove consisted of eight sensors on the fingers: one sensor each on the ring finger and pinky finger, and two sensors each on the middle finger, thumb and index fingers. There was also a sensor on the wrist. The lower body straps and the head strap were not used since only the hands and arms were needed to perform the gestures. This suit uses IMU based sensors implanted on wearable straps and gloves. The motion capture system is connected to a hub through a cable from the hip and the hub is connected to the computer through WiFi. The Perception Neuron system uses the Axis Neuron software to receive data from the hub and map the movements on to a kinematic skeleton. The Axis Neuron software in turn sends the data to the Unity game engine that runs the Axis Neuron Unity Plug-in and the game.

Game: The Winter, Summer and the Fall mini-games are used. The games were described in Section 5.1.

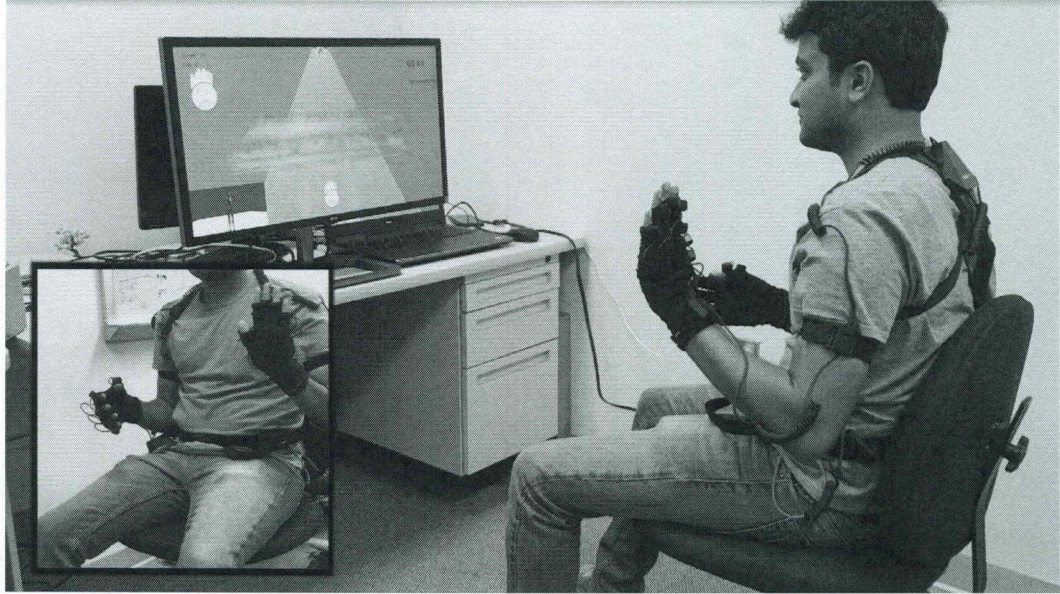
Both the motion capture system software and the game were installed on a single computer.

#### **5.4 Procedure**

Each participant was given a consent form to read and sign. This was followed by a demographic questionnaire which included questions about previous gaming experience. It included questions like: (1) "Have you played gesture based games before? If yes, name the games you have played.", (2) "How often do you play gesture based games?", (3) "How good are you at playing gesture based games?", etc. The participant then put on the motion capture suit and the suit was calibrated. This took about 5 minutes. Verbal instructions were given to the participant about each mini-game's background, game-play and the gestures used in each mini-game were demonstrated. The participant played the mini-game for two minutes as a practice session followed by 5 minutes of recorded game-play. This was done for all the three mini-games. After the participant played each mini-game, he was asked to answer the User Experience Questionnaire (UEQ) which is a general user interface evaluation survey and is a leading online free user experience evaluation and assessment system. It consists of a rating scale from -3 to +3 where -3 is the lowest and +3 is the highest for each metric. The UEQ consists of a set of 26 parameters that are grouped categorically into 6 individual scales: attractiveness (aesthetic), perspicuity (easy to learn), efficiency (complete their task), dependability (in control of the interaction),

stimulation (exciting and motivating) and novelty (innovative approach). A video recording was made for each game-play of the participant. This video was used to analyze the reliability and accuracy of gestures in a real time scenario. After the user study was completed, the experimenter asked a few questions to the participant regarding their experience. The participants were asked to rate each game-play based on their perceived level of engagement and also the perceived reliability of gestures during the game-play on a scale of 1 to 10 with 1 being the lowest and 10 being the highest. Additionally, the participants were asked if the gestures seemed natural or if they wanted to change the gestures to any other different gestures. Finally their opinion on their experience was recorded. This user study was approved by The University of Alabama in Huntsville's Institutional Review Board (IRB) of Human Subjects Committee. The approval letter can be found in the Appendix, Figure A.2.

A total of 15 participants participated in the user-study (10 males and 5 females). The age of the participants ranged from 21 to 28 with a mean age of 23.8 years. From the answers given by the 15 participants regarding their previous gaming experience, we observed that 7 participants had not played any gesture based games before, 7 had played a gesture based game or two in the past with a frequency of less than once a year, and one participant played gesture based games using a Wii remote and a Kinect at least once a week. Out of the 7 participants who said they had played gesture based games before, 3 had used only a Kinect, 2 had used only a Wii remote, 1 had used Leap Motion controller and 1 had used both a Kinect and a Wii remote controller to play gesture based games. Fourteen out of 15 participants said they had very little or no skill playing gesture based games.



**Figure 5.5:** An example frame of the user study showing the user performs the stop gesture in the winter game to make the ball slide under the ice block.

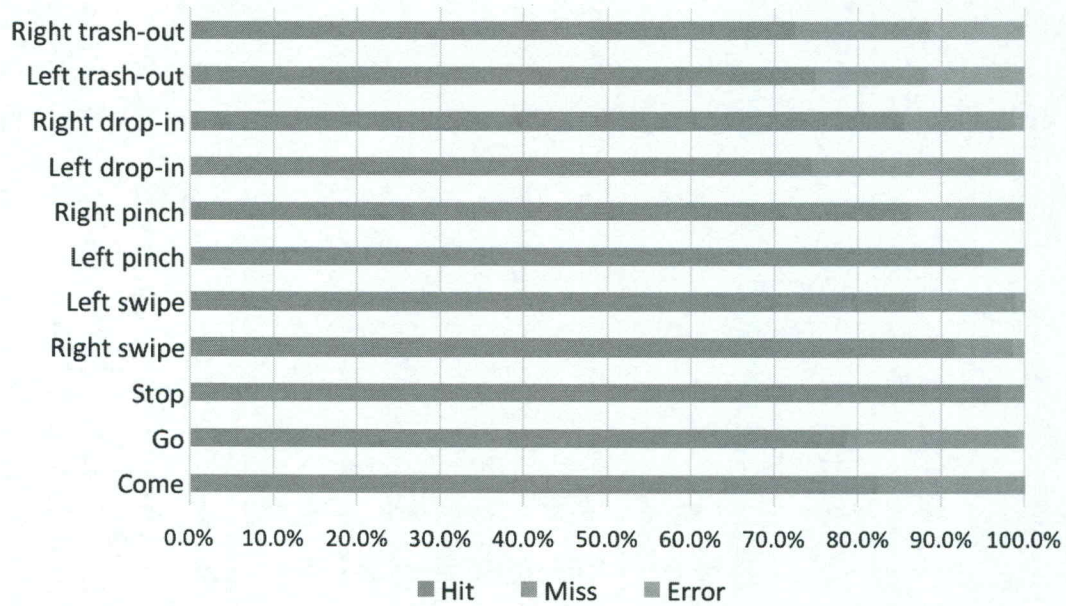
## 5.5 Results

**Accuracy:** The accuracy of the gestures used in the mini-games, namely “swipe”, “stop”, “come”, “go”, “pinch”, “drop-in” and “trash-out”, was assessed for each game. The accuracy of “pinch”, “drop-in”, “trash-out” and “swipe” gestures was assessed separately for the right and left hands because these gestures slightly differ when performed with different hands. The “stop”, “come” and “go” gestures are performed exactly the same way using either of the hands and usually with only one hand throughout the game. The accuracy assessment was carried out manually by watching the videos of game-play in slow-motion and categorizing each gesture action performed by the participant into one of the three categories: hit, miss and error. A gesture action was categorized as a hit if the gesture performed was correct and it

triggered the intended action in the game. A gesture action was said to be a miss if the gesture was performed but the intended action did not occur. A gesture action was categorized as error if the gesture performed resulted in an unintended action.

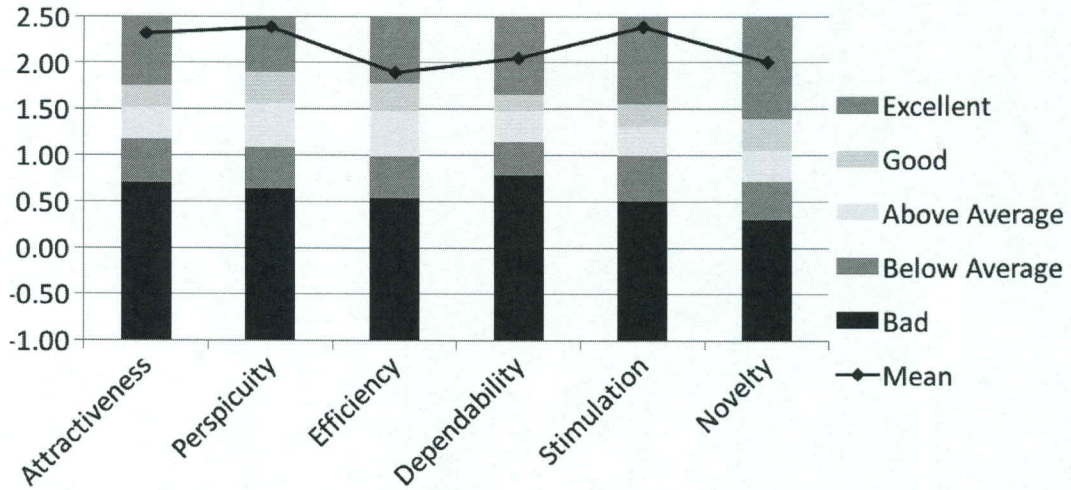
The accuracy of results of all the gestures evaluated are shown in Figure 5.6. The overall mean accuracy of all the gestures was 84.1%. The “stop” gesture was the most accurate gesture with an average accuracy of 97%. Since the “stop” gesture was used in two mini-games, the accuracy was averaged across the two. The least accurate gesture was the right hand “trash-out” with an average accuracy of 72.6%. The error rate was low at an average of 2.6% overall. The left hand “trash-out” had an average error rate of 11.7% and the right hand “trash-out” had an average error rate of 11.4% which were the highest error rates observed. The “swipe” gesture had the next highest error rate of 1.3% which was substantially lower. The highest average miss rate of 24.5% was seen in the left hand “drop-in” gesture. The next highest average miss rate was 2.4% seen in the “stop” gesture which is again substantially lower.

UEQ: The UEQ responses were recorded during the experiment and there were 3 UEQ measurements per participant. Figure 5.7 shows the mean scores of all the 6 scales namely Attractiveness, Perspicuity, Efficiency, Dependability, Stimulation and Novelty, across the three mini-games. The figure also compares the scores obtained from our experiment with the measurement’s average scores obtained from 9905 other participants from 246 studies, as of September 2016. This shows the relative quality of the user experience with the game and gestural interface [69]. The UEQ’s average scores were obtained from other user experience studies involving websites, business software, social networking sites and Internet commerce sites. The scores from this

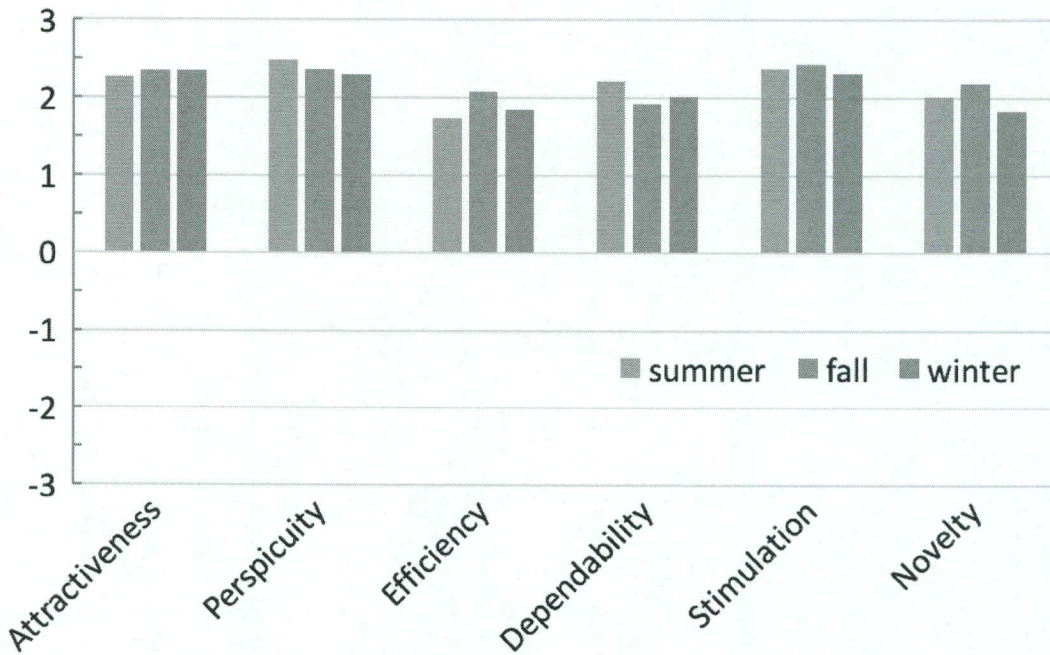


**Figure 5.6:** Gesture accuracy scores for each gesture according to accurate hits, misses, and errors.

study were in the top 10% range. The UEQ report showing the result is shown in the Appendix, Figure A.3. An ANOVA (Analysis of Variance) was applied to each of the UEQ's scales. ANOVA is a common statistical method for comparing the difference between two or more means. Since only a single factor (user's response) was being analyzed, a one way ANOVA test (also called single-factor ANOVA) was used. There was no significant difference observed across the three mini-games, Attractiveness,  $F(2,41) = 0.06$ ; Perspicuity,  $F(2,41) = 0.26$ ; Efficiency,  $F(2,41) = 0.63$ ; Dependability,  $F(2,41) = 0.67$ ; Stimulation,  $F(2,41) = 0.11$ ; Novelty,  $F(2,41) = 0.38$ ; where a 95% confidence level ( $p=0.05$ ) was used for every scale. The results suggest that the user experience was relatively stable across each game, as shown in the Figure 5.8.



**Figure 5.7:** The mean UEQ scores for each scale across all 3 games and compared with UEQ scores from other studies.



**Figure 5.8:** Mean UEQ scores for each scale for each game type, Summer, Fall, and Winter. Scales are in the range of [-3,3], scores bigger than 0.8 represent a positive evaluation, scores smaller than -0.8 represent a negative evaluation, and scores between -0.8 and 0.8 represent a neutral evaluation.



User Experience: The participants were asked to give their opinion on their experience playing the happy-ball game. The answers were all positive with common remarks such as, “Exciting”, “Interesting”, “Inventive”, “Real fun” and “Enjoying”. The participants were asked to rate their perceived level of engagement in each mini-game on a scale of 1 to 10 with 1 being the lowest and 10 being the highest. The average rating for the Summer game was 8.6, for the Fall mini-game was 8.7 and for the Winter game was 9.0. Similarly, the participants were asked to rate the perceived reliability of gestures for each mini-game and the average rating for the Summer mini-game was 8.8, for the Fall mini-game was 8.1, and for the Winter mini-game was 8.9. The participants were also asked if they were satisfied with the naturalness of gestures. All participants said “Come”, “Swipe” and “Stop”, “Pinch” and “Trash” were very natural, and that they would not change these gestures. 12/15 said the “Go” gesture was natural, and they would not change the gesture. The other three participants said “Go” was natural but could be improved.

## 5.6 Discussion

Ideally all gestures should exhibit a high level of accuracy, as can be seen in the “Stop” gesture (97%). However, the low accuracy of the “Drop-in” and “Trash-out” gestures (72%) was easily accounted for when the video of the game-play was reviewed. Most often when the “Trash-out” gesture resulted in an error, the participant performed the gesture action, but the system recognized it as some gesture other than the intended one. It was observed that the participants would initiate the “Trash-out” gesture action with a downward movement before the palm was upward

resulting in a ‘cocking’ action. Given that the gesture was based on the natural action of picking something up and throwing something away, a person may try to add a force before giving the throw direction by pulling his/her hand backwards first and then propelling it forward in the direction of the intended throw. This indicates a minor flaw in the gesture design that leads to a conflict between the “Drop-in” and “Trash-out” gestures. The low accuracy rates with the “Drop-in” and “Trash-out” gestures can be accounted for with this conflict.

Another possible reason that the accuracy level of gestures was not as expected was that the accuracy was measured in a real time environment (i.e., a continuous game) with participants who had little or no skill playing gesture based games instead of a controlled environment. In previous studies, though some of them use vision based systems that are deemed to provide inaccurate hand and finger tracking, high rates of accuracy are seen. However, the accuracy measurements in almost all of those studies have been performed in a controlled environment with/without an interactive setup but not in a real-time environment like ours. These had the participants consciously performing gestures with very little or no need to perform gestures to interact with a system. For example, in Luzhnica et al. [32], participants sat in front of a computer, watched a video of how to perform that particular gesture, had an on-screen instruction of when to start performing a gesture, and then the participants performed the gesture. This data was recorded and then fed into a learning algorithm for gesture recognition. In our study, the participants interact in real-time with a video game where the participants are required to perform the gestures to play the video game with their own cognitive ability and no other intrusions. Also, our

gesture recognition system recognized gestures in real time (with no apparent latency issues). Due to a real-time interaction with the video game, it was also observed that participants tended to forget gestures. More precisely, participants often became so engaged in the game play that they would forget the right way of performing a gesture in the game and would perform the gesture incorrectly. For example, some participants would perform the “Come” and “Go” gesture with their palm oriented sideways while the palm needed to be oriented upwards for “Come” and downwards for the “Go” gesture. This may also have contributed to why the gesture accuracy level was low. Future efforts to rectify these factors would be redesigning the “Drop-in” and/or “Trash-out” gesture so that there is no conflict between the gestures and to make changes in the design of the “Trash-out” gesture to suit the more natural “cocking” action as mentioned earlier in this section.

The game and the gestures together produced high levels of user satisfaction across all UEQ scales. Not all of the scales on the UEQ applied directly to gestures. Some of the scales applied more to the game and visualization rather than the gestures as input modality for controlling the video-game, like the scale for attractiveness. Efficiency and Dependability scales directly related to the gestures. Perspicuity, Stimulation and Novelty related to both the gesture and to the game. The scores for Efficiency and Dependability were low compared to the scores on other scales, although the overall scores were in top 10% of the scores when compared to other interfaces evaluated by the UEQ. Also, the user ratings for the gesture’s reliability were high. The data from these studies support that this type of gesture vocabulary

is promising as a viable input modality. However, there is lot of room for betterment in this area.

## CHAPTER 6

### CONCLUSIONS

The gesture recognition systems and interfaces built in the past suffer some drawbacks that significantly upset the whole idea of using gestures as an input modality for human computer interaction. One major drawback is that they force the users' hands into positions that induce high physical exertion, to perform gestures without giving them the privilege of performing these gestures in more natural resting positions. Also, most of the gesture recognition systems use learning algorithms to train the system which is a very cumbersome, laborious, intricate and not a developer-friendly process. We developed a method for recognizing both static and dynamic hand gestures that uses the concept of angular velocity and hand orientation. We developed an adaptive hand model with articulated joints of the fingers that possess high degrees of independence. Using this, we defined a gesture vocabulary that contains a small set of 6 hand gestures that allows the user to perform the gestures in a normal resting position. The adaptive hand model is simple and the methodology is very easy to understand, without involving learning algorithms or training of the system, and is developer friendly. It was shown that the newly developed gestures result in similar levels of perceived exertion as that of using a keyboard while the

traditional mid-air hand gestures show high levels of perceived exertion. The results obtained provide support for the purpose of the experiment. Also, the accuracy and reliability of the gestures was evaluated in real time use with the aid of gaming environments. It was found that different gestures possessed different accuracy levels but were found to be reliable and engaging. One possible limitation of this study would be that a limited gesture vocabulary set was used and the data was only collected over a 30 minute duration. In addition, an objective estimation of physical exertion (time) was used.

One of the future efforts will be to conduct a user study with participants performing the gestures for a prolonged amount of time (say continuously for 3 hours), and also to have an accurate objective measure of physical exertion. Other future efforts will be to improve the gestures by modifying their design where it is required and improving their accuracy. Expanding the gesture vocabulary by adding more gestures suits different needs in different applications. The best potential use of hand gestures would be in virtual reality and augmented reality environments. Data is increasing day by day and the organized storing of data is becoming an important aspect of data science. Images are a major part of growing data. Be it personal images or images from security cameras, there are millions of images that need to be sorted and stored efficiently. Since data is increasing, trying to sort images with traditional means of input like the mouse and keyboard or even touchscreen displays is a laborious task. Natural hand gestures are shown to be a promising means of user input, and can be used to sort images efficiently and naturally. Hence, the ultimate goal is to keep striving to improve the gesture framework and gesture vocabulary to

be more efficient, easy to use, and to try integrating it into different types of interfaces in which gestures can be the means of primary or secondary user input modality.

## APPENDIX





August 26<sup>th</sup> 2016

Sarah Meacham  
Research Associate I  
Systems Management and Production Center  
The University of Alabama in Huntsville

Dear Ms. Meacham,

The UAH Institutional Review Board of Human Subjects Committee has reviewed your proposal, *Gorilla Arms*, and found it meets the necessary criteria for continued meets the requirements outlined in 45 CFR 46.110 category (7) to receive expedited review. This study was found to involve no more than minimal risk and was approved. Your proposal is in compliance with this institution's Federal Wide Assurance (FWA) 00019998 and the DHHS Regulations for the Protection of Human Subjects (45 CFR 46).

Please note that this approval is good for one year from the date on this letter. If data collection continues past this period, you are responsible for processing a renewal application a minimum of 60 days prior to the expiration date.

No changes are to be made to the approved protocol without prior review and approval from the UAH IRB. All changes (e.g. a change in procedure, number of subjects, personnel, study locations, new recruitment materials, study instruments, etc) must be prospectively reviewed and approved by the IRB before they are implemented. You should report any unanticipated problems involving risks to the participants or others to the IRB Chair.

If you have any questions regarding the IRB's decision, please contact me.

Sincerely,

William Wilkerson  
IRB Chair  
Dean, Honors College

OFFICE OF THE VICE PRESIDENT FOR RESEARCH  
Von Braun Research Hall M-17      Huntsville, AL 35899

T 256.824.6100

F 256.824.6783

**Figure A.1:** Institutional Review Board approval letter for the user experiment to study the gorilla arm effect.



October 17<sup>th</sup> 2016

Chao Peng, PhD  
Assistant Professor  
Department of Computer Science  
University of Alabama in Huntsville

Dear Dr. Peng,

The UAH Institutional Review Board of Human Subjects Committee has reviewed your proposal, *Reliability and Engagement of Hand Gesture Enhanced Input in Gaming Environments*, and found it meets the necessary criteria for approval. Your proposal seems to be in compliance with this institutions Federal Wide Assurance (FWA) 00019998 and the DHHS Regulations for the Protection of Human Subjects (45 CFR 46).

Please note that this approval is good for one year from the date on this letter. If data collection continues past this period, you are responsible for processing a renewal application a minimum of 60 days prior to the expiration date.

No changes are to be made to the approved protocol without prior review and approval from the UAH IRB. All changes (e.g. a change in procedure, number of subjects, personnel, study locations, new recruitment materials, study instruments, etc) must be prospectively reviewed and approved by the IRB before they are implemented. You should report any unanticipated problems involving risks to the participants or others to the IRB Chair.

If you have any questions regarding the IRB's decision, please contact me.

Sincerely,

William Wilkerson  
IRB Chair  
Dean, Honors College

OFFICE OF THE VICE PRESIDENT FOR RESEARCH  
Von Braun Research Hall M-17      Huntsville, AL 35899

T 256.824.6100

F 256.824.6783

**Figure A.2:** Institutional Review Board approval letter for the reliability of hand gesture enabled input in video games: A study.

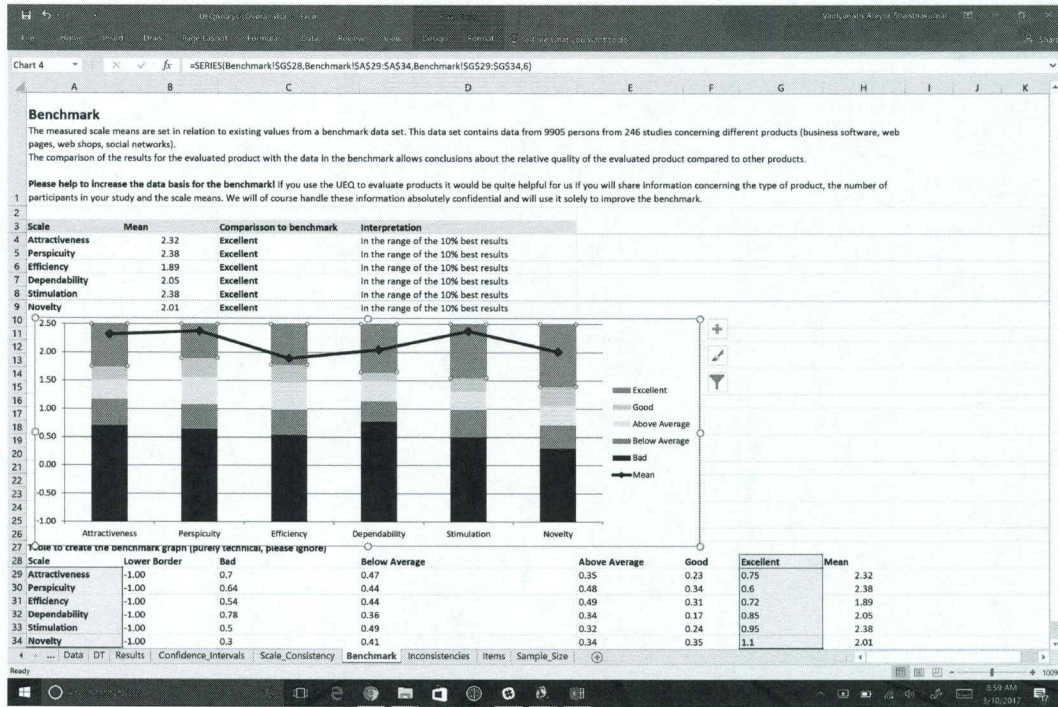


Figure A.3: The UEQ Analysis Report obtained from the results of the reliability of hand gesture enabled input in video games: A study

## REFERENCES

- [1] O. Ostill, "Portrait of an afro american police officer holding a hand up to motion 'stop' while blowing whistle on white background," in <https://www.shutterstock.com/search/traffic+officer>. [Online; accessed March 7, 2017].
- [2] T. W. Times, "Republican presidential candidate donald trump gestures prior to a campaign stop at winnacunnet high school in hampton," in <http://www.washingtontimes.com/news/2015/aug/21/donald-trump-no-ill-use-word-anchor-baby/>, 2015. [Online; accessed March 7, 2017].
- [3] hillaryiscoming@Tumblr, "Happy national thumbs up day! (august 1)," in <http://hillaryiscoming.tumblr.com/post/125669527063/happy-national-thumbs-up-day-august-1>, 2015. [Online; accessed March 7, 2017].
- [4] B. Tysh, "Golden state warriors," in <http://scoreboredsports.com/tag/lebron-james/>, 2016. [Online; accessed March 7, 2017].

- [5] Alamy, “Stock photo - a male new york city traffic enforcement officer directing cars at east 34th street and park avenue in new york city,” in <http://www.alamy.com/stock-photo-a-male-new-york-city-traffic-enforcement-officer-directing-cars-at-99356228.html>, 2016. [Online; accessed March 7, 2017].
- [6] MamamiaTeam, “Queen elizabeth ii,” in <http://www.mamamia.com.au/diamond-jubilee-and-11-things-you-might-not-know-about-the-queen/>, 2012. [Online; accessed March 7, 2017].
- [7] S. Gursten, “Good-riddance-to-2014-no-fault-reform-bills,” in <http://www.michiganautolaw.com/blog/2015/01/20/good-riddance-2014-no-fault-reform/>, 2015. [Online; accessed March 7, 2017].
- [8] S. Skjold, “asian-volunteer-policeman-age-23-directing-traffic-grand-old-day-street-fap6y2.,” in <http://www.alamy.com/stock-photo-asian-volunteer-policeman-age-23-directing-traffic-grand-old-day-street-92686806.html>, 2015. [Online; accessed March 7, 2017].
- [9] J. Jain, A. Lund, and D. Wixon, “The future of natural user interfaces,” in *CHI '11 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '11, Vancouver, BC, Canada, pp. 211–214, 2011.
- [10] A. Cooper, R. Reimann, and D. Cronin, *About face 3: the essentials of interaction design*. John Wiley & Sons, 2007.

- [11] R. McGloin and M. Krcmar, “The impact of controller naturalness on spatial presence, gamer enjoyment, and perceived realism in a tennis simulation video game,” *Presence: Teleoper. Virtual Environ.*, vol. 20, pp. 309–324, Aug. 2011.
- [12] D. McNeill, *Hand and Mind: What Gestures Reveal about Thought*. University of Chicago Press, Chicago, IL, 1992.
- [13] J. Norton, C. A. Wingrave, and J. J. LaViola, Jr., “Exploring strategies and guidelines for developing full body video game interfaces,” in *Proceedings of the Fifth International Conference on the Foundations of Digital Games, FDG '10*, Monterey, CA, pp. 155–162, 2010.
- [14] C. Silpasuwanchai and X. Ren, “Designing concurrent full-body gestures for intense gameplay,” *International Journal of Human-Computer Studies*, vol. 80, pp. 1 – 13, 2015.
- [15] K. A. Siek, Y. Rogers, and K. H. Connelly, “Fat finger worries: How older and younger users physically interact with pdas,” in *Proceedings of the 2005 IFIP TC13 International Conference on Human-Computer Interaction, INTERACT'05*, Rome, Italy, pp. 267–280, 2005.
- [16] D. Vogel and G. Casiez, “Hand occlusion on a multi-touch tabletop,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '12*, Austin, Texas, USA, pp. 2307–2316, 2012.

- [17] S. S. Arefin Shimon, C. Lutton, Z. Xu, S. Morrison-Smith, C. Boucher, and J. Ruiz, "Exploring non-touchscreen gestures for smartwatches," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, Santa Clara, California, USA, pp. 3822–3833, 2016.
- [18] S. Boring, M. Jurmu, and A. Butz, "Scroll, tilt or move it: Using mobile phones to continuously control pointers on large public displays," in *Proceedings of the 21st Annual Conference of the Australian Computer-Human Interaction Special Interest Group: Design: Open 24/7*, OZCHI '09, Melbourne, Australia, pp. 161–168, 2009.
- [19] M. Nielsen, M. Störring, T. B. Moeslund, and E. Granum, "A procedure for developing intuitive and ergonomic gesture interfaces for hci," in *International Gesture Workshop*, pp. 409–420, Springer, 2003.
- [20] M. R. Morris, J. O. Wobbrock, and A. D. Wilson, "Understanding users' preferences for surface gestures," in *Proceedings of Graphics Interface 2010*, GI '10, (Toronto, Ont., Canada), pp. 261–268, 2010.
- [21] J. O. Wobbrock, M. R. Morris, and A. D. Wilson, "User-defined gestures for surface computing," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, Boston, MA, USA, pp. 1083–1092, 2009.

- [22] R. Aigner, D. Wigdor, H. Benko, M. Haller, D. Lindbauer, A. Ion, S. Zhao, and J. T. K. V. Koh, “Understanding mid-air hand gestures: A study of human preferences in usage of gesture types for hci,” Tech. Rep., Microsoft Research, November 2012.
- [23] D. Rempel, M. J. Camilleri, and D. L. Lee, “The design of hand gestures for human–computer interaction: Lessons from sign language interpreters,” *International Journal of Human-Computer Studies*, vol. 72, no. 10–11, pp. 728 – 735, 2014.
- [24] J. Qin, M. Marshall, J. Mozrall, and M. Marschark, “Effects of pace and stress on upper extremity kinematic responses in sign language interpreters,” *Ergonomics*, vol. 51, pp. 274–289, March 2008.
- [25] J. J. DeCaro, M. Feuerstein, and T. A. Hurwitz, “Cumulative trauma disorders among educational interpreters. Contributing factors and intervention,” *Am. Ann. Deaf*, vol. 137, pp. 288–292, Jul 1992.
- [26] J. D. Hincapié-Ramos, X. Guo, P. Moghadasian, and P. Irani, “Consumed endurance: A metric to quantify arm fatigue of mid-air interactions,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI ’14, Toronto, Ontario, Canada, pp. 1063–1072, 2014.
- [27] W. Rohmert, “Ermittlung von erholungspausen für statische arbeit des menschen,” *European Journal of Applied Physiology and Occupational Physiology*, vol. 18, no. 2, pp. 123–164, 1960.



- [28] D. Xu, "A neural network approach for hand gesture recognition in virtual reality driving training system of spg," in *Proceedings of the 18th International Conference on Pattern Recognition - Volume 03*, ICPR '06, Washington, DC, USA, pp. 519–522, 2006.
- [29] C. Wang, Z. Liu, and S. C. Chan, "Superpixel-based hand gesture recognition with kinect depth camera," *IEEE Trans. on Multimedia*, vol. 17, pp. 29–39, Jan 2015.
- [30] S. S. Rautaray and A. Agrawal, "Vision based hand gesture recognition for human computer interaction: a survey," *Artificial Intelligence Review*, vol. 43, no. 1, pp. 1–54, 2015.
- [31] G. Marin, F. Dominio, and P. Zanuttigh, "Hand gesture recognition with jointly calibrated leap motion and depth sensor," *Multimedia Tools and Applications*, vol. 75, no. 22, pp. 14991–15015, 2016.
- [32] G. Luzhnica, J. Simon, E. Lex, and V. Pammer, "A sliding window approach to natural hand gesture recognition using a custom data glove," in *2016 IEEE Symposium on 3D User Interfaces (3DUI)*, Greenville, SC, USA, pp. 81–90, March 2016.
- [33] S. Alavi, D. Arsenault, and A. Whitehead, "Quaternion-based gesture recognition using wireless wearable motion capture sensors," *Sensors*, vol. 16, no. 5, p. 605, 2016.

- [34] Y. Song, D. Demirdjian, and R. Davis, "Continuous body and hand gesture recognition for natural human-computer interaction," *ACM Trans. Interact. Intell. Syst.*, vol. 2, pp. 5:1–5:28, Mar. 2012.
- [35] K. Liu and N. Kehtarnavaz, "Real-time robust vision-based hand gesture recognition using stereo images," *Journal of Real-Time Image Processing*, vol. 11, no. 1, pp. 201–209, 2016.
- [36] J. P. Wachs, M. Kölsch, H. Stern, and Y. Edan, "Vision-based hand-gesture applications," *Commun. ACM*, vol. 54, pp. 60–71, Feb. 2011.
- [37] K. Murakami and H. Taguchi, "Gesture recognition using recurrent neural networks," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '91*, New Orleans, Louisiana, USA, pp. 237–242, ACM, 1991.
- [38] P. Neto, D. Pereira, J. N. Pires, and A. P. Moreira, "Real-time and continuous hand gesture spotting: An approach based on artificial neural networks," in *IEEE International Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany, pp. 178–183, May 2013.
- [39] Z. Lu, X. Chen, Q. Li, X. Zhang, and P. Zhou, "A hand gesture recognition framework and wearable gesture-based interaction prototype for mobile devices," *IEEE Trans. on Human-Machine Systems*, vol. 44, pp. 293–299, April 2014.

- [40] M. Romaszewski, P. Głomb, and P. Gawron, “Natural hand gestures for human identification in a human-computer interface,” in *2014 4th International Conference on Image Processing Theory, Tools and Applications (IPTA)*, Paris, France, pp. 1–6, Oct. 2014.
- [41] C. Xu, P. H. Pathak, and P. Mohapatra, “Finger-writing with smartwatch: A case for finger and hand gesture recognition using smartwatch,” in *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, Santa Fe, New Mexico, USA, HotMobile ’15, pp. 9–14, 2015.
- [42] J. Guna, G. Jakus, M. Pogačnik, S. Tomažič, and J. Sodnik, “An analysis of the precision and reliability of the leap motion sensor and its suitability for static and dynamic tracking,” *Sensors*, vol. 14, no. 2, p. 3702, 2014.
- [43] M. A. Brown, W. Stuerzlinger, and E. J. M. Filho, “The performance of uninstrumented in-air pointing,” in *Proceedings of Graphics Interface 2014*, GI ’14, Montreal, Quebec, Canada, pp. 59–66, 2014.
- [44] G. Welch and E. Foxlin, “Motion tracking: no silver bullet, but a respectable arsenal,” *IEEE Computer Graphics and Applications*, vol. 22, pp. 24–38, Nov 2002.
- [45] C. E. Lang and M. H. Schieber, “Human finger independence: Limitations due to passive mechanical coupling versus active neuromuscular control,” *Journal of Neurophysiology*, vol. 92, no. 5, pp. 2802–2810, 2004.

- [46] T. Horton, S. Sauerland, and T. Davis, "The effect of flexor digitorum profundus quadriga on grip strength," *The Journal of Hand Surgery: British & European Volume*, vol. 32, no. 2, pp. 130 – 134, 2007.
- [47] A. H. Watt and M. Watt, *Advanced animation and rendering techniques*. ACM Press, New York, NY, USA, 1992.
- [48] R. A. Wachsler and D. B. Learner, "An analysis of some factors influencing seat comfort," *Ergonomics*, vol. 3, no. 4, pp. 315–320, 1960.
- [49] V. I. Pavlovic, R. Sharma, and T. S. Huang, "Visual interpretation of hand gestures for human-computer interaction: a review," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 677–695, Jul. 1997.
- [50] C. Hummels and P. J. Stappers, "Meaningful gestures for human computer interaction: beyond hand postures," in *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, Nara, Japan, pp. 591–596, Apr. 1998.
- [51] A. Ramamoorthy, N. Vaswani, S. Chaudhury, and S. Banerjee, "Recognition of dynamic hand gestures," *Pattern Recognition*, vol. 36, no. 9, pp. 2069 – 2081, 2003.
- [52] J. Nielsen, "The usability engineering life cycle," *Computer*, vol. 25, pp. 12–22, Mar. 1992.

- [53] J. Cassell, "A framework for gesture generation and interpretation," in *Computer Vision in Human-Machine Interaction*, pp. 191–215, Cambridge University Press, Cambridge, UK, 1998.
- [54] L. E. Potter, J. Araullo, and L. Carter, "The leap motion controller: A view on sign language," in *Proceedings of the 25th Australian Computer-Human Interaction Conference: Augmentation, Application, Innovation, Collaboration, OzCHI '13*, Adelaide, Australia, pp. 175–178, 2013.
- [55] S. Zhai, P. Kristensson, C. Appert, T. Andersen, and X. Cao, "Foundational issues in touch-surface stroke gesture design: An integrative review," *Found. Trends Hum.-Comput. Interact.*, vol. 5, pp. 97–205, Feb. 2012.
- [56] H. S. Badi and S. Hussein, "Hand posture and gesture recognition technology," *Neural Computing and Applications*, vol. 25, no. 3, pp. 871–878, 2014.
- [57] H. Hasan and S. Abdul-Kareem, "Static hand gesture recognition using neural networks," *Artificial Intelligence Review*, vol. 41, no. 2, pp. 147–181, 2014.
- [58] P. Trigueiros, F. Ribeiro, and L. P. Reis, "A comparison of machine learning algorithms applied to hand gesture recognition," in *7th Iberian Conference on Information Systems and Technologies (CISTI)*, pp. 1–6, 2012.
- [59] R. A. Bolt, "Put-that-there: Voice and gesture at the graphics interface," in *Proceedings of the 7th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '80, Seattle, Washington, USA, pp. 262–270, 1980.

- [60] J. Underkoffler and H. Ishii, "Illuminating light: A casual optics workbench," in *CHI '99 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '99, Pittsburgh, Pennsylvania, pp. 5–6, 1999.
- [61] D. Howe, "The free on-line dictionary of computing," in <http://www.dictionary.com/browse/gorilla-arm>, 2017. [Online; accessed March 28, 2017].
- [62] T. Mathew, "Manova in the multivariate components of variance model," *Journal of Multivariate Analysis*, vol. 29, no. 1, pp. 30 – 38, 1989.
- [63] J. Twisk, *Applied Longitudinal Data Analysis for Epidemiology: A Practical Guide*. Cambridge University Press, Cambridge, UK, 2003.
- [64] P. McCullagh and J. Nelder, "An outline of generalized linear models," in *Generalized Linear Models*, pp. 21–47, Springer, 1989.
- [65] D. A. Harrison and C. L. Hulin, "Investigations of absenteeism: Using event history models to study the absence-taking process.," *Journal of Applied Psychology*, vol. 74, no. 2, p. 300, 1989.
- [66] G. A. Ballinger, "Using generalized estimating equations for longitudinal data analysis," *Organizational Research Methods*, vol. 7, no. 2, pp. 127–150, 2004.
- [67] I. Granic, A. Lobel, and R. C. M. E. Engels, "The benefits of playing video games," *Journal of American Psychologist*, vol. 69, no. 1, pp. 66 – 78, 2013.

- [68] E. S. A. (2015), “Essential facts about the computer and video game industry,” Tech. Rep., Entertainment Software Association, 2015.
- [69] B. Laugwitz, T. Held, and M. Schrepp, *Construction and Evaluation of a User Experience Questionnaire*, pp. 63–76. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.