Hand Pose Estimation and Prediction for Virtual Reality Applications

Se-Joon Chung

CMU-CS-18-119

September 2018

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

This research was sponsored by the Asian's Student's Association Presidential Fellowship, NSF awards under grant numbers IIS-1218182, IIS-1344361, and IIS-1637853, and NVIDIA corporation with the donation of a Titan X Pascal GPU.

Keywords: hand pose estimation, hand pose prediction, virtual reality, augmented reality

To my wonderful parents and lovely sister.

Abstract

Use of hands is the primary way we interact with the world around us. Recent trends in virtual reality (VR) also reflect the importance of interaction with hands. Mainstream virtual reality headsets such as Oculus Rift, HTC Vive, and the Playstation VR all support and encourage the use of their hand tracking controllers.

However, tracking hands is very challenging due to their small size and various occlusions. For example, significant portions of the hand can get occluded when people are holding hands together or holding an object. For this reason, makers of VR headsets let their users hold controllers that are more reliably tracked than tracking hands directly.

Another problem of hand tracking is that it often adds latency to the system. Furthermore, networked multiplayer interactions are even more challenging to deliver without users noticing delays due to the addition of network delays.

In this thesis, we propose ways to overcome the current limitations of hand use in VR by addressing these challenges. To address difficulty of hand tracking, we present a way to estimate the entire hand pose given a few reliably tracked points on the hand. We used a commonly available multi-touch tablet to track the fingertip positions and estimated the entire hand pose from the tracked fingertip positions using a quadratic encoding method. We show that quadratic encoding method yields smooth motions for smooth changes in the fingertip positions and show some demos of manipulation tasks using the interface. We also present a way to handle changing number of fingertip contacts and show that we can identify unknown fingertips.

To address the latency in hand tracking and multiplayer interactions, we propose a method to augment hand pose prediction with eye tracking which will be commonly available in the next generation of VR headsets. We will first motivate our approach by presenting our observations of hand-eye coordination. We show that gaze leads grasping actions by an average of 0.726s, leaves the manipulation area as soon as the action is complete, fixates on the tool tips when a tool is held, and sometimes inspects the object without the hand directly interacting with the inspected area. All of the observations will be used to determine how gaze should be used to predict hand pose during different actions.

Then, we present a study on predicting grasp types to show that gaze is effective in predicting hand interactions. We found interesting patterns of gaze during bottle grasps before the hand reaches the bottle. We use neural networks to show that these gaze patterns can be learned in order to improve prediction accuracy.

Finally, we conclude with application of grasp type prediction in VR and a user study which evaluates the usefulness and quality of hand pose generated based on grasp type prediction. We found similar gaze patterns in VR as in the real-life experiment. The user study shows potential for usefulness of grasp type prediction in VR applications with rooms to improve in the future.

Acknowledgments

I could not be more thankful to Nancy Pollard for being the best advisor I could have ever asked for. Throughout the six years of my PhD, she has been a constant source of encouragement and inspiration. She has truly extraordinary virtues of being positive and supportive: she never showed a single moment of negativity even after harsh reviewer comments or numerous paper rejections. Furthermore, with patience and through engaging discussions, she has taught me how to see the bigger frame of things and ask questions like a true scholar. Even if I were to do my PhD again, I do not think I would want to ask for a different advisor.

I would also like to thank my thesis committee members Kris Kitani, Katerina Fragkiadaki, and Carol O'Sullivan. Through their expertise and interest in my work, they gave valuable insights and discussions that led to enhancements in my dissertation. Kris provided honest feedback and critical questions that led to a more solid ground in my work. Katerina advised me at various times with her expertise in machine learning methods, and Carol helped me think about future applications of my work.

I am fortunate to have worked with amazing collaborators with whom I have shared enriching discussions. In the approximate order that I have collaborated with, they are Junggon Kim, Shangchen Han, Maria Khutoretsky, and Seungmin Hwang.

Of course, I cannot forget to mention my gratitude to Justin Macey who runs the Motion Capture Lab at Carnegie Mellon University. Most of the work presented in this dissertation could not have been done without his help, as my work heavily relies on capturing and analyzing real human hand movements.

My colleagues at Nancy's "All Hands" group and Carnegie Mellon University's Graphics Lab have helped me with insightful discussions and helpful feedback on various talks. I would like to give special thanks to Yuzuko Nakamura, Michael Koval, and Jim McCann for being there for me even during last minute preparations.

I am also thankful to the administrative staff at Carnegie Mellon University who are always working around the clock to make everything run smoothly. I am especially thankful to Deb Cavlovich and Jessica Butterbaugh for helping me out on countless occasions.

Last but not least, I would like to thank my friends and family for being supportive throughout my PhD. I would like to especially acknowledge my 5 year housemate Gihyuk Ko who was there through all the ups and downs of my PhD life. It was truly fortunate to have lived with a such wonderful friend who was always there for me as a dining companion, a drinking buddy, a board game pal, and at times, an unsuspecting test subject. I am equally thankful to all my other friends who have shared company with me during my PhD. There are too many names to list all here, but just to name a few of my Pittsburgh friends, I would like to thank Euiwoong Lee, Jun Woo Park, Sunhee Baik, Christine Baek, Jinoh Oh, Kijung Shin, Youngwook Do, Joshua Hanjoon Kim, Dokyun Lee, Michelle Tsang, Jisu Kim, and Joohyung Kim. Some friends have supported me through their remote presence, and these friends include Hyun Jung, Henry Hyunsik Moon, Seungho Yang, Hyung Seok Choi, Seongmin Kim, Jinhyung Park, and Alex Shin.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

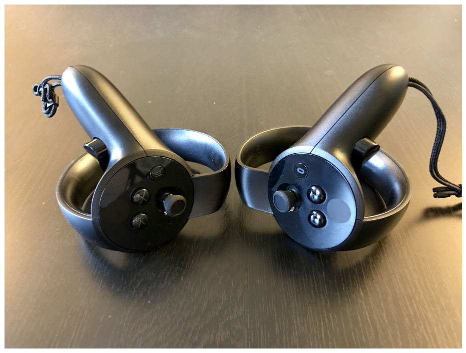
Use of hands is the primary way we interact with the world around us. We use our hands every day to pick up objects from the table or open various containers. We use our hands to eat, to write, or to turn on the light switch. We can even accomplish more complex tasks such as assembling furniture which requires the use of various tools and coordination between both hands to hold the pieces together.

And perhaps the most fascinating part of our use of hands is that we pay little attention to how we actually control our hands. Most of the time, we are thinking about what we want to do with the target object after we acquire it, not how we want to bend our fingers in what order to pick it up. Use of hands is just a natural way for us to interact with this world.

On the other hand, the use of hands in personal computing devices such as desktop computers or smartphones has been somewhat different due to the fact that we are interacting with a 2D virtual world in a screen. Instead of freely using our hands to grasp 3D objects directly like we would in reality, we rely on tools such as a mouse or a touchscreen. These tools are excellent for interacting with a 2D environment, because they provide reliable and precise tracking of 2D movements on a surface.

However, with the advent of consumer level virtual reality (VR) technology, the time has come for us to consider a suitable interface to interact with a 3D virtual world. Given our observation of how natural it is for us to use our hands to interact with real-life 3D objects, it is only logical that we let users interact with VR using their own hands.

Indeed, this is precisely what the industry is doing. Mainstream virtual reality headsets such as Oculus Rift, HTC Vive, and the Playstation VR all support and encourage the use of their hand tracking controllers (figure 1.1). The hand controllers enable users to grasp


(a) Oculus Touch for Oculus Rift VR system.

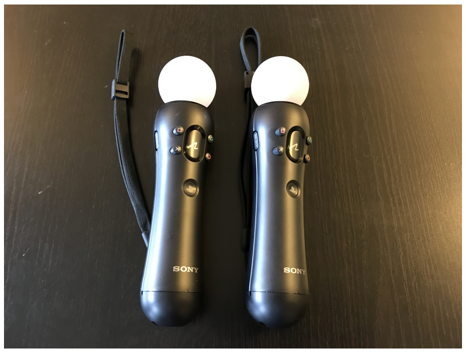
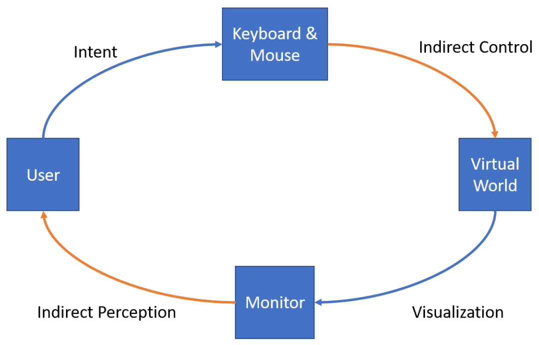

(b) Playstation Move controller for Playstation VR.

Figure 1.1: Examples of VR hand tracking controllers.

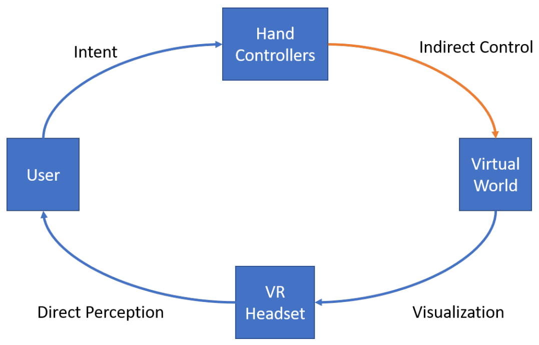
various objects in sandbox games like *Toybox*, shoot guns in *The London Heist*, and cast spells with gestures in *The Unspoken*. Nonetheless, there are still some challenges for use of hands in VR.

The first challenge is that because it is difficult to track hands directly, hand controllers have to be used instead of using our hands directly. And the problem with controllers used in popular VR systems is that they do not provide us with all the freedoms of our hand movement. As tools that are held in our hands, the controllers track gross position and orientation of the hand and provide us with some buttons to trigger pre-programmed actions. The current generation of VR systems take this approach, because the tiny size of our hands relative to our body, combined with the numerous degrees of freedom in our hands make hand tracking from outside very challenging. The problem is further exacerbated by occlusions: either by hand parts blocking out itself or by objects that are held within the hand. If we can overcome these challenges and allow users to utilize all degrees of freedoms of our hand movement, new VR interaction modes will be possible such as solving an intricate puzzle box in *I Expect You to Die* or adding spins to basketball shoots in *VR Sports Challenge*.

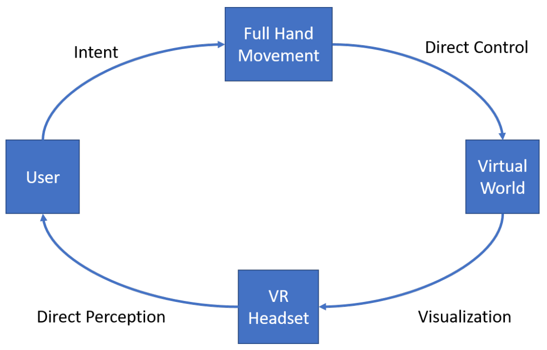
Another critical challenge in use of hands in VR is the tracking delay. VR headsets already adopt prediction for their head tracking in order to reduce the delay between measurement of user's actual head pose and rendering of virtual scenes from that head pose (motion-to-photon delay). Without reducing the motion-to-photon delay, users are certain to experience motion sickness from the mismatch between perceived head pose from proprioception and head pose from rendered perspective. However, a similar method for reducing hand tracking delay is more challenging, because hand movements are less con-



(a) Control and perception for personal computers.



(b) Control and perception for current VR systems.



(c) Control and perception for an ideal VR system.

Figure 1.2: Control and perception for current and ideal systems.

strained than the head movements. Although delays in hand tracking will not likely cause motion sickness, delays will cause difficulty during fast-paced activities that require fine motor control such as a VR tennis game. Furthermore, when we are interacting with other people over the network (which is likely a very essential use of VR due to its removal of physical barriers), we will experience delays in other people's actions. This will make it difficult for people to collaborate or play against each other in activities that require the use of hands.

Last but not least, all of the above problems need to be addressed in real-time, because VR systems are interactive. Not only does this mean that our methods have to be computationally efficient, this also means that we do not have the ability to fix our outputs retrospectively.

We summarize the flow of control and perception for various systems in figure 1.2.

For traditional personal computers, we have both indirect control through input devices and indirect perception through a flat screen (figure 1.2a). Current VR systems make a step forward by allowing users to perceive the virtual world in a direct and natural manner through the use of VR headsets (figure 1.2b). Still, users of the current VR systems have indirect control through the use of hand controllers. We propose a step towards the idealized VR system where all control and perception are direct (figure 1.2c) by overcoming the aforementioned challenges.

In order to overcome these challenges, we build upon some insights of hand motion and propose a framework for estimating and predicting hand pose in real-time.

The first insight is that hand movement is usually coordinated, and although we have many joints in the hand, due to the anatomy of muscles in the hand, many joint movements are correlated. This insight lets us simplify the hand pose problem as a set of simple quadratic functions which successfully estimate the entire hand pose by tracking the fingertip positions. In section 4, we propose a method to estimate the 3D hand pose from 2D fingertip positions given by a commonplace multi-touch tablet.

The second insight is that for many activities where hand-eye coordination is required, the eye gaze leads hand movement by a significant amount of time. This observation allows us to not only exploit the position of the gaze, but also the timing in order to develop a new prediction model. In section 6, we propose a method to predict the hand pose by tracking user's hand and eye movements. Then, we apply our findings to a VR experiment (chapter 7) and evaluate user experience in a user study (chapter 8).

This thesis presents a framework that addresses challenges in use of hands in VR. Our contributions are a two-part framework which may be used independently of each other:

- The *hand pose estimation* framework shows that a simple model can significantly ease the problem of hand tracking. As long as the fingertips are reliably tracked, we can estimate the entire hand pose.
- The *hand pose prediction* framework shows that using gaze improves early grasp type prediction accuracy, and this early prediction can be used to generate hand pose.

We hope our framework sheds light on the nature of hand motion and show that other signals such as gaze can be helpful in improving hand pose prediction.

Chapter 2

Contributions

2.1 Contributions of the Hand Pose Estimation Framework

Our hand pose estimation framework can estimate the entire hand pose from fingertip positions tracked by a multi-touch device in real-time. The contributions of our method are:

- We give the first example of hand pose estimation from multi-touch input.
- We introduce the quadratic encoding technique for smooth hand pose estimation from fingertip positions.
- We demonstrate that changing contacts can be handled gracefully by using the same quadratic function for all contact conditions.
- We present an algorithm for estimating which fingers are responsible for which contact on a multi-touch input device.
- We demonstrate that quadratic encoding can represent joint angle values successfully even for hand motion in manipulation tasks.

2.2 Contributions of the Hand Pose Prediction Framework

Our hand pose prediction framework can predict grasp type early by tracking user's current hand pose and gaze. Then, it outputs an appropriate hand pose based on the predicted grasp type. The contributions of our method are:

- We show that gaze information improves grasp type prediction, especially among similar grasps at different locations.
- We show that gaze information improves grasp type prediction not only in real-life, but also in virtual reality.
- We show that it is possible to generate hand pose based on the predicted grasp type.
- We show in a user study that users quickly adapt to the hand pose prediction framework, and they are faster at grasping with the hand pose prediction framework.

Chapter 3

Related Work

This work could not have existed without building upon the knowledge base of numerous related works. In this section, we summarize a selection of related works by dividing them up with regards to their relevance to hand pose estimation (section 3.1) or hand pose prediction (section 3.2), then by grouping them into similar subtopics.

3.1 Hand Pose Estimation

Hand pose estimation is the problem of estimating the current pose of the hand given some set of features related to the hand in the current time step. Here, we outline the works related to our hand pose estimation approach.

3.1.1 Animating Hands

The difficulty of animating hands is a well recognized problem and there has been a great deal of research directed towards this problem. Animation of hands has been approached from the perspectives of modeling hand geometry [Gourret et al., 1989, Kry et al., 2002, Albrecht et al., 2003, Huang et al., 2011, Garre et al., 2011], modeling hand anatomy and muscle forces that drive hand motion [Albrecht et al., 2003, Tsang et al., 2005, Sueda et al., 2008], modeling the process of grasping [Rijpkema and Girard, 1991, Pollard and Zordan, 2005, Zhao et al., 2013], understanding the dynamics of contact and modeling grasp contact forces [Kry and Pai, 2006, Kry et al., 2008, Bai and Liu, 2014], and creating manipulation motions from task goals and constraints through optimization [Liu, 2008, 2009, Mordatch et al., 2012, Kumar et al., 2014, Bai et al., 2016]. Although our work does produce animated hands, the aim is different in that we are trying to estimate user's hand pose in real-time using a commonplace input device rather than producing a realistic animation.

3.1.2 Hand Motion Capture

Diverse methods used for hand motion capture are well outlined in Sturman and Zeltzer [Sturman and Zeltzer, 1994]. In this area, Majkowska et al. [Majkowska et al., 2006] recognizes the difficulty of capturing full-body motion and hand movement at the same time, and proposes a method for combining the two motion captures after the fact. Some people choose to use specialized glove hardware to capture the hand motion as outlined in Zimmerman et al. [Zimmerman et al., 1987]. Vision based approaches to hand motion capture have been developed by a number of researchers with the primary goal of scene or action understanding (e.g., [Oikonomidis et al., 2011a, Ballan et al., 2012]) or in the context of robotics applications such as learning from imitation (e.g., [Romero et al., 2010, Kjellstrom et al., 2008]). Wang and Popović [Wang and Popović, 2009] demonstrated compelling results for hand tracking to the graphics community using a single camera and a color glove. More recent works in the area adopt the use of depth cameras either in combination with marker-based motion capture [Zhao et al., 2012] or in completely markerless settings [Oikonomidis et al., 2011b, Wang et al., 2011, Keskin et al., 2012, Zhao et al., 2013, Xu and Cheng, 2013, Tompson et al., 2014, Sharp et al., 2015, Sun et al., 2015, Ge et al., 2016, Simon et al., 2017]. However, to our knowledge, there has been no previous research to extract hand pose from multi-touch inputs, even though multi-touch devices are widely available and inexpensive, as well as providing precise fingertip positions and giving the user some sense of tactile feedback.

3.1.3 Estimation from Reduced Dimensional Input Data

Some researchers have also focused on estimation from reduced dimensional input data. For example, Chai and Hodgins used a reduced set of markers to match motions in their database captured with full set of markers [Chai and Hodgins, 2005]; Liu and his colleagues also explore estimating motions from a reduced marker set [Liu et al., 2006]; Yin and Pai and Bränzel and his colleagues estimate full body motions from pressure-sensing floor [Yin and Pai, 2003, Bränzel et al., 2013], and Slyper and Hodgins use five accelerometers to match the acceleration information in the database [Slyper and Hodgins, 2008]. Some even suggest how to effectively choose the reduced marker set and estimate hand motions from these markers [Kang et al., 2012, Wheatland et al., 2013, Schröder et al., 2015]. Many of thes approaches use techniques related to nearest neighbor or local models. We show that in our case, nearest neighbor approaches are prone to generating discontinuities in pose, whereas our quadratic encoding approach produces smooth results.

For hands in particular, El Koura and his colleagues estimate hand motions specifically for guitar playing, using a motion capture database to capture sympathetic motions of the fingers [ElKoura and Singh, 2003]. Hamer and colleagues [Hamer et al., 2011] estimate hand motion from object motion by retrieving acceptable hand motions from a captured database. Ye and colleagues [Ye and Liu, 2012] estimate plausible hand motion from motion capture of the full body up to and including the wrist along with motion capture of the manipulated object. Mulatto et al. perform an inverse kinematics approach based on thumb and index finger positions measured by a haptic device, taking into account a set of linear dependency between the joint angles that they call synergies [Mulatto et al., 2013]. Hoyet and colleagues [Hoyet et al., 2012] provide evidence from human subjects experiments for the perceptual validity of estimating hand motion from reduced marker sets. Chang and her colleagues [Chang et al., 2007] explore minimal marker sets for grasp discrimination.

3.1.4 Quadratic Encoding

Quadratic encoding has been used previously to encode energy values and center of mass trajectories for different foot placements in humanoid robot walking [Kim et al., 2013]. However, to our knowledge there has been no attempt to encode joint angles in this manner as we do in our work.

3.2 Hand Pose Prediction

Hand pose prediction is the problem of predicting the future pose of the hand given some set of features related to the hand in the current timestep. Here, we outline the works related to our hand pose prediction approach.

3.2.1 Biological Research on Gaze

Biological studies show that gaze fixations allows us to predict spatial goals and timing of actions. For example, a study by Johansson et al. shows that subjects almost exclusively fixate on landmarks critical for object manipulation [Johansson et al., 2001]. Another study by Hayhoe et al. suggests that location of fixation is highly dependent on the intended task. They found subjects fixate the middle of the jar for grasping and the rim for putting on the lid [Hayhoe et al., 2003]. This agrees with findings by Land that eye movement strategies are very task-specific [Land, 2006]. The survey by Flanagan et al. summarizes that gaze fixations not only predict the spatial goals of sequential actions, but also the timing of the actions [Flanagan et al., 2006]. The timings seem to vary by task. Studies found that gaze leads grasping actions by about 1s during a bar moving task [Johansson et al., 2001] and by about 0.56s during a tea making task [Land et al., 1999]. For moving objects, Bulloch et al. found their subjects to fixate on the leading edge in the direction of the target's movement, but eventually converge their gaze towards the final index finger contact point on the target [Bulloch et al., 2015]. These findings suggest that gaze can give strong cues about our intended manipulation before we start performing them. Our decision to include gaze as one of inputs to our prediction system was inspired from these studies.

3.2.2 Gaze in Computer Animation

Biological research on importance of gaze has strongly influenced computer animation research. It has been studied that gaze Earlier works have included gaze as a post-processing step to increase realism in the animation [Yamane et al., 2004, Tsang et al., 2005]. Later on, gaze has been used as a central part of coordinating character movements as biological studies suggested. Yeo et al. were able to generate convincing animations for ball catching tasks where the ball trajectory observed by the eyes guide the body movements [Yeo et al., 2012]. Duchowski et al. go even further into modeling the gaze jitter present in natural eye motions to make animated gaze more realistic [Duchowski et al., 2016]. Our work differs in that we do not control the movement of eye or body to produce a convincing animation, but predict in real-time what the human user will be doing in the near future given the user's gaze.

3.2.3 Prediction of Hand Motion Without Eye Tracking

There have been several studies on predicting hand motion without the use of eye tracking. Fan et al. have studied forecasting future locations of human hands and objects [Fan et al., 2017]. Pérez-D'Arpino and Shah use time series classification to predict targets of human reaching motion with 70% prediction accuracy after observing 400ms of hand trajectory [Pérez-D'Arpino and Shah, 2016]. Chan et al. were able to identify finger motion constraints and proposed a constraint-based motion prediction method for hand motions [Chan et al., 2008]. Rather than predicting the full hand motion directly, we focus on grasp type prediction and show eye gaze can improve grasp type prediction accuracy.

3.2.4 Classification and Prediction of Discrete Actions Without Eye Tracking

Some have focused on classifying or predicting function of the hand, also without the use of eye tracking. Rogez et al. have studied classifying grasp types from RGB-D images without eye tracking [Rogez et al., 2015]. They were able to achieve 20% accuracy on unseen subjects and objects across 71 grasps collected from 8 subjects. Our combination of hand images for grasp type prediction in chapter 6 was inspired by their work. Fermüller et al. focuses on predicting manipulation actions such as drinking or pouring from a cup before the hand reaches the objects [Fermüller et al., 2016]. They show that it is possible to predict the type of manipulation well before the hand reaches the object, albeit with less certainty. Our work confirms their findings and further shows inclusion of gaze information improves prediction accuracy.

3.2.5 Human Action Detection Using Gaze

Eye tracking has been used for detection of actions such as various steps of making sandwiches or moving an object to specified landmarks. Some used implicit gaze signals from egocentric cameras [Ma et al., 2016, Matsuo et al., 2014, Bertasius et al., 2017] and others used explicit gaze signals captured by a dedicated tracker [Fathi et al., 2012, Li et al., 2015, Pinpin et al., 2016]. All of these works show that our actions are closely coupled with our attention which can be inferred from our gaze.

3.2.6 Action Prediction in Human-Robot Interaction

In robotics, human motion prediction as a part of human-robot interaction exist in many different works. Foka and Trahanias have modeled prediction of human walking destinations to aid in autonomous robot navigation [Foka and Trahanias, 2010]. Broz et al. further generalizes their approach to model more general notion of human intention and time-dependent action. They use this human intention model to determine a human driver's intention in a Pittsburgh left turn simulator [Broz et al., 2013]. Jain et al. explore how a human driver's action among a group of possible actions can be anticipated using recurrent neural networks [Jain et al., 2015]. Other work utilizing user intent inference for teleoperation also exist [Yu et al., 2005, Dragan and Srinivasa, 2013, Hauser, 2013]. Koppula and Saxena anticipate human activities by considering object affordances [Koppula and Saxena, 2016]. Our work tries to go beyond predicting the action classification and includes predictions of the hand pose as well.

3.2.7 Gaze as an Interface

Because gaze is closely coupled with our intended actions, there have been many efforts in using gaze as an interface for interacting with virtual environments. The study by Tanriverdi and Jacob shows that eye movement-based interaction was faster than pointing, especially for distant objects [Tanriverdi and Jacob, 2000]. Pinpin et al. studied utilizing gaze as an interface for controlling a future hybrid bionic system [Pinpin et al., 2008]. By detecting gaze fixations at obligatory landmark areas near object contact points, they showed that they can predict the target contact points and trigger appropriate robot movement commands. More recently, Pfeuffer et al. have studied using gaze in conjunction with hand based gestures for interacting with virtual targets [Pfeuffer et al., 2017]. Because of the usefulness of gaze, we believe eye tracking will be included in future generation virtual reality headsets.

3.2.8 Gaze Prediction

Our method heavily relies on gaze data, but this data can be predicted from previous motion data without using a gaze tracker. Studies show that gaze can be predicted using various cues in one's actions [Ou et al., 2008, Li et al., 2013].

3.2.9 Estimation from Reduced Dimensional Input Data

Estimation of human motion from as outlined in section 3.1.3 has some relevance to hand pose prediction in the sense that they give understandings of which features of the motion are important. However, the fact that we are trying to predict future motions of the hand distinguishes this work from theirs.

3.2.10 Understanding Hand Grasps

Predicting hand motions cannot be done without understanding the motions first. Cai et al. show that machines can learn to distinguish grasps based on their appearances [Cai et al., 2015], and the performance is further improved when object attributes are considered [Cai et al., 2016]. In a different context, we also show that machines can learn to distinguish grasps, and this result is improved when gaze information is included.

3.2.11 Human Motion Generation

Various works have explored using probabilistic models trained on motion capture data for generation of stylized human motion. Models such as bilinear spatiotemporal basis models [Akhter et al., 2012], Hidden Markov Models [Brand and Hertzmann, 2000], linear dynamical systems [Pavlovic et al., 2001], Gaussian process latent variable models [Wang et al., 2008, Urtasun et al., 2008] and its multilinear variants [Hsu et al., 2005, Wang et al., 2007], and dynamical models based on Restricted Boltzmann Machines [Taylor et al., 2007, Sutskever et al., 2009, Taylor and Hinton, 2009, Taylor et al., 2010] have been used to model full body human motions. These models were then used to either predict missing motion data by filling them in or through extrapolation, or to improve performance in human motion tracking through predictions. A more recent work by Fragkiadaki et al. showed that recurrent autoencoder networks [Fragkiadaki et al., 2015] can outperform many of the above mentioned methods and produces stable prediction results over long periods of time. We also use recurrent neural networks which are good at learning complex sequence of gaze patterns.

3.2.12 Learning Control Policies

Some groups have explored learning the underlying control policies rather than generating the motions directly. Tan and colleagues have proposed a method to search for both the parameterization and the parameters of a policy and demonstrated the use in various bicycle stunt animations [Tan et al., 2014]. Others sought to learn the control policies from example motions by using random walks [Liu et al., 2016] or by using inverse optimal control [Mainprice et al., 2015, 2016]. Our work generates hand pose from predicted grasp types which consider the timing and history of the motion through the use of long short-term memory units (LSTMs).

Chapter 4

Hand Pose Estimation from Fingertip Positions

In order to get us a step towards direct manipulation in VR, one of the biggest challenges we must overcome is the difficulty of tracking hands. Because of the small size of hands relative to our body along with their susceptibility to various occlusions, many VR systems provide a set of hand controllers instead of directly tracking the hands. However, these controllers do not provide with all the freedoms available in our natural hand movement.

One way to overcome the difficulty in tracking hands is to track a few reliably tracked features of the hands and to estimate the entire hand pose from these features. In order to make a step towards being able to use our hands directly in VR environments, we propose a method to estimate the entire hand pose from fingertip positions which are easily tracked using a multi-touch device.

We used a multi-touch device to track the fingertip positions, because they are widely available and inexpensive, they deliver precise fingertip position information, and they provide some feeling of touch to the user and support for the hand, features which may be important for successful virtual manipulation. However, in order for our estimation method to be practical, we must be able to estimate complete hand poses from fingertip positions in real-time. Furthermore, the result should be simultaneously smooth, controlled, precise, and natural.

We present a quadratic encoding approach that allows us to create smooth, natural hand motion in real-time from multi-touch fingertip inputs. A quadratic function is used to evaluate each joint angle in the hand model from the captured fingertip positions and the contact status. Thus, we need N different quadratic functions to restore the posture of an N-dof hand model. A relationship between the fingertip state and the full hand configuration is learned from a hand motion database captured using a conventional marker-based system and encoded offline as quadratic functions for use in online pose estimation.

We do not explicitly assume a reduced dimensional configuration space which is often used to improve processing speed but can limit the performance in representing diverse hand postures. However, representing hand postures as quadratic functions of multi-touch inputs reduces the dimensionality of the stored space of natural hand motions in a different manner, as only coefficients of our quadratic functions need to be stored. We show that our quadratic encoding approach can outperform PCA based inverse kinematics in both quality of results and computation time. While our quadratic encoding approach produces high quality results in real-time, PCA based inverse kinematics is considerably slower.

There are three main benefits to using quadratic encoding for online hand motion estimation:

- Quadratic encoding produces smooth pose transitions by providing similar outputs for similar inputs.
- Quadratic functions are trivial to compute, leading to fast estimations in real-time.
- Hand poses with different fingers contacting the multi-touch device can be handled in a seamless manner, using the same encoding functions.

This section details our quadratic encoding approach, including how we handle changing finger contacts on a multi-touch surface. Our results show that quadratic encoding is superior to linear encoding, cubic encoding, nearest neighbor, and PCA based inverse kinematics. We demonstrate the ability to simulate a hand manipulating objects in a virtual environment in real-time using a multi-touch device as input.

The contributions of our method are:

- We give the first example of hand pose estimation from multi-touch input.
- We introduce the quadratic encoding technique for smooth hand pose estimation from fingertip positions.
- We demonstrate that (perhaps surprisingly) changing contacts can be handled gracefully by using the same quadratic function for all contact conditions.
- We present an algorithm for estimating which fingers are responsible for which contact on a multi-touch input device.

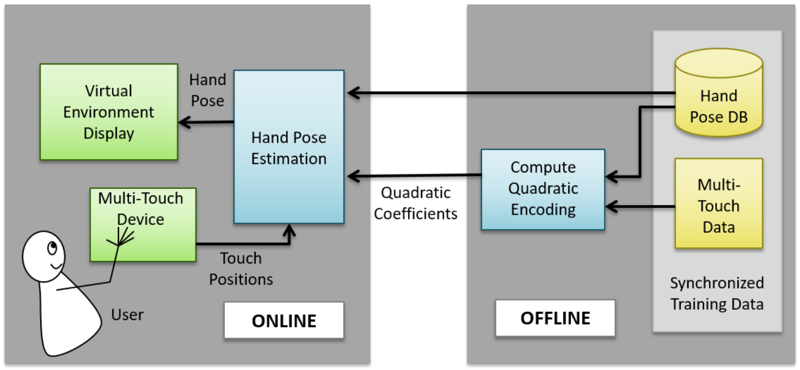


Figure 4.1: System flow. Quadratic encoding coefficients are computed offline from training data and used online for hand pose estimation from multi-touch input.

- We demonstrate that quadratic encoding can represent joint angle values successfully even for hand motion in manipulation tasks.

4.1 System Overview

An overview of our system is shown in figure 4.1. The problem we address here is estimation of the user's hand pose given the location of the user's fingertips. Our system's input is a set of 2-dimensional fingertip positions from an iPad® 2. There may be anywhere from one to five fingertip positions representing one to five fingers in contact with the multi-touch device. Notably, the fingers are not identified, so we do not know at this stage which finger is making contact at which position. All fingertip positions are supplied to a pose estimation step, which has access to (1) a set of quadratic coefficients which have been computed offline and (2) a database of hand poses. The output of pose estimation is hand position, hand orientation, and joint angles. The estimated hand pose is displayed to the user to allow hand motion and object manipulation within a virtual environment in real-time.

The process that supports hand pose estimation takes as input a synchronized training database of hand poses and multi-touch data. This training data is preprocessed to label the fingers in contact, and a quadratic function is learned from the labeled data for each of the hand pose degrees of freedom. Data processing and computation of the quadratic function coefficients are performed offline.

In the next sections, we first describe our quadratic encoding function, followed by a

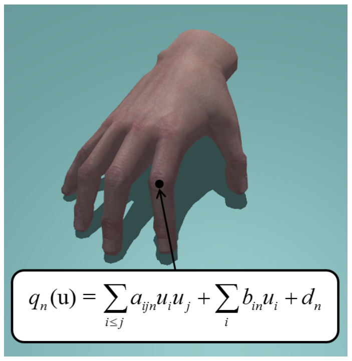

$q_n(\mathrm{u}) = \sum_{i\le j} a_{ijn}u_iu_j + \sum_i b_{in}u_i + d_n$

Figure 4.2: Each degree of freedom has its own quadratic encoding function.

discussion of offline training and online pose estimation.

4.2 Quadratic Encoding

Given a database of hand postures and contact information from a multi-touch device, we aim to represent each degree-of-freedom of the hand as a quadratic function of the contact variables. Specifically, let our input be vector $u \in \Re^{15}$ as follows:

$$u = (p_1, \cdots, p_5, s_1, \cdots, s_5) \tag{4.1}$$

where $p_i = (x_i, y_i) \in \Re^2$ denotes the position of the i-th fingertip on the multi-touch device, s_i denotes contact status whose value is 1 if the fingertip is in contact and 0 otherwise, and the subscripts represent the indices of the fingertips, i.e., i = 1 (thumb), 2 (index), $\cdots$, 5 (little).

Then the n-th degree-of-freedom of the hand can be expressed as a quadratic function of u as follows:

$$q_n(u) = \sum_{i\le j} a_{ijn}u_iu_j + \sum_i b_{in}u_i + d_n \tag{4.2}$$

where $a_{ijn}(i \le j)$, b_{in} and d_n are the coefficients that must be determined. A total of N quadratic functions are needed to estimate the full configuration of an N-degree-of-freedom hand model (figure 4.2). Because the u vector is 15-dimensional, we must solve for $120 + 15 + 1 = 136$ coefficients per degree-of-freedom.

For our implementation, we chose the fingertip position (x_i, y_i) to be given in inches from the top left corner of the screen, and it is set as $(-1, -1)$ if the finger is not in contact

($s_i = 0$). We chose to work in physical dimensions for better portability across different multi-touch devices. The value of $(-1, -1)$ was chosen to give some separation in the 15-dimensional input space when the fingers are not in contact. We experimented with numbers of higher magnitude, but found that $(-1, -1)$ achieved the best results. The multi-touch device orientation is fixed and users are assumed to be sitting directly in front of the device.

Our use of quadratic encoding is divided into two phases: offline training phase and online pose estimation phase. During the offline training phase, we compute the coefficients of quadratic functions using a hand pose database. During the online pose estimation phase, we estimate the hand pose based on the given fingertip positions. Details within these phases are given below.

4.2.1 Offline Training

Given a training set of contact inputs u corresponding to hand postures $q_n(u)$, we solve for coefficients a_{ijn}, b_{in}, and d_n using an off the shelf least squares solver. Specifically, in our experiments we use MATLAB's lsqcurvefit with max iteration of 1000, and maximum number of function evaluations of $10000 *$ numberOfVariables. In our examples, we solve for a total of 136 coefficients for each of 48 degrees-of-freedom, which were computed from a training set of 7,083 input hand postures containing a wide variety of finger contact combinations. Details of our training and test datasets are given in Section 4.4.

4.2.2 Online Pose Estimation

Pose estimation at runtime consists of four separate parts. First, we preprocess the data for translation invariance, so that we may make more effective use of our existing database. Second, we label the fingers in contact so that a correctly formed input vector u can be created for computing the values of the quadratic functions in equation 4.2. Third, we use the resulting input vector u to compute $q_n(u)$ in equation 4.2 for each degree-of-freedom of the hand to estimate the hand pose. Fourth, if desired, we pass the estimated hand pose through a physics simulator. These four steps are detailed below.

(1) Enforce Translational Invariance. In order to remove the input redundancy (i.e., the same hand pose can appear anywhere over the multi-touch device), all fingertip locations on the multi-touch device are measured relative to the leftmost finger contact point (instead of the multi-touch device origin). Once the hand pose is estimated by evaluating the encoding functions using the relative positions, the global location of the hand wrist is restored using the offset. Introducing this translational invariance to the encoding function not only improves the encoding performance but also greatly reduces the data sample size required to train the encoding functions.

(2) Label the fingers in contact. To label the fingers in contact with the multi-touch device, we use the straightforward mechanism of generate and test. To estimate which finger is responsible for which observed contact point, we check all possible combinations of the finger indices, estimate hand pose using our quadratic encoding for each combination, and then choose the estimated pose having a minimum Euclidean distance from the nearest pose in our hand pose database. To maintain a consistent user experience, this identification process is performed only when an unseen finger is introduced into contact, and the selected finger indices are kept in use until the next such change occurs.

(3) Estimate the Hand Pose. To create a hand pose using quadratic encoding at runtime, we evaluate the quadratic function in equation 4.2 for each degree of freedom of the hand model using the coefficients a_{ijn}, b_{in}, and d_n that have been computed offline.

(4) Apply a Physics Simulation. If physics is enabled (e.g., for object manipulation), we set the estimated pose as the desired hand pose for a physics simulation running in the background. In our implementation, we used Bullet for the physics engine in an out-of-the-box settings except for setting the maximum impulse values to $100N \cdot s$. In Bullet, velocities are corrected using impulses, and the maximum impulse value determines how aggressively the simulator will attempt to reach the target posture. Bullet handles all collision detection and response. We simply set the desired hand pose from our computed estimation at each time step.

4.3 Comparison Methods

In order to evaluate our system, we implemented a few other methods of hand pose estimation. In this section, we give a brief overview and details about our implementation of each of these methods.

4.3.1 Nearest Neighbor

A nearest neighbor method was implemented as a naïve base case test. A KD-tree datastructure was created from a set of vectors of fingertip positions recorded from the multi-touch device during training. Each vector $p = (p_1, \cdots, p_5)$ where $p_i = (x_i, y_i) \in \Re^2$, denotes the position of the i-th fingertip on the multi-touch device as in Section 4.2. We chose a value of $(-10000, -10000)$ for p_i if the finger is not in contact for our nearest neighbor implementation, so that close matches would be found only when the finger labeling was identical. Each p in the training database is associated with a pose q which corresponds to ground truth joint angles.

At run-time, we follow the same procedures for translational invariance and fingertip labeling as in the quadratic encoding method. For every multi-touch input, we identify the nearest neighbor based on Euclidean distance in the space of multi-touch fingertip positions (vector p). From all possible finger labeling, we select the one having the shortest distance between the corresponding vector p and that of its nearest neighbor.

4.3.2 Other Encoding Methods

Linear and cubic encoding methods were also implemented to test against quadratic encoding. They follow the same procedure as the quadratic method except their joint angles are defined as

$$q_n(u) = \sum_i a_{in} u_i + d_n \tag{4.3}$$

for linear encoding and

$$q_n(u) = \sum_{i \leq j \leq k} a_{ijkn} u_i u_j u_k + \sum_{i \leq j} b_{ijn} u_i u_j + \sum_i c_{in} u_i + d_n \tag{4.4}$$

for cubic encoding.

4.3.3 Principal Component Based Inverse Kinematics (PCB IK)

Inverse kinematics (IK) techniques have been explored for generating natural motion from sparse input, especially when end effector positions are known, as in our application. We compare our results vs. the principal component based inverse kinematics (PCB IK) method from [Mulatto et al., 2013], which was designed specifically for estimating hand poses. The main assumption behind this method is that there is a set of linear dependencies between joint variables of the hand. In particular, the pose of the hand is represented by

$$q_a(t) = Sz(t) + q_m \tag{4.5}$$

where $q_a(t)$ is the vector of joint angles at time t, S is the synergy matrix of principal components, $z(t)$ is the vector of synergy variables at time t, and q_m is the mean pose of training data. Using $q_a(t)$ from above and $q_v(t)$ which represents the translation in the root joint, we can compute the fingertip positions by direct kinematics:

$$\begin{aligned} x_a(t) &= \mathcal{K}\left(\begin{bmatrix} q_a(t) \\ q_v(t) \end{bmatrix}\right) \\ &= \mathcal{K}\left(\begin{bmatrix} Sz(t) + q_m \\ q_v(t) \end{bmatrix}\right) \end{aligned} \tag{4.6}$$

Our goal is to find $\hat{z}(t) \approx \begin{bmatrix} z(t) \\ q_v(t) \end{bmatrix}$ that approximates $z(t)$ and $q_v(t)$ at time t. Since we know the fingertip positions, we can use inverse kinematics to iteratively update $\hat{z}(t)$ towards a pose that achieves the desired fingertip positions.

At time t, we perform k such iterations. Starting with $\hat{z}(0) = \vec{0}$, we update $\hat{z}(k)$ as:

$$\hat{z}(k+1) = \hat{z}(k) + C_s \sigma(k) \tag{4.7}$$

where C_s is a user-defined compliance matrix and $\sigma(k)$ is defined as follows:

$$\sigma(k) = \hat{J}(\hat{z}(k))^T F(k) \tag{4.8}$$

where

$$\hat{J}(\hat{z}) = \begin{bmatrix} J_a(q_a, q_v)S & J_v(q_a, q_v) \end{bmatrix} \tag{4.9}$$

$$F(k) = x_h(t) - x_a(t) \tag{4.10}$$

J_a and J_v are Jacobian matrices corresponding to the hand pose and root joint translation respectively. $x_h(t)$ is the target fingertip positions at time t. For further details, we direct the reader to [Mulatto et al., 2013].

However, unlike the original work, fingertip labels are unknown. Therefore, we perform an exhaustive search over possible fingertip labellings similar to that of the quadratic encoding method except we use a different metric for determining the best labels. We found that Euclidean distance between fingertips yielded better results than Euclidean distance between pose vectors q for identifying a nearest neighbor that would return a plausible finger labeling, and so we use Euclidean distance between fingertips as our metric for determining finger labels. Upon some investigation, we observe that the IK technique does not estimate poses from the database especially faithfully, and so using the original

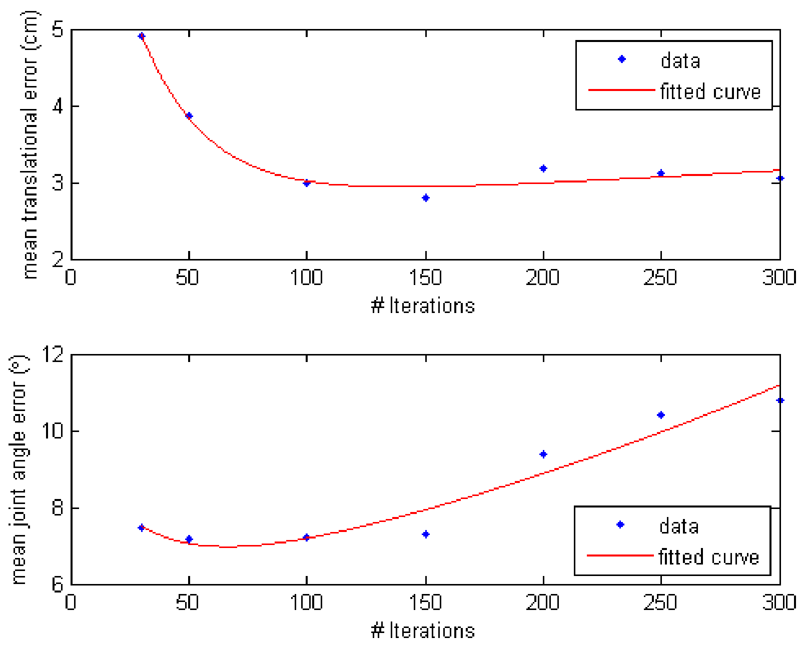


Figure 4.3: Estimation error of PCB IK method across number of iterations (top: translational error, bottom: joint angle error).

data – i.e., the fingertip positions – to find near matches in the database is more reliable than using the pose that the technique estimates.

The number of principal components that can be accommodated depends on the number of fingers in contact: more fingers in contact corresponds to a greater number of constraints that can be used to guide the PCB IK process. Each recorded fingertip position provides us with three constraints, because we use the (x, y) position from the multi-touch device and we can constrain the fingertip's z value to device height while fingertip is in contact with the device. As a result, we are able to use (number of fingers in contact)$*3-3$ principal components for computing joint angles (three constraints are subtracted because PCB IK also solves for root translations.)

With our training data, we found that for 5 fingers in contact, the available 12 principal components capture 97% variance. For 4 fingers in contact, the available 9 components yield 94%. For 3 fingers in contact, the available 6 components yield 87%. For 2 fingers in contact, the available 3 principal components yield 68%. We used a diagonal C_s matrix with values of 0.1 on the first three diagonal and 25.0 on the rest. This is because we

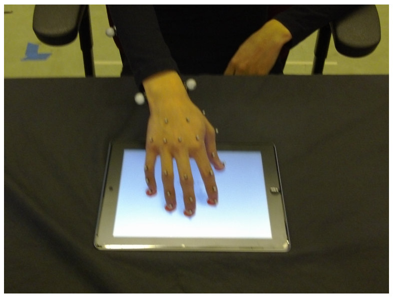


Figure 4.4: Motion capture session.

use meters and radians for our translation and joint angles in our application. We tested convergence of the algorithm vs. number of iterations and found 100 iterations to be sufficient for good convergence (figure 4.3). PCB IK is run for 100 iterations in all of our experiments.

4.4 Results

For our results, we train all models using a set of hand motions captured by using a conventional marker-based motion capture system and a multi-touch device at the same time. We used a conventional marker-based system from Vicon to capture the full 3D hand motions. A total of 23 retro-reflective markers were placed on the hand, and the marker positions in 3D space are captured using 10 cameras set at a close proximity to the hand. At the same time, a multi-touch device (iPad® 2) is used to obtain the fingertip contact positions. We used an open source software, TUIO [TUI], for transmitting the captured touch data from the touch device to the main computing machine. The motion capture setup is pictured in figure 4.4.

Since there was no synchronized clock between the two systems, we synchronized the data using the following method. First, we took the motion capture data and set a threshold on the finger height to estimate when the finger was in contact with the multi-touch device. Then, based on the number of contact points from the multi-touch device data, we found the time offset that minimized the difference in 1 ms resolution. Once the optimal time offset was found, the multi-touch contact points were matched to the closest finger in the motion capture data and labeled accordingly. The entire synchronization process was automated on the computer.

The hand model (figure 4.5) used in our experiments has 42 degrees of freedom in the hand and 6 degrees of freedom at the root joint for a total of 48 degrees of freedom.

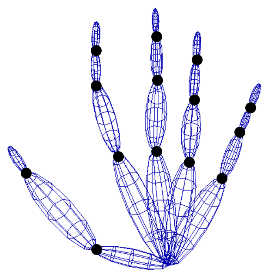


Figure 4.5: Our hand model contains 14 ball joints shown as black circles and 6 degrees freedom in the root joint for a total of 48 degrees of freedom.

This model is something of an arbitrary choice which was made to match the output we obtained directly from our Vicon system. It is a redundant representation, as actual hand motions likely exhibit many fewer degrees-of-freedom. We did not attempt to reduce the number of degrees-of-freedom because our method performed well from the raw Vicon data. However, our encoding technique should work equally well using a reduced representation that better matches actual human hand kinematics, or a reduced dimensional representation created using PCA, as long as the number of degrees-of-freedom used are sufficient to represent the user's actual hand motions.

For our training set, the subject was asked to perform moving, grasping, pinching, and rotating motions with various combinations of fingers. In particular, we captured sequences for 5 fingers, 2 different combinations of 4 fingers, 3 different combinations of 3 fingers, and 4 different combinations of 2 fingers. These finger combinations were chosen with the goal of representing a complete set of finger combinations that would be commonly used, feel natural, and involve more than one finger. The training database consists of 41 short trials of the above actions and finger combinations for a total of 7,083 postures.

The data pairs are used to train the encoding functions with least squares, as discussed in Section 4.2. Total time for offline processing was approximately 5 minutes.

A separate database was captured on a different day, in a different motion capture session, to be used in ground truth evaluation. Users were allowed to perform free-form gestures on the multi-touch device as if they were interacting with a virtual clay blob. There were some frames from this motion that used a single finger or a finger combination that was not in the training dataset. We removed these finger combinations for a resulting ground truth dataset containing a broad variety of motions and finger combinations in 2,538 total frames.

	Distance (cm)		Angle (degrees)	
	Mean	SEM	Mean	SEM
Linear	2.6419	0.0240	8.6017	0.0336
Quadratic	2.3664	0.0429	8.3840	0.0373
Cubic	4.0373	0.0922	12.3728	0.0702
NN	2.7463	0.0236	8.4193	0.0382
PCB IK	2.9907	0.0331	7.1985	0.0271

(a) Ground Truth Finger Labels Given

	Distance (cm)		Angle (degrees)	
	Mean	SEM	Mean	SEM
Linear	3.9837	0.0510	9.0774	0.0362
Quadratic	2.5520	0.0479	8.5656	0.0368
Cubic	3.1652	0.0415	9.9411	0.0424
NN	4.5403	0.0576	10.1003	0.0449
PCB IK	4.1488	0.0440	11.7594	0.0455

(b) Finger Labels Not Given

Table 4.1: Wrist position errors and overall joint angle errors in different methods. SEM stands for standard error of the mean.

4.4.1 Performance Comparison

Table 4.1 gives results comparing all techniques on the ground truth motion dataset. Two experimental setups were tested. The first experiment was designed to test the raw performance of each estimation technique when the input vector u (equation 4.1) does not contain errors, i.e., when the finger labeling exactly corresponds to the true contact pattern on the multi-touch device. Table 4.1a shows results from this experiment. Each row in the table is a different estimation technique described above. The “Distance” columns give the mean and standard error of the mean for Euclidean distance between the true wrist position as measured from the motion capture data and the wrist position as estimated by each algorithm. The “Angle” columns give the mean and standard error of the mean of the joint angle differences between ground truth and estimation, averaged over all joint angles in the hand model. For example, we see that for our quadratic encoding method, translational error was 2.37cm on average, and angular error was 8.38 degrees on average, whereas for the PCB IK approach, the corresponding errors are 2.99cm and 7.20 degrees. In this situation where finger labels are perfect, the performance of quadratic encoding and PCB IK are similar, with quadratic encoding performing better in estimating translations

	Accuracy (%)			
	2 Fingers	3 Fingers	4 Fingers	5 Fingers
Linear	52.66	34.04	64.35	66.87
Quadratic	55.66	70.72	79.36	99.88
Cubic	56.24	52.38	86.49	99.33
NN	11.78	21.34	46.74	99.55
PCB IK	51.85	51.50	48.15	62.88

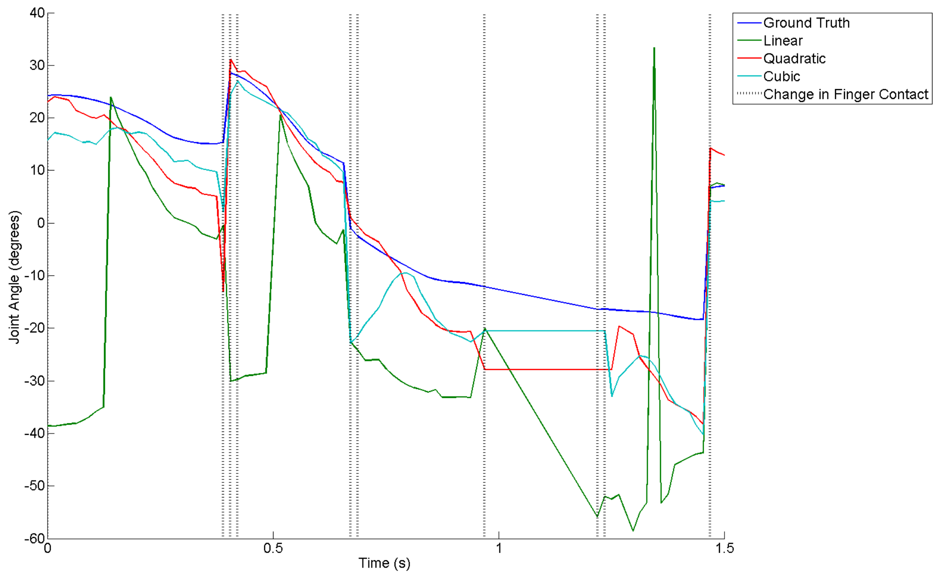
Table 4.2: Finger labeling accuracy in different methods per number of fingers.

and PCB IK performing better at estimating joint angles.

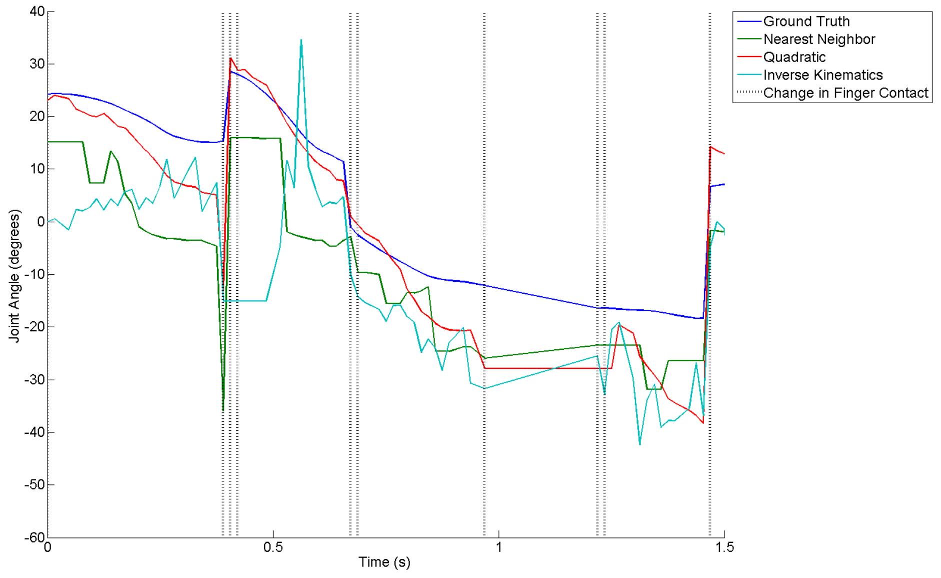
The second experiment shows errors when finger labels are computed using the techniques outlined in Sections 4.2 and 4.3. Specifically, all encoding techniques estimate all possible finger labeling patterns and choose the estimated hand pose that is closest to its nearest neighbor in the training database. The PCB IK and NN techniques simply choose the result with nearest neighbor in the space of the input vector. Table 4.1b shows results from this experiment. Here we see that quadratic encoding performs better than all other approaches, and it performs substantially better than PCB IK at estimating the ground truth pose.

To investigate performance further, we examine the accuracy of each algorithm in producing a correct finger labeling. The same ground truth motion capture dataset is used, and finger labelings are computed every frame. Table 4.2 shows these results. Here we again see that quadratic encoding has the overall best performance, although cubic is able to identify 4-finger labelings better than any of the alternatives for this dataset. For 3-finger motions in particular, quadratic encoding performs at the 70% level, whereas cubic and IK are both close to 50%. Even though the system sometimes gives incorrect finger labels, we found users were able to recover easily from wrong poses by lifting their hand and trying again, and in some cases, users were agnostic to which fingers were chosen because it did not make a difference for the intended task. This was especially true for the case of two fingers, where most combinations of fingers produce a similar result for object manipulation.

Estimation error is not the only metric of interest. We are also interested in the smoothness of the resulting motion. Figure 4.6 shows an example of the estimated wrist z rotation (yaw) over a section of ground truth motion. Wrist yaw is important because it is the most visible axis of wrist orientation for our subjects, and its smoothness and accuracy affect the naturalness of appearance of the entire hand. From this plot, we can see that linear encoding is often far from the ground truth, by as much as 60 degrees in this example. A linear



(a) Comparison across encoding methods.



(b) Comparison across other methods.

Figure 4.6: z rotation of wrist over time across different methods.

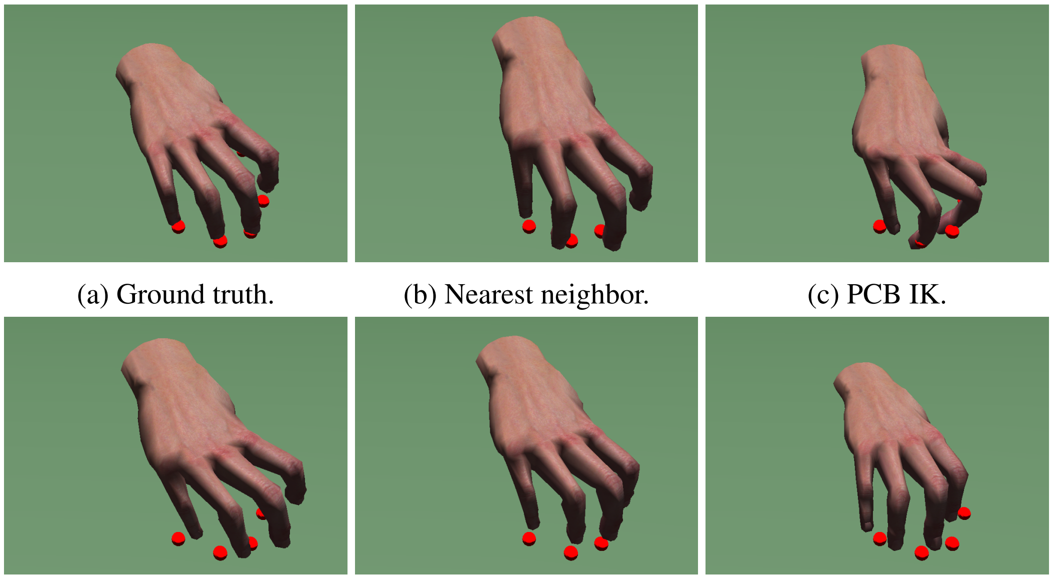

(a) Ground truth. (b) Nearest neighbor. (c) PCB IK.
(d) Linear encoding. (e) Quadratic encoding. (f) Cubic encoding.

Figure 4.7: Estimation results across different methods from the same input.

function does not appear to be sufficient to represent the collection of poses in the training dataset. Cubic encoding performs reasonably well in this example, but it occasionally shows large divergences, such as the one at 0.8 seconds. Such divergences are also seen in the accompanying video. These divergences suggest that cubic encoding may be overfitting to the training dataset. When inputs diverge from those seen in the training data, cubic encoding may perform poorly. The nearest neighbor approach is also often far from the ground truth, indicating that this example may be somewhat outside the range covered by the training dataset. In addition, it can give non-smooth results, as shown especially in the range of 0.7 to 0.9 seconds. Inverse kinematics does not match the ground truth well at times (e.g., around the 0.6 second mark). In addition, it is quite noisy, with solutions showing some oscillatory variation of 5 to 10 degrees from frame to frame. Quadratic encoding does a much better job of staying near the ground truth. It is also substantially smoother than the other estimation methods. We believe the smoothness is a direct result of using a continuously smooth quadratic function for encoding. As long as sudden input changes do not occur, the quadratic function guarantees a continuously smooth output value.

Figure 4.7 shows some side by side comparisons of the various approaches, indicating some of the failure modes that are observed. The nearest neighbor method (figure 4.7b) has trouble matching the fingertips to the target location, because the given pose is not in the training database. Inverse kinematics method (figure 4.7c) on the other hand, tries to fit the target position excessively and produces an unnatural pose. Note that the pose generated by the IK method does not always reach the user's contact points because we use a limited number of principal components. Across the encoding methods on the bottom row of figure 4.7, we can see that quadratic encoding (figure 4.7e) generalizes best over the training data and handles unseen data well.

More comparison examples can be found in the accompanying video. Please note the ground truth comparisons in the demo video show the results from computing finger labels for every frame to best give a sense of the raw performance of different algorithms. For this reason, the comparison results show large numbers of discontinuities for some of the methods. In the rest of the demos, the finger labels were kept until contact change, as outlined in our work.

4.4.2 Manipulation Demos

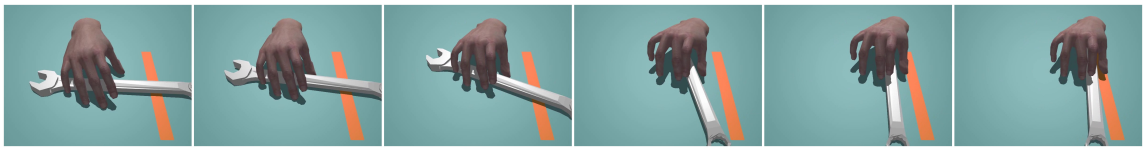
We tested our real-time system for ability to allow users to manipulate objects in a virtual environment. Subjects were given the tasks of moving a virtual box to a goal position, rotating a virtual wrench and sliding it to a goal position, and sliding a virtual box around. The accompanying demo video shows the results of these tasks and figure 4.8 shows some captures of these tasks. All of these tasks were successfully performed without any special instructions or training of the subjects.

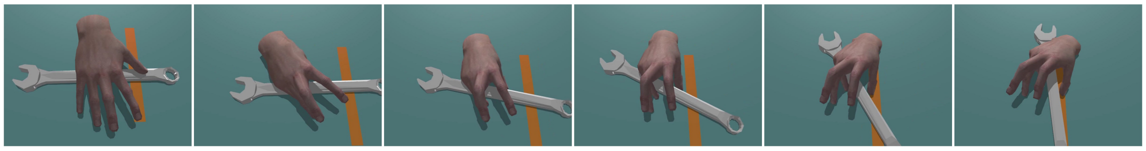
We also tested users with more challenging tasks where keyboard commands allowed users to translate the hand in all three dimensions. Subjects were given the tasks of moving the wrench to a target position on top of a box and rotating it, and stacking three boxes on top of each other. The accompanying demo video also shows the results of these tasks and figure 4.9 shows screenshots of the video.

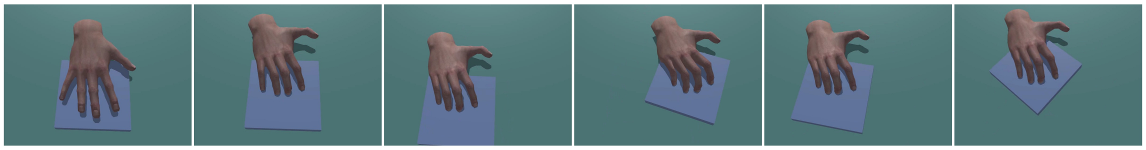
Note that no manipulation motions or transitions between different contacts were included in the training data. Instead, the training data consisted of short, isolated "scrubbing" type motions with fixed finger contacts. We suspect that a training set more specific to the task would give even better results.

4.5 Discussion

We have introduced a quadratic encoding technique for estimating hand pose from fingertip positions obtained from multi-touch devices in real-time. We have shown that this technique offers smooth, natural results with changing contacts and performs well even


(a) Rotating wrench with thumb and middle finger.


(b) Rotating wrench with thumb and pinky.


(c) Sliding box with four fingers.

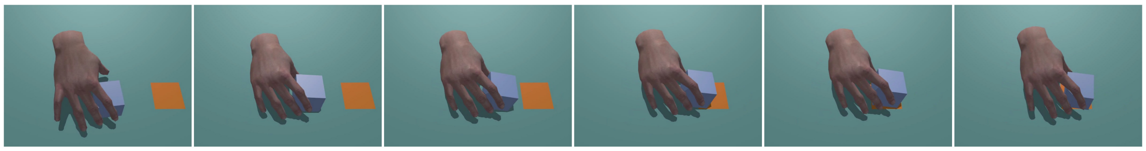
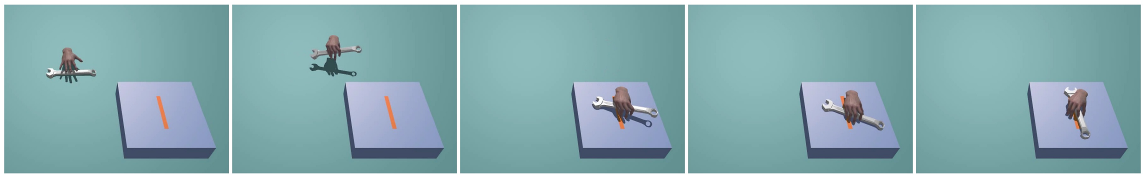

(d) Moving box to target.

Figure 4.8: Various manipulation tasks using only multi-touch device.


(a) Moving wrench to target.

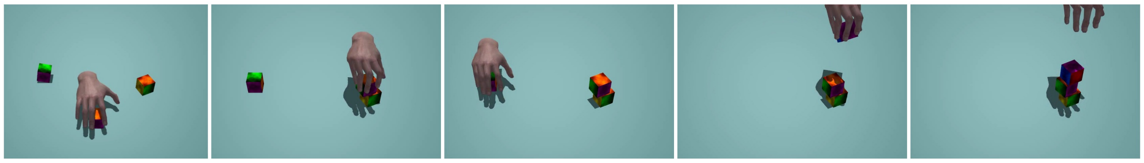

(b) Stacking boxes.

Figure 4.9: Various manipulation tasks using multi-touch device and keyboard.

though finger identities are not available from the input data. In this section, we further discuss why use quadratic encoding and cover limitations and future opportunities of this research.

4.5.1 Why Quadratic Encoding?

Why should quadratic encoding perform better than competing approaches? In its simplest form, the quadratic encoding method can be thought as a kind of approximation for the inverse transform, or inverse kinematics of the hand when in contact with the multi-touch device. The inverse kinematics function, which takes the end-effector position as input and returns the joint angles as output, is highly nonlinear. Even for a very simple two-link manipulator, the inverse kinematics solution involves a nonlinear arctangent function which is difficult to linearize. Our intuition is that the nonlinearity is important. However, the range of natural human finger motions is quite limited. Thus, a very simple nonlinear function – the quadratic function which we pose here – can approximate the inverse transform very well within the range of hand motions accommodated by our input device. More complex inputs may require more complex functions in order to represent the data well.

We can also ask why introduce quadratic encoding when we already have IK techniques that are widely used, are comparatively easy to implement, and have proven to be quite reliable in other circumstances. We have two answers to this question. First, a basic IK was actually the first approach which implemented, following the approach described in [Toh et al., 2012]. We subsequently implemented principal component based IK as described in [Mulatto et al., 2013] with the hope of obtaining more pleasing results. In both cases, we found that it was difficult to control the hand to move in the way we would expect, as the IK approach favored certain directions in pose space, which did not always match what the user was trying to do. Too much movement in any one direction would produce awkward and unnatural looking poses. In addition, it proved difficult to produce IK results that were both smooth and did not diverge from natural looking poses. We conclude that it is much better to use a learned transformation directly from inputs to outputs and demonstrate that the simple quadratic function that we propose here performs very well, as shown in the results and the accompanying video.

We further note that quadratic encoding dramatically outperformed the IK approach in terms of finger labeling. Use of a multi-touch device presents the interesting challenge of having precise fingertip positions but no idea of which finger is responsible for which contact. Quadratic encoding is capable of estimating natural hand poses so well that it can use the estimated pose to identify likely finger labels, producing a more seamless experience

	Computation Time (ms)
Linear	0.0021
Quadratic	0.0021
Cubic	0.0519
NN	0.0446
PCB IK (100 Iterations)	66.1898

Table 4.3: Computation time per pose across different methods.

for the user, whereas the IK approach fails in this regard, as shown in Tables 4.1b and 4.2.

In addition, quadratic encoding is extremely fast for real-time operation. With the IK technique that we tested, we were always pushing the boundaries of real-time performance, and when we were faced with the problem of resolving finger labels in addition, obtaining responsive estimations became very challenging. Table 4.3 shows timings for the various techniques discussed in our work. The table shows per pose computation time across different methods, which means that the actual computation time per frame can be many times higher if labeling is unknown. The computation times were computed on a machine with Intel® i7-2600 Quadcore CPU @ 3.40GHz.

There are other well known methods of regression such as radial-basis functions [Rose et al., 1998] and Gaussian process regression [Ikemoto et al., 2009] that would be interesting to compare against our approach. However, these alternative techniques can have difficulties with smoothness because of their ability to model more complex functions. We have seen that using a single low order encoding function, as we do with quadratic encoding, results in smooth and consistent results over the entire input space, even for situations when abrupt changes in the input occur such as lifting or putting down of the fingers on the multi-touch device. We believe that this smoothness and consistency gives quadratic encoding an advantage, especially for virtual manipulation tasks where smooth consistent behavior is critical. Fully testing this hypothesis, however, is left to future research.

4.5.2 Limitations and Future Work

There are several limitations of our system. First, the system is limited to estimating hand poses where at least two fingers are placed on the multi-touch surface. In addition, the multi-touch device does not identify non-fingertip contacts (e.g., contacts by knuckles), and this limits the types of hand poses that can be estimated by our system. However, we believe that our method can be generalized easily to a broader range of hand pose captures, given a mechanism for localizing and identifying important landmarks on the hand such as the fingertips and knuckles. Using one or more depth cameras would be a promising approach to extending our system. Incorporating new multi-touch technologies, such as sensing fingers with no touch [Sony Mobile, Inc.] and distinguishing between different parts of the finger on a multi-touch device [Qeexo Inc.], would also increase the capability of our system in the future. In general, our estimation approach is complementary to traditional motion capture. Theoretically, we can take as input whatever inputs are available and reliable and learn simple estimations from those inputs, which can be computed very quickly at runtime.

A second limitation is that we can only handle cases with more than one finger, because for a one finger case, there is too much ambiguity as to which finger it can be. We may be able to overcome this limitation by using more information on the contacts such as contact surface area in the future. Incorporating other system for finger contact identification (e.g., using a color based sensing similar to [Wang and Popović, 2009]) may further improve the overall estimation performance of our quadratic encoding technique.

Finally, we do not enforce environment contact for the physical simulation for fingers that are in contact with the multi-touch device – we are working from raw data only. This is to show the actual performance of the encoding method. It could easily be fixed in an actual manipulation application with some inverse kinematics.

Although our training data only includes short, isolated "scrubbing" type motions with fixed finger contacts, we demonstrate various manipulation tasks. We believe that with the use of a more task dependent training data, users will be able to perform manipulation tasks with more ease and better accuracy. One may even keep a set of various task dependent training datasets and swap them out on a need basis according to the given task.

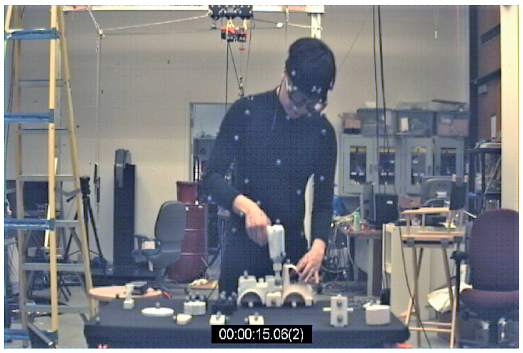
In the future, we hope to see how our method can coexist and actually boost performance of specialized hardware for hand pose tracking such as Leap Motion ([Lea]) or 3Gear Systems ([3Ge]). Our method of hand pose estimation is a form of regression which takes lower dimensional data and maps to the full dimension. It can supplement vision based devices to help in cases where there are occlusions to parts of the hand. Due to its simplicity, it can also be used in combination with other algorithms in a decision forest type of setting without adding much overhead.

There is also a new trend towards immersive virtual environments. With new exciting display technology such as Oculus Rift, it is only natural that users would want to see a real hand in the virtual environment even for tasks that require only simple gestures. With the help of precise fingertip positions provided by a multi-touch device, our work can provide an intuitive real time feedback in immersive virtual environments.

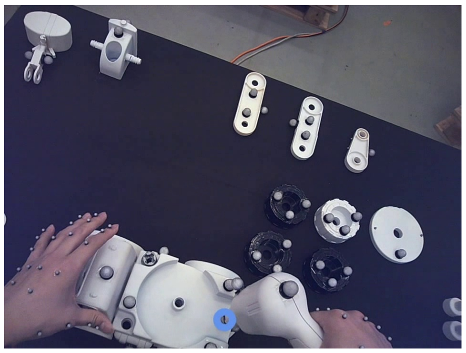
In summary, we find that human motions can be encoded with simple quadratic encoding functions that are trivial to evaluate and produce smooth pose transitions. Quadratic encoding may be useful in approaches beyond estimating hand pose, including full body pose estimation when constraints can be reliably obtained either through observation or user input.

Chapter 5

Observations of Hand-Eye Coordination



(a) Laboratory view.



(b) Egocentric view.

Figure 5.1: Toy truck assembly task.

Before we delve into hand pose prediction, we would like to present our motivations for using gaze to enhance hand pose prediction. To observe behaviors of gaze, we have captured a toy truck assembly task (figure 5.1). The eye movement during these activities were captured using an SMI Eye Tracking Glasses. Using this data set, we made many interesting observations that agree with previous literature from section 3.2.1 that gaze leads hand movements [Johansson et al., 2001], and it often gives cue for position and timing of the action [Flanagan et al., 2006]. In this section, we will summarize our observations of hand-eye coordination and discuss how each observation can help us to predict hand pose.

We present our observations of gaze behavior during different actions. We break up actions into grasping actions, manipulation actions, tool use actions, and inspection actions. We define grasping action as an action in which a hand moves towards the object in order to grasp the object into possession. Manipulation actions consist of the hand moving in order to to achieve a higher level goal than just grasping an object into possession. Examples of manipulation actions include spreading peanut butter on a piece of bread or tightening a screw to hold toy truck parts together. Tool use actions area actions in which a tool held by the hand interacts with other objects. Tool use actions are usually part of a manipulation action. Finally, inspection actions are actions in which hands orient the object to inspect task completion.

5.1 Gaze During Grasping Actions

The first observation is that the gaze leads the grasp by an average of 0.726s during grasping actions. The timing we observed is consistent with findings in previous literature [Johansson et al., 2001, Hayhoe et al., 2003]. Our average was obtained by measuring timings of all instances of grasping actions during the toy truck assembly task. The measurements were obtained by noting the time difference between the moment gaze focuses on the target object and the moment the hand touches the target object. Detailed timings of individual grasping actions are shown in figure 5.2 and table 5.1 lists the objects grasped during these actions. Some objects were grasped multiple times during the sequence.

Objects
Bolts
Drill Body
Nuts
Phillips Screwdriver Bit
Truck Base
Truck Cab
Truck Chassis
Truck Crane Base
Truck Crane Beam (Left, Right)
Truck Crane Beam Extension
Truck Crane Body
Truck Seat
Wheels

Table 5.1: List of objects grasped during our analysis of grasping actions.

This observation that the gaze leads the grasp is the key foundation in our approach of using the gaze to predict hand pose. We claim that because gaze gives out strong cues of

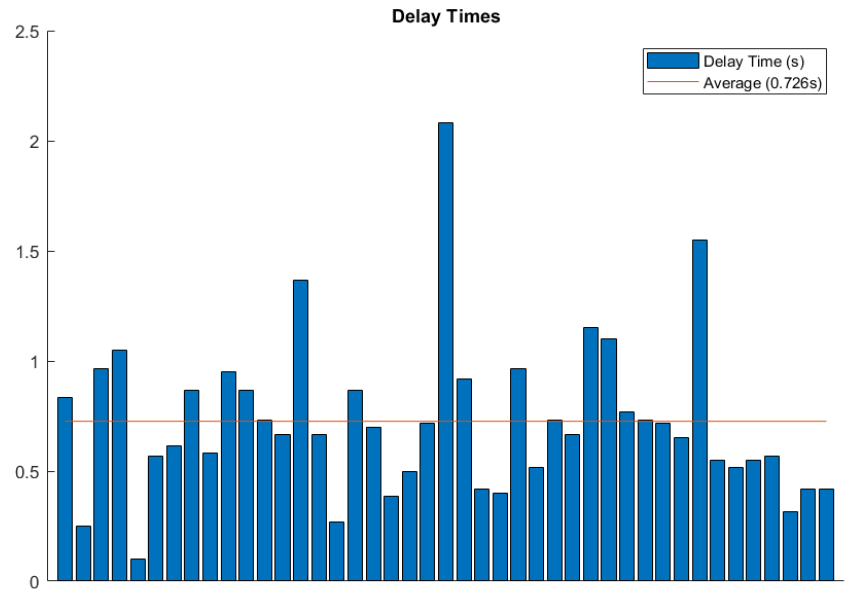


Figure 5.2: Delay between gaze and grasp in the toy assembly task.

the user's intention before the hand even touches the object, gaze can improve hand pose prediction performance.

5.2 Gaze During Manipulation Actions

The second observation is that gaze leaves for the next target object immediately after a manipulation action is finished. Figure 5.3 shows the start and end times for object manipulation and gaze fixation during eight different manipulation actions that happen during the same one minute segment of toy truck assembly. It can be seen that end times for gaze fixation and object manipulation lines up. The gaze does not necessarily precede object manipulation because of our definition of manipulation action. The target objects are usually already in possession of the hand when the manipulation action starts. Descriptions of the manipulation actions can be found in table 5.2.

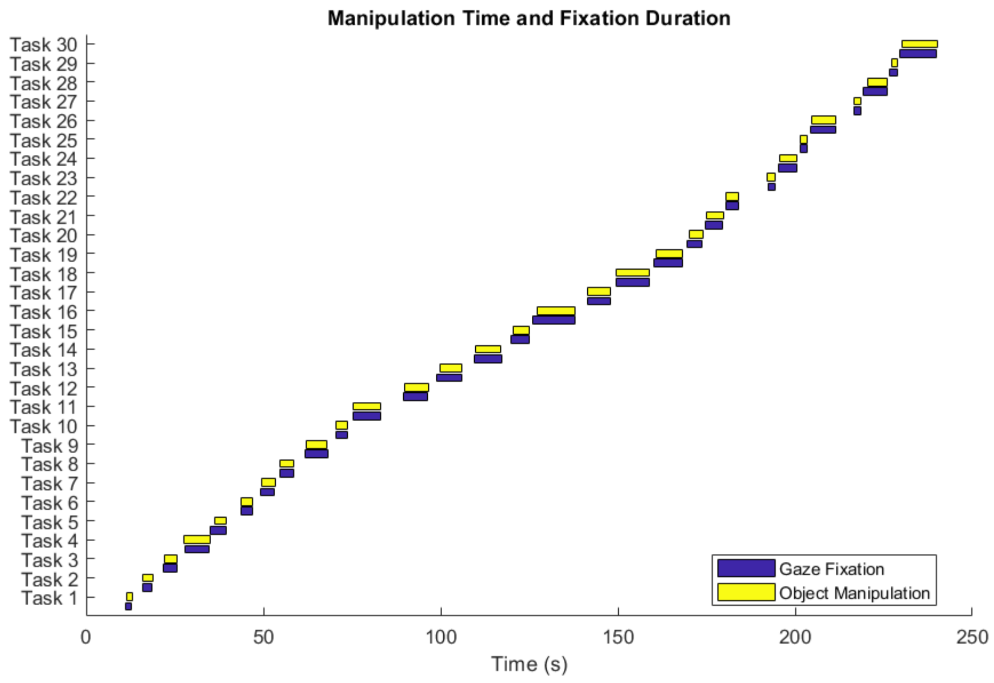


Figure 5.3: Manipulation time and fixation duration.

This observation gives us the insight that when the position of gaze changes, it signals the end of the current manipulation action and that a new intent should be predicted. We

Task	Description
Task 1	Place truck seat on top of the truck chassis.
Task 2	Fit the Phillips screwdriver bit on the drill body.
Task 3	Screw a bolt to fix truck seat on the truck chassis.
Task 4	Fit truck cab around the truck chassis.
Task 5	Fit truck base on top of the truck chassis.
Task 6	Screw a bolt to fix upper left corner of the truck base.
Task 7	Screw a bolt to fix right side of the truck base.
Task 8	Screw a bolt to fix lower left corner of the truck base.
Task 9	Fit truck crane base on truck base.
Task 10	Fit truck crane body on top of truck crane base.
Task 11	Screw a bolt to fix both truck crane base and truck crane body to the truck base.
Task 12	Insert right truck crane beam into right axle of truck crane body.
Task 13	Fasten right truck crane beam using a nut.
Task 14	Insert left truck crane beam into left axle of truck crane body.
Task 15	Push left truck crane beam in more.
Task 16	Fasten left truck crane beam using a nut.
Task 17	Situate truck crane beam extension between the two beams.
Task 18	Put a bolt through the two beams and truck crane beam extension.
Task 19	Fasten truck crane beam extension using a nut.
Task 20	Attach truck crane to truck crane beam extension.
Task 21	Put a bolt through truck crane beam extension and truck crane.
Task 22	Fasten truck crane using a nut.
Task 23	Attach front left wheel.
Task 24	Fasten wheel using a nut.
Task 25	Attach rear left wheel.
Task 26	Fasten wheel using a nut.
Task 27	Attach rear right wheel.
Task 28	Fasten wheel using a nut.
Task 29	Attach front right wheel.
Task 30	Fasten wheel using a nut.

Table 5.2: List of manipulation actions.

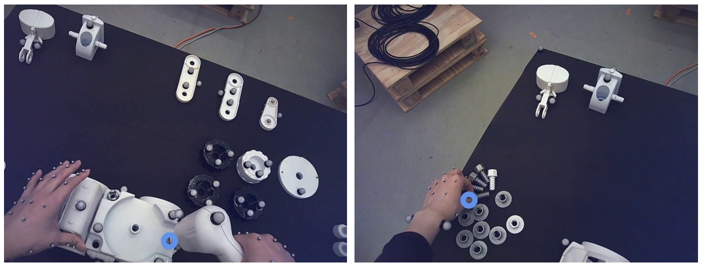

(a) Gaze during tool use. (b) Gaze without tool use.

Figure 5.4: Gaze behavior with and without tool use.

believe we can use these timings to switch to predictions of new actions when gaze position shifts after a manipulation action.

5.3 Gaze During Tool Use Actions

The third observation is that when a tool is held, the tool effectively becomes an extension of the hand. Gaze tends to fixate on the point of interaction, and we observed that gaze fixates on the tool tip when a tool is held. Figure 5.4 shows the difference of gaze fixation points depending on the presence of a tool in the hand. Figure 5.4a shows that the tip of the drill is fixated on while a screw is being fastened. On the other hand, when a bare hand is picking up a screw, the fingertip is fixated upon (figure 5.4b). This observation of gaze behavior means that when predicting the hand pose, we must also account for if the user is holding a tool in his hand.

5.4 Gaze During Inspection Actions

The last observation is that during inspection actions, the connection between gaze and hands are less apparent. Rather than directly acting at the point of fixation as in previous actions, hands stay in the peripheral vision and orient the object around while the eyes inspect completion of the task. Figure 5.5 shows some instance of the inspection action. In figure 5.5a, the eyes are inspecting whether the truck cab has been fit properly around the grill. In figure 5.5b, the eyes are inspecting the seam between the truck base and truck cab while the hands align the truck base over the groove behind the truck cab.

This observation signals us to look more closely towards the higher level intention

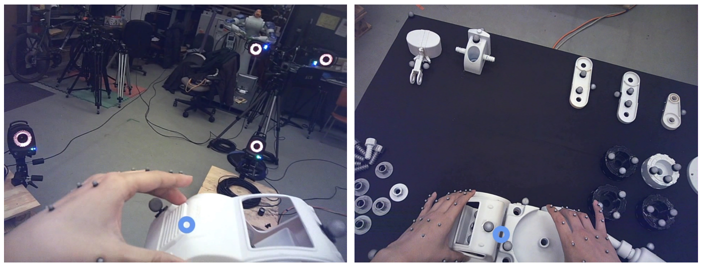

(a) Inspection while aligning the truck cab around the grill. (b) Inspection while fitting the truck base over the groove behind the truck cab.

Figure 5.5: Gaze behavior during inspection actions.

during inspection actions rather than using the gaze position as a target point for hand movement. This poses some challenge in predicting hand pose during these actions. However, occurrences of inspection actions were very rare in our data sets.

5.5 Discussion

We have confirmed findings from biological studies outlined in section 3.2.1 that gaze leads hand movements, and it often gives cue for position and timing of the action. In our analysis of a toy truck assembly task, we found that gaze leads the hand by an average of 0.726s during grasping actions, and duration of gaze closely aligns with object manipulation time during manipulation actions. Furthermore, we have observed that when tools are held, these tools effectively become an extension of the hand and shifts the fixation of the gaze towards the tool tip. However, gaze behavior during inspection actions were not directly related to hand movements, and care should be taken when relating gaze behavior to hand movement in this context.

Overall, our observations suggest we should be able to use gaze to predict grasps, because hand movements are led by gaze during grasping actions. The observations that are most relevant for predicting grasps are that gaze leads grasps, and gaze fixation duration corresponds with the manipulation duration. The fact that these behaviors were prominent in an everyday task suggests that we should observe similar behaviors in our experiments.

Chapter 6

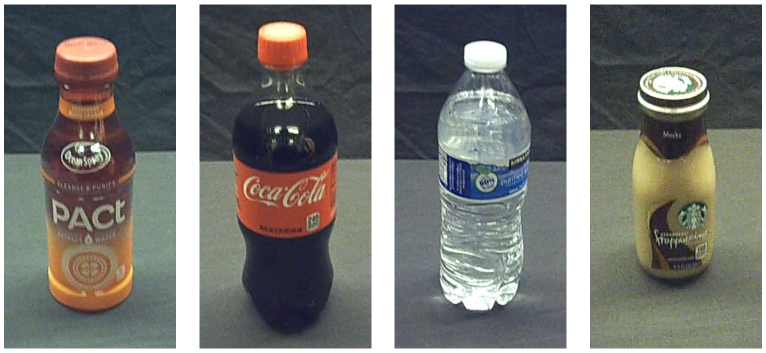
Augmenting Grasp Type Prediction with Gaze

Another important problem we must solve in order to advance towards direct manipulation in virtual reality is hand pose prediction. Predicting hand pose is an important problem in virtual reality where there is an inherent lag between pose tracking and rendering. This lag is further exacerbated when multiple users are trying to interact over a network connection. Suppose two people start to reach for an object simultaneously in a virtual environment. The delay in action delivery could cause each person to think that they have the object, because respective local systems will not be able to receive the other person's action until much later. In order to prevent awkward multiplayer interactions due to disagreement between simulation states of participating users, we must be able to predict hand pose and render them before the hand tracking data arrives.

It is well known that hand-eye coordination is important for object manipulation, but using this relationship to improve the accuracy of hand pose estimation is a relatively unexplored topic. We show that we can leverage eye tracking to improve hand tracking performance, and even predict future motions of the hand.

Eye tracking will be ubiquitous in the next generation of VR headsets, because of its numerous benefits to VR. It can help allocate rendering resources in the focused area with foveated rendering, allow for a more accurate stereoscopic rendering, help target objects using gaze, and add realism to social interactions. Major VR companies are already investing heavily into eye tracking for VR and being able to predict hand movement using gaze will only add to the benefits of eye tracking.

In this section, we will present our work on grasp type prediction done in a real-life



(a) Object 1 (b) Object 2 (c) Object 3 (d) Object 4

Figure 6.1: Objects used in the experiment.

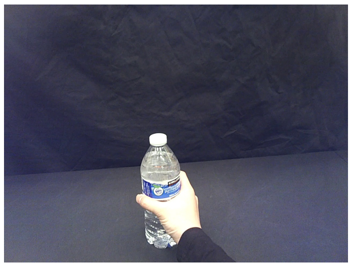
setting. Our work differs from previous works in that we use eye tracking information to improve grasp prediction. Our work also differs in that we focus on early prediction of grasp type before the hand reaches the target object. We show that both grasp type and grasp location prediction can be improved by utilizing gaze information. Although we only predict discrete grasp types here, we show the potential to further improve on hand pose prediction by demonstrating that eye tracking contains valuable information related to future hand movement.

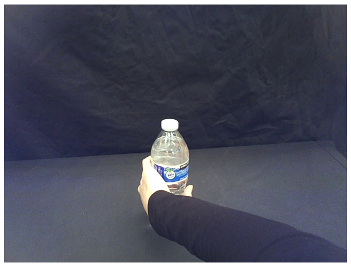
6.1 Grasp Type Prediction

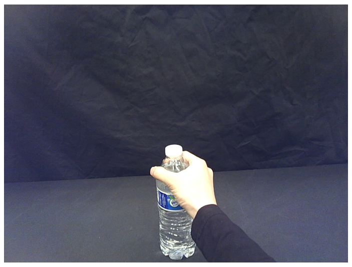
To start building towards our hand pose prediction framework, we designed a grasp type prediction experiment that shows gaze is helpful in predicting hand movement. The goal of the experiment was to simplify many aspects of the hand pose prediction problem in order to focus on a discrete grasp type prediction problem in a controlled environment.

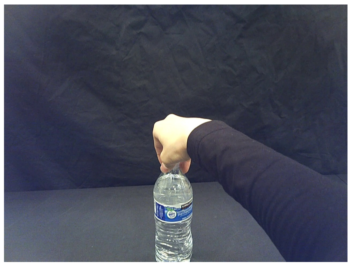
6.2 Experimental Setup

The experiment consisted of five subjects grasping four different types of bottles (figure 6.1) using five different grasps (figure 6.2). The objects were chosen such that they represent commonplace bottles with different shapes and sizes. The grasps were chosen such that they represent interactions with various parts of the bottles. Some grasps such as grasps 1 & 2 and grasps 4 & 5 were chosen to see the effects of gaze behavior when


(a) Grasp 1: body normal grasp.


(b) Grasp 2: body backhand grasp.


(c) Grasp 3: neck normal grasp.


(d) Grasp 4: top precision grasp.

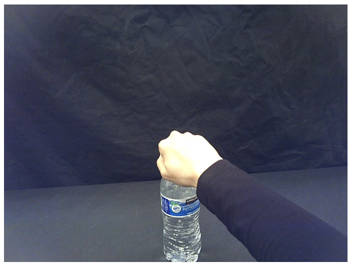

(e) Grasp 5: top wrapping grasp.

Figure 6.2: Grasps used in the experiment.

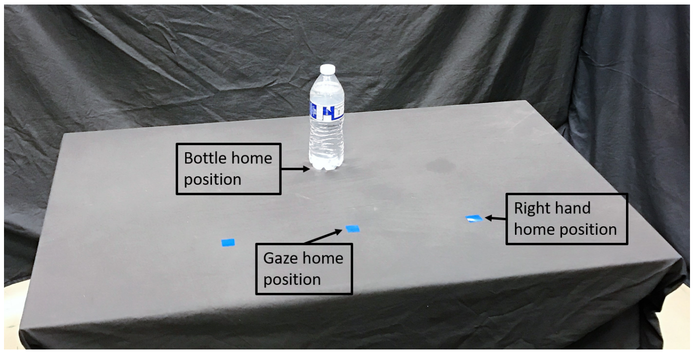


Figure 6.3: Home positions for grasp type prediction experiment.

grasping a similar region of the bottles.

During the capture session, each subject was asked to wear the SMI Eye Tracking Glasses. The Eye Tracking Glasses were calibrated using a 3-point calibration method. Then, each subject was asked to perform the five types of grasps in a random order. They were instructed to pick up the bottle briefly and put it back down at the same position to ensure a firm grasp. Also, they were required to rest their hands and gaze at their respective home positions before the start of each grasp (figure 6.3). The randomization of order and home position requirement allow us to obtain data that is independent of previous grasps. This set of five grasps was repeated repeated five times for the same object, for a total of 25 grasps.

We obtained a total of 58308 frames. The detailed breakdown of number of frames obtained per grasp is shown in table 6.1. Because non-grasps occur between every grasp, we have about 5 times more frames of non-grasps than any of the grasps.

6.3 Problem Setup

The problem we are solving in this experiment is an early prediction problem. Given the data from SMI Eye Tracking Glasses at each time step, we predict the grasp type from one of the 6 possible classes: one of the five grasps or idle. Because we predict the grasp type at each time step, many of the predictions are performed before the hand reaches the object. We focus on these early prediction results in our study.

The SMI Eye Tracking Glasses give us an egocentric video and gaze positions as pixel coordinates in the video frame. The video is captured at 24 fps and gaze position is tracked at 60 Hz. We manually labelled the begin and end times of each grasp and used this as

Subject	Object	Grasp 1	Grasp 2	Grasp 3	Grasp 4	Grasp 5	Non-Grasp	Total
Subject 1	Object 1	232	234	271	259	275	1581	2852
	Object 2	243	242	270	276	343	1243	2617
	Object 3	253	276	304	206	392	1465	2896
	Object 4	234	233	263	234	274	1371	2609
Subject 2	Object 1	293	297	293	292	312	1361	2848
	Object 2	319	343	338	318	337	1423	3078
	Object 3	297	308	301	292	303	1388	2889
	Object 4	283	297	290	267	276	1377	2790
Subject 3	Object 1	290	315	303	309	329	1583	3129
	Object 2	329	372	381	350	384	1692	3508
	Object 3	298	314	319	321	324	1609	3185
	Object 4	285	293	294	333	302	1416	2923
Subject 4	Object 1	198	233	216	225	228	1193	2293
	Object 2	214	235	254	231	258	1133	2325
	Object 3	235	179	231	258	259	1295	2457
	Object 4	1287	227	255	245	268	273	2555
Subject 5	Object 1	344	367	339	359	360	1434	3203
	Object 2	360	422	391	401	398	1686	3658
	Object 3	364	383	380	372	437	1478	3414
	Object 4	332	369	349	394	374	1261	3079
Total		5630	5967	6032	5965	6438	28276	58308

Table 6.1: Number of frames obtained for each grasp.

ground truth. The begin time was defined as the moment when the hand starts to move for the bottle. The end time was defined as the moment when the hand is completely out of contact after finishing the grasp.

6.4 How is Gaze Related to Grasps?

To answer the question “how is gaze related to grasps?”, we made some observations of gaze behavior during the given tasks. Figures 6.4 to 6.8 shows the average gaze region across all grasps in the data set, grouped by each grasp type. The average gaze is shown in green in each of these images. We obtained these images by normalizing each grasp instance across time into 48 time steps and averaging all the images at each time step.

The most interesting parts are in the first row for all of the grasps before the hand reaches the bottle. We can see that for body normal grasp (figure 6.4), the gaze is focused on the center area of the bottle before the hand reaches the bottle whereas for body back-hand grasp (figure 6.5), the gaze is focused on the left side of the bottle. This difference in gaze location shows that even when grasping similar area, gaze location can be different if the contact location is different. For the neck normal grasp (figure 6.6), the gaze is focused on the neck area before the hand reaches there as expected. For the top precision grasp (figure 6.7) and top wrapping grasp (figure 6.8), gaze is focused on the area above the neck towards the bottle cap. However, no obvious difference in gaze location exists between the last two grasps. This suggests that even if the grasps are different, if the contact area is similar, gaze alone cannot distinguish different grasps.

Another thing to note is that for all grasps, the gaze pattern diverges and focused area gets progressively fainter towards the end of the sequence. We noticed there are variances in the time it takes for each subject to lift and put the bottle back down. Subject 2 who preferred to spend more time holding the bottle up fixated on the grasp location while the bottle was held. The other subjects who were quicker at performing the lifting action did not or only briefly fixated on the grasp. The divergence in gaze pattern after initial contact with the bottle implies that history of gaze information should be important. Identifying fixations at key locations in the beginning of the grasp and keeping a memory of that state should help identify grasps across different individuals.

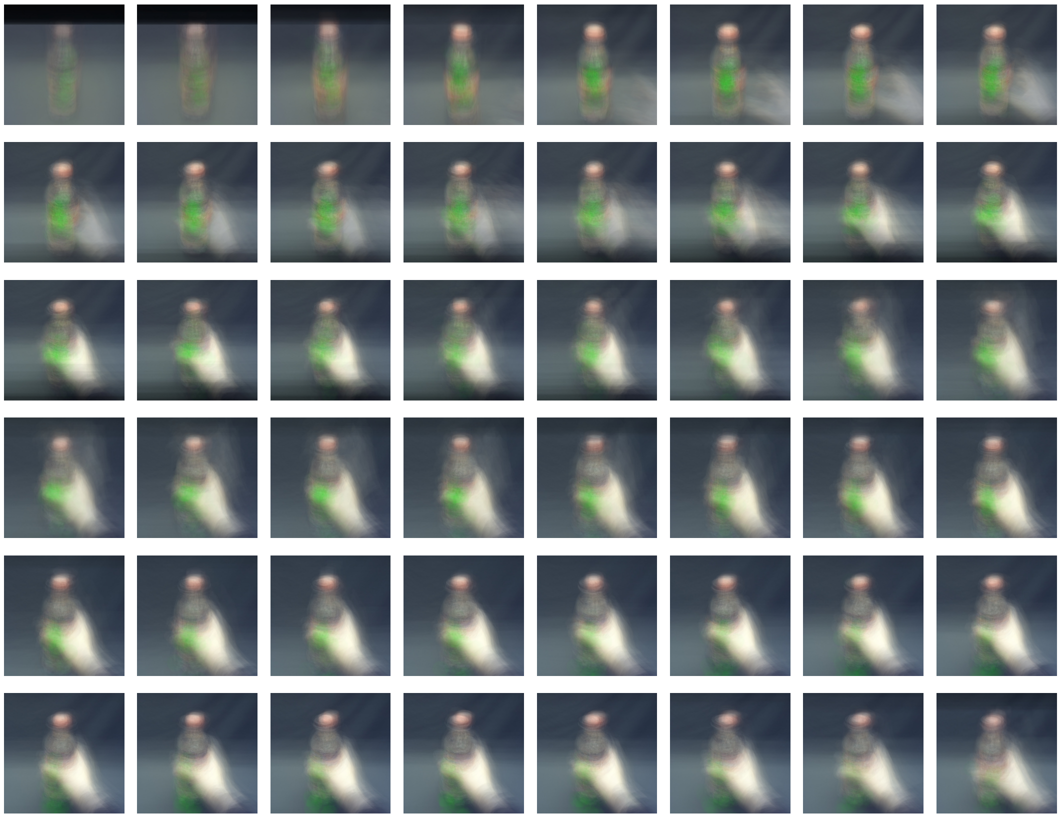


Figure 6.4: Average gaze for grasp 1 (body normal grasp).

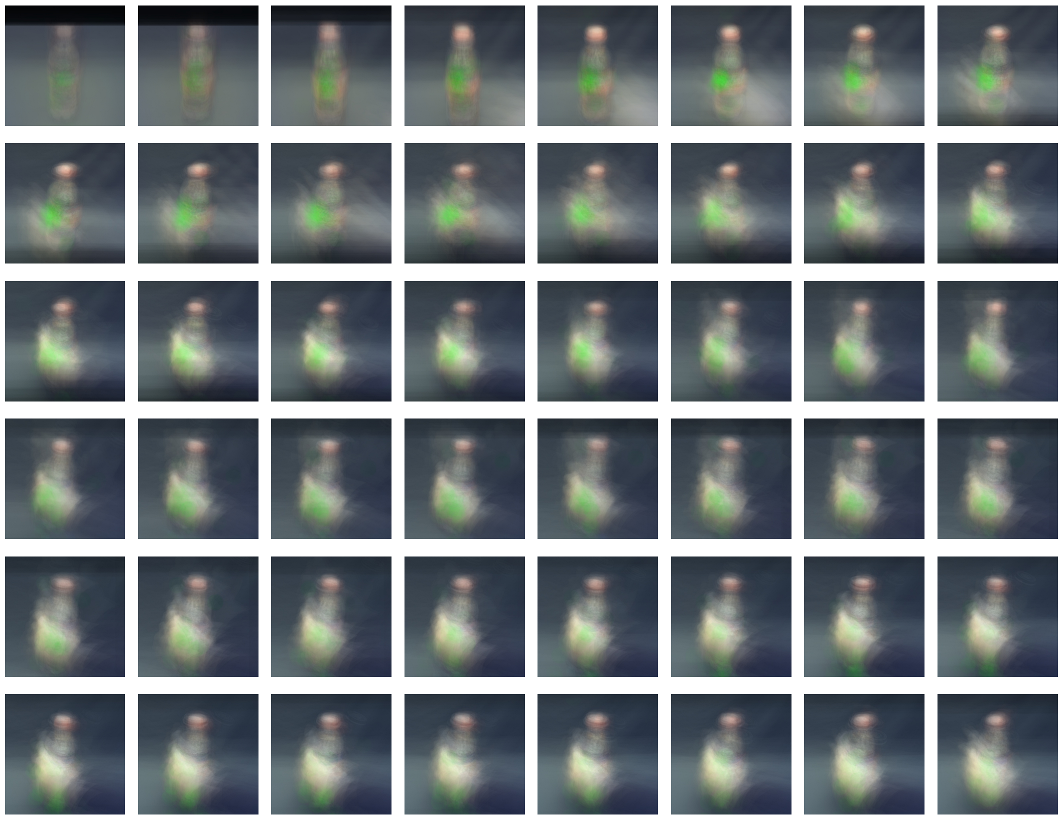


Figure 6.5: Average gaze for grasp 2 (body backhand grasp).

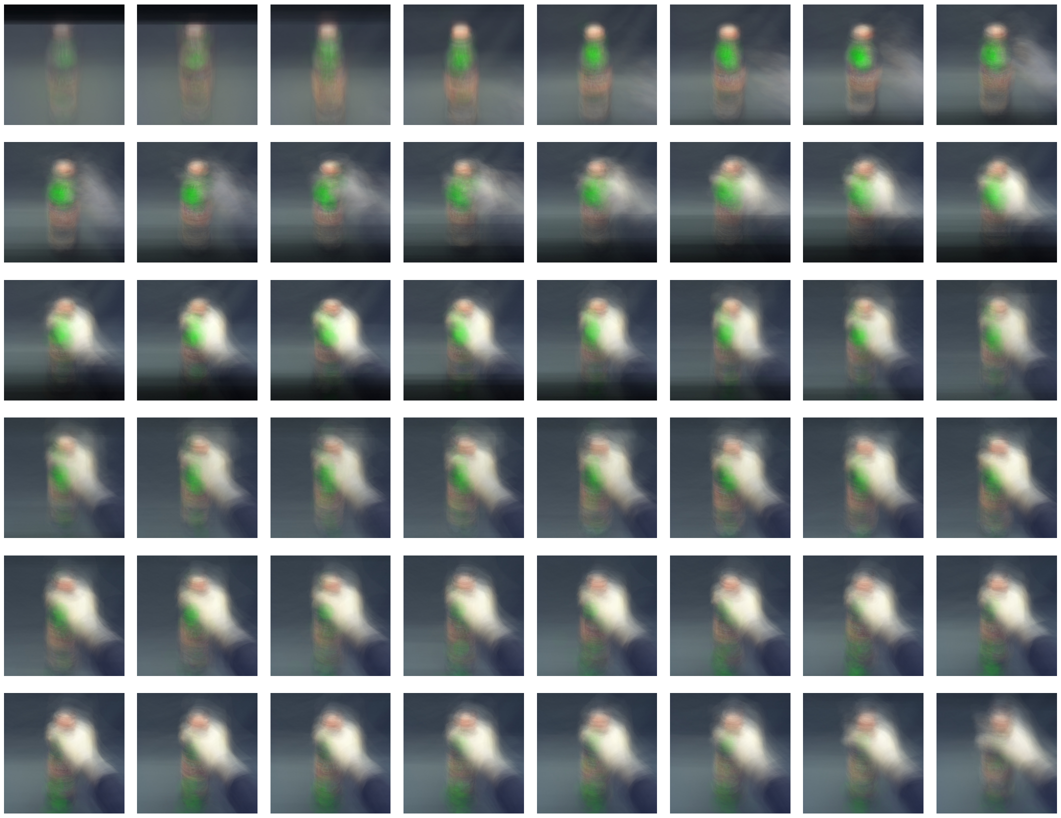


Figure 6.6: Average gaze for grasp 3 (neck normal grasp).

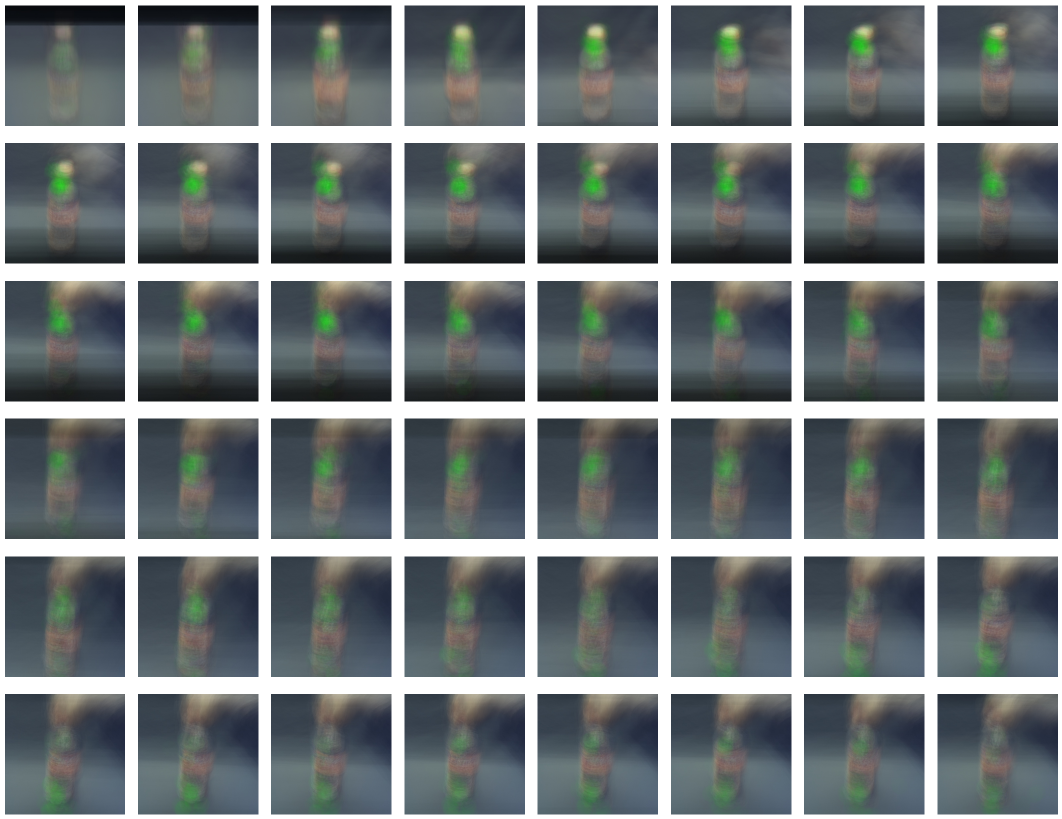


Figure 6.7: Average gaze for grasp 4 (top precision grasp).

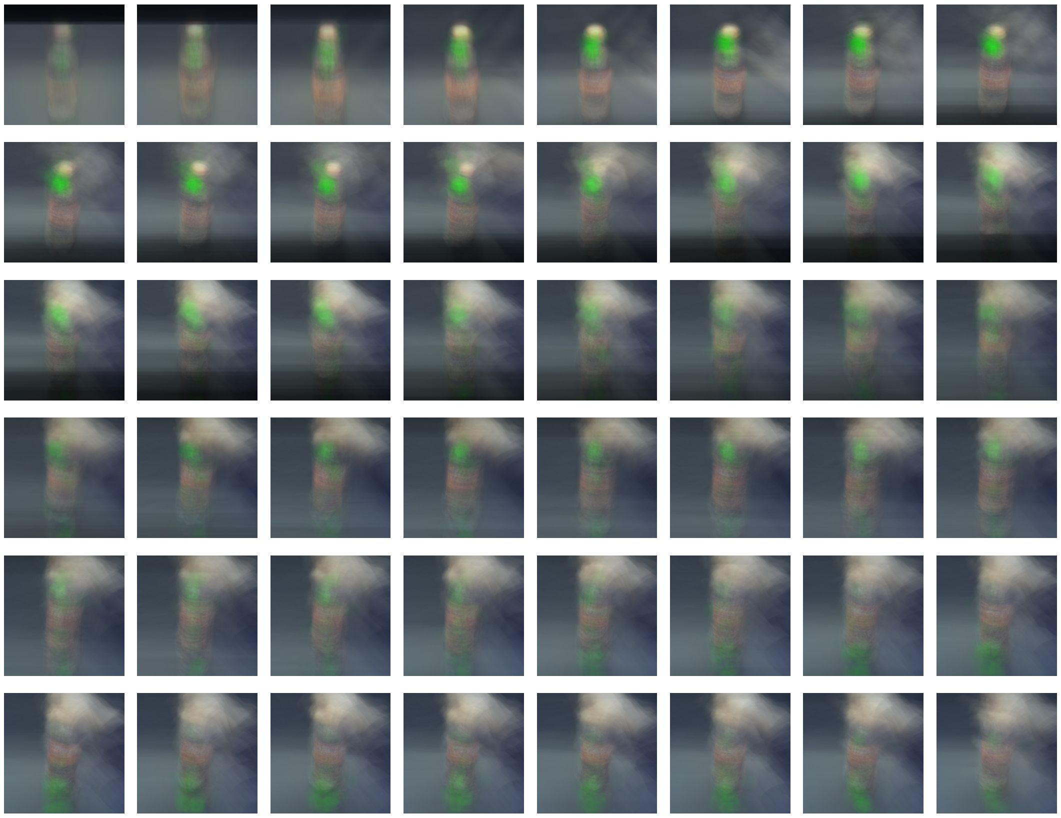
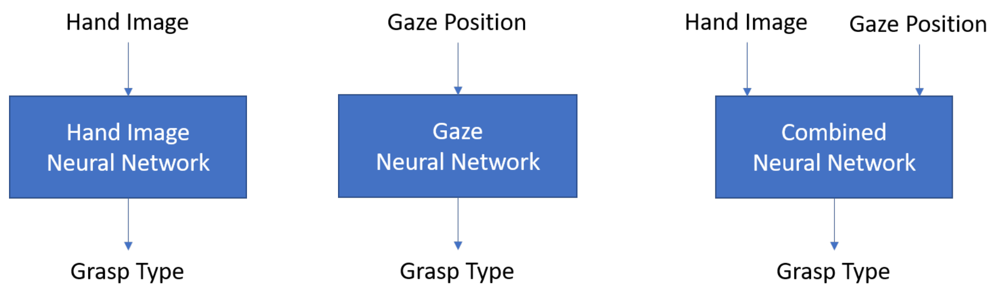


Figure 6.8: Average gaze for grasp 5 (top wrapping grasp).



(a) Hand image neural network only uses hand image as input.

(b) Gaze neural network only uses gaze position as input.

(c) Combined network uses both hand image and gaze position as inputs.

Figure 6.9: Inputs and Outputs of the three neural networks we designed.

6.5 Methods

Given our observations of gaze behavior, we concluded that people tend to look at regions not points and the timing of gaze before the hand is somewhat variable. However, there is definitely a pattern of gaze behavior albeit with some variances. Therefore, we decided to use neural networks which are good at learning complex patterns in data and can handle time dependence when long short-term memory (LSTM) units are used.

We designed three neural networks in order to compare the effectiveness of hand and gaze as separate inputs, then combine them to see if we can get a increased performance compared to using hand and gaze as separate inputs. Figure 6.9 shows the inputs and outputs of the three different networks. All networks output the likelihood of each grasp type, and we take the grasp type with the highest likelihood as the final prediction. The time step is measured in frames where the frames are 41.67 ms apart (24 fps). We will now describe each network in more detail.

6.5.1 Hand Image Neural Network

We call the first network 'hand image neural network.' This network takes in two versions of the image as inputs: the full image and a cropped image around the hand. The output of the hand image neural network is the likelihood of each grasp class. A detailed diagram of the components in this network is shown in figure 6.10. We employ a pre-trained VGG-19 convolutional neural network which is originally an object detection network. Object

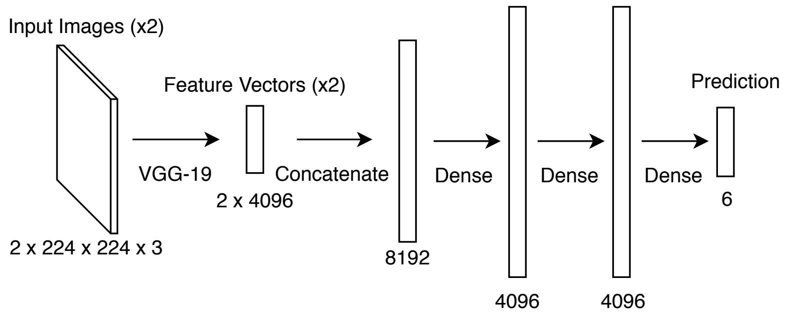
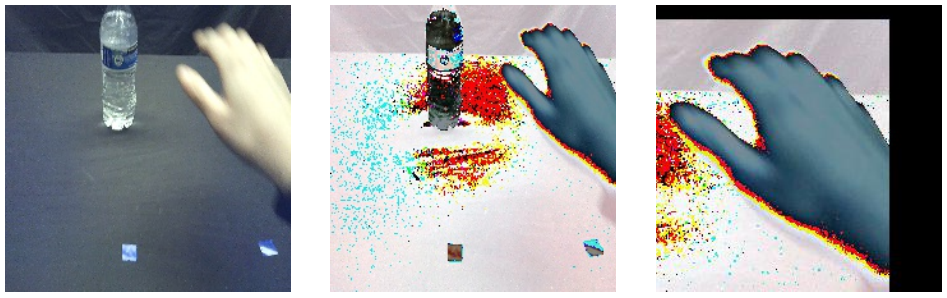


Figure 6.10: Hand image neural network. The numbers denote the dimensions of the vectors going through each layer.

detection problem is the problem of correctly identify and draw bounding boxes around objects. The VGG-19 network corresponds to configuration E in the work by Simonyan and Zisserman [Simonyan and Zisserman, 2014] and it was trained with millions of images from ImageNet. We chose to adopt a pre-trained object detection network, because in order to distinguish between numerous objects, it has to know about detecting various shape features. Here, we repurchase the VGG-19 network as a feature extractor to distinguish between various hand shapes. The technique of taking an existing network and repurposing it is called transfer learning, and it is quite common in the machine learning community. The decision to use multiple versions of the input image was inspired by the approach in Rogez et al. [Rogez et al., 2015], but we empirically determined that the full image and cropped image combination yields the best results. Rather than using support vector machines (SVMs) as in Rogez et al., we use neural networks to learn from the features output by VGG-19 which we have also found to yield better results.



(a) Original frame.
(b) Mean-subtracted frame (input 1).
(c) Mean-subtracted and cropped frame (input 2).

Figure 6.11: Input image processing for hand image neural network.

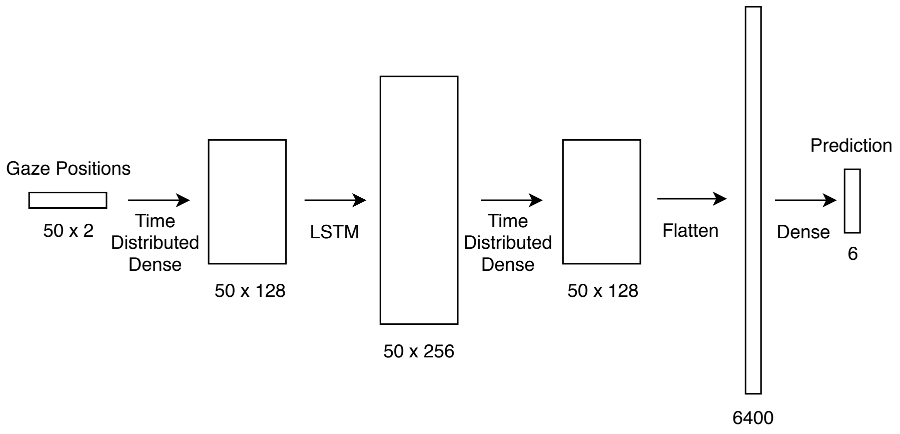


Figure 6.12: Gaze neural network. The numbers denote the dimensions of the vectors going through each layer.

In order to obtain the inputs to the hand image neural network, we first re-size the images to 224 by 224 pixels in order to match the pre-trained network dimensions. Then, we subtract mean RGB value of $(123.68, 116.779, 103.939)$ from each pixel of segmented hand images to zero-center the pixel values. The mean RGB value is obtained from the VGG network. The cropped image was obtained by first segmenting the hand using the hand segmentation tool by Li and Kitani [Li and Kitani, 2013], then cropping the image around the centroid of the hand segmentation. Sample input images are shown in figure 6.11. These input images are processed through a pre-trained VGG-19 network which is used as a feature extractor.

The last three layers are fully connected layers which learn the mapping between various features outputted by the VGG-19 network to the likelihood of the hand belonging to a grasp type.

Due to memory constraints, we ran the training with the weights in VGG-19 fixed. We used stochastic gradient descent optimizer with learning rate of $1\mathrm{e}{-4}$, decay of $1\mathrm{e}{-6}$ and Nesterov momentum of 0.9. The training was run for 40 epochs with a batch size of 32. Class weighting of 0.2 for non-grasps and 1 for all grasps was used to work out the imbalance of sample size across different classes.

6.5.2 Gaze Neural Network

The second network is the 'gaze neural network.' This network takes in a sequence of relative gaze positions with respect to the object and outputs the likelihood of each grasp

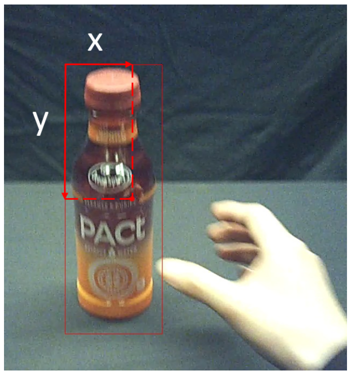


Figure 6.13: Relative gaze position with respect to the upper left corner of detected object bounding box is used to generalize better to different bottles.

class. A diagram of this network is shown in figure 6.12. We employ a sequence of previous relative gaze positions, because we observed the gaze fixation location across different subjects only agree at the beginning of the grasp (section 6.4). Having a history of recent gaze positions would help out with the prediction by being able to identify if the subject had looked at one of the key regions that signal a specific type of grasp. We used a sequence length of 50, because we found most grasps span around 50 frames. In order to process and learn to remember important information in the gaze position sequence, we use a long short-term memory (LSTM) layer in the gaze neural network. A relative gaze position with respect to the object is used in order to generalize better to different bottle locations in the scene (figure 6.13). We detect the bottles in the scene using an object detector network called YOLOv2 [Redmon and Farhadi, 2016], and use the upper left corner of the bounding box as the origin. We found the object bounding box to easily miss parts of the bottle especially while the hand is grasping the bottle, but the upper left corner of the bounding box was relatively stable even when parts of the bottle was occluded. The gaze network was trained with the same settings as the grasp network except with a learning rate of $1\mathrm{e}{-2}$.

6.5.3 Combined Neural Network

The third network is the 'combined neural network.' This neural network combines the first two networks by concatenating the vectors going into the last three dense layers of hand image neural network and the gaze neural network. Since the layers after the concatenation have the same architecture as the layers after the flatten layer of hand image neural network, this effectively means that the combined network is same as a network

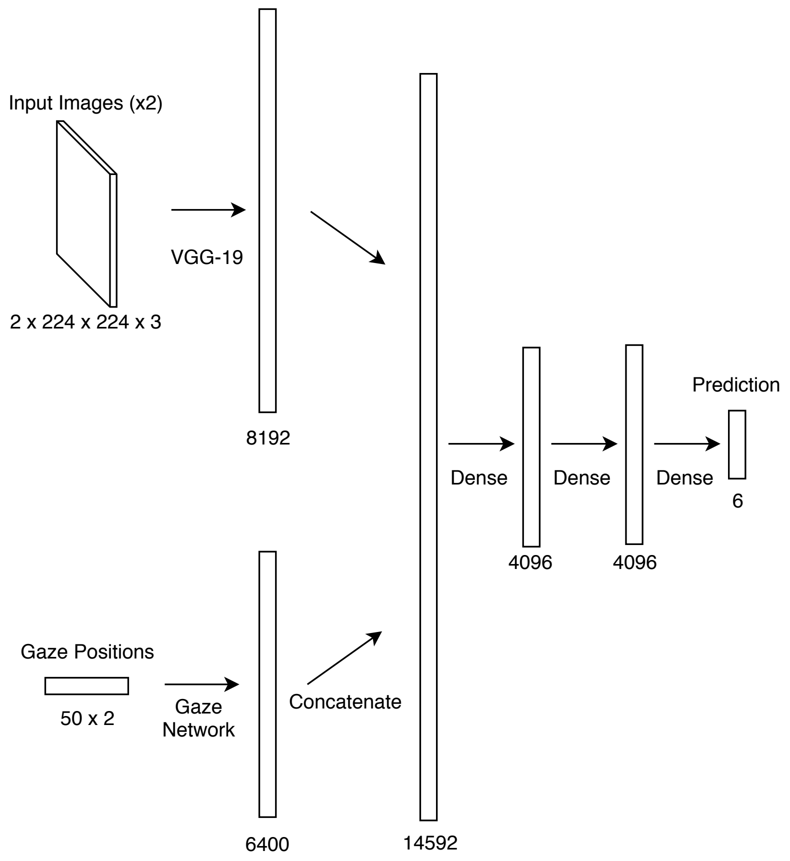


Figure 6.14: Combined neural network. The numbers denote the dimensions of the vectors going through each layer.

where the processed gaze features added to the hand image neural network. Same settings were used as hand image neural network except with a learning rate of $1\mathrm{e}{-3}$.

6.6 Results

We performed a leave-one-out cross validation across all four objects (table 6.2) and all five subjects (table 6.3) using the neural networks that were described in section 6.5. We evaluated the prediction error on a per frame basis. Because non-grasps had the most number of frames and all networks were better at telling grasps from non-grasps than distinguishing between the grasps, evaluating prediction error across all labels do not reflect how accurately the networks can tell the grasps apart. Therefore, we present two metrics: error which represents the prediction error across all frames, and grasp error which represents the prediction error only within the frames that were labeled as a grasp.

In both cross validations, all of the networks perform better than the random chance error which is 0.83. Furthermore, we can see that adding gaze information to the hand image inputs improves the prediction results with the exception of subject 5 in table 6.3. For subject 5, we observed the subject's tendency to fixate on the neck of the bottle during body grasps which would have reduced the effectiveness of added gaze information in the combined neural network.

Overall, we can see improvements in the combined network for the overall mean error rates, which includes performance improvement for distinguishing non-grasps from grasps, as well as for grasp error rate, which shows prediction among different grasps. We found the resulting improvement in the combined neural network over the hand image neural network to be statistically significant with t-test's p-value being less than 0.001. This value was obtained by comparing the distribution of predicted likelihood corresponding to the ground truth grasp types at each frame.

Figure 6.15 shows the normalized confusion matrices for the hand image and combined neural networks. We can see that grasp 1 has the most improvement in prediction results with the added gaze information. This makes sense because grasp 1 was often confused as grasp 3 without using gaze information. Grasps 1 and 3 are the same normal grasps at body and neck respectively, which would be hard to distinguish from only the hand shape. We also see reduced confusion with non-grasps across all grasps.

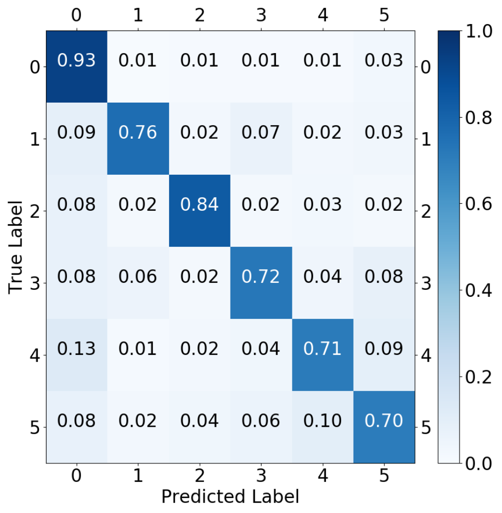
To observe in more detail how gaze helps improve the prediction quality, we made area plots of predicted likelihood over time during grasp 1. The plots in figure 6.16 was created by normalizing all instances of grasp 1 in our prediction results into 50 time steps which

		Hand Image Neural Network	Gaze Neural Network	Combined Neural Network
Object 1	Error	0.131	0.304	**0.098**
	Grasp Error	0.225	0.471	**0.187**
Object 2	Error	0.170	0.336	**0.129**
	Grasp Error	0.166	0.571	**0.164**
Object 3	Error	0.182	0.304	**0.165**
	Grasp Error	0.335	0.523	**0.309**
Object 4	Error	0.161	0.324	**0.136**
	Grasp Error	0.289	0.566	**0.245**
Mean	Error	0.161	0.317	**0.132**
	Grasp Error	0.254	0.533	**0.226**

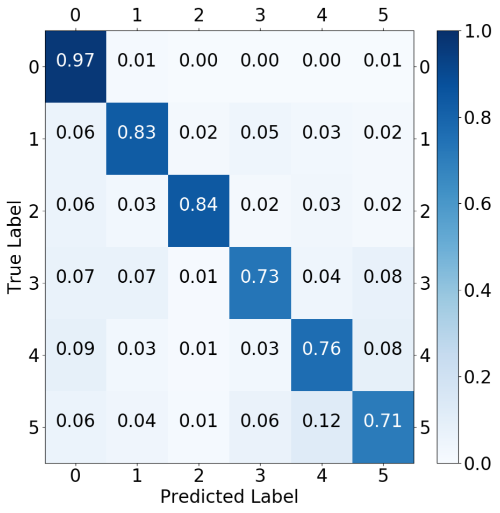
Table 6.2: Prediction error in leave-one-object-out cross validation (best results in bold).

		Hand Image Neural Network	Gaze Neural Network	Combined Neural Network
Subject 1	Error	0.139	0.382	**0.108**
	Grasp Error	0.247	0.551	**0.189**
Subject 2	Error	0.116	0.326	**0.092**
	Grasp Error	0.179	0.544	**0.161**
Subject 3	Error	0.205	0.329	**0.160**
	Grasp Error	0.382	0.610	**0.293**
Subject 4	Error	0.199	0.378	**0.165**
	Grasp Error	0.293	0.659	**0.287**
Subject 5	Error	**0.184**	0.424	0.191
	Grasp Error	**0.308**	0.683	0.335
Mean	Error	0.168	0.368	**0.143**
	Grasp Error	0.282	0.609	**0.253**

Table 6.3: Prediction error in leave-one-subject-out cross validation (best results in bold).



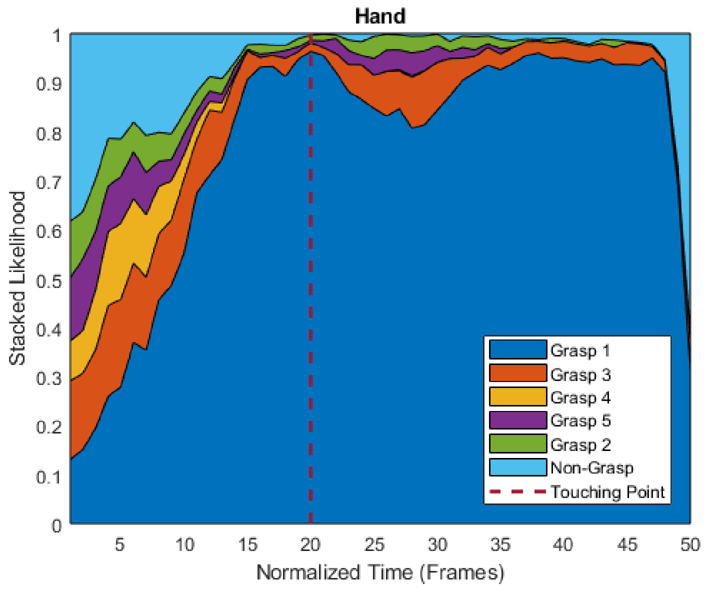
(a) Normalized confusion matrix for hand image neural network.



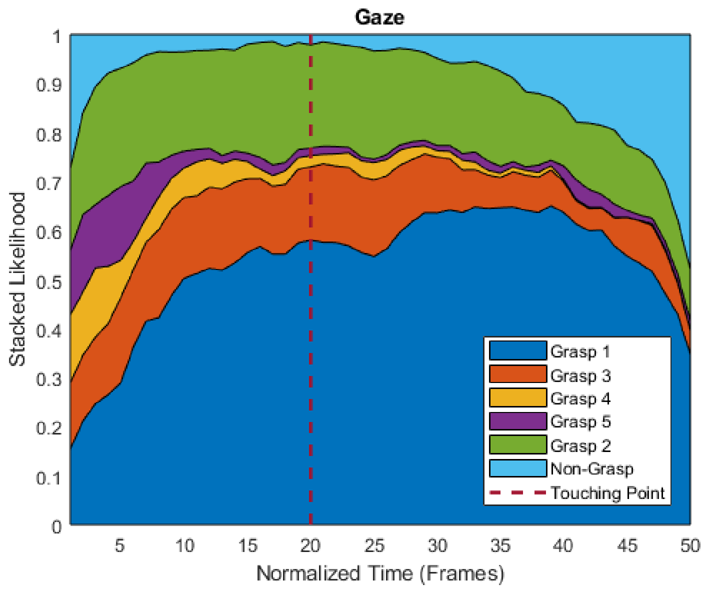
(b) Normalized confusion matrix for combined neural network.

Figure 6.15: Normalized confusion matrices for the hand image and combined neural networks. 0 denotes non-grasp and other numbers correspond to grasp types.

is roughly equal to 50 frames since most grasps are around 50 frames long. Figure 6.16a shows that the hand image neural network takes some time to reduce the likelihood of a non-grasp after the grasp had begun. It also predicts the likelihood of grasp 3 to be somewhat high at all phases of the grasp, confirming that it is indeed hard to distinguish grasp 1 from 3 using only the hand shape. The predictions of gaze neural network (figure 6.16b) shows different qualities. The gaze neural network is able to rule out the possibility of non-grasp almost instantly after the grasp has begun. However, it suffers from confusion between grasp 1 and 2. This makes sense because grasps 1 and 2 are both located at the body of the bottle, so their gaze patterns will be located in similar areas. The combined neural network is where the good qualities of both inputs work together to improve the overall quality of prediction. Figure 6.16c shows that by adding the gaze information, the combined neural network is able to rule out the possibility of non-grasp quicker than the hand image neural network. It also shows that likelihood of predicting grasp 3 has been reduced with the help of gaze inputs. But the combined network does not suffer from the confusion between grasp 1 and 2 when using the gaze inputs, because hand image inputs are available to help distinguish between those grasps. Therefore, the overall likelihood of predicting grasp 1 correctly is improved throughout.

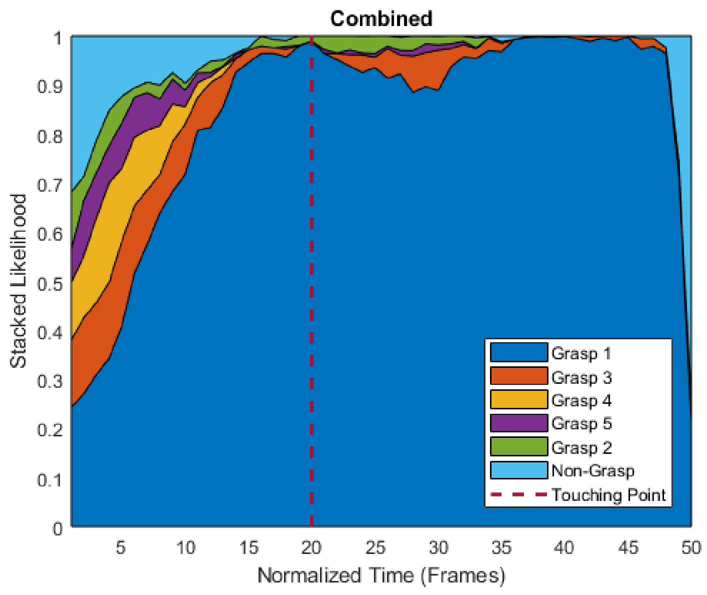


(a) Area plot of predicted likelihood for hand image neural network during grasp 1.



(b) Area plot of predicted likelihood for gaze neural network during grasp 1.

Figure 6.16: Area plot of predicted likelihood across neural networks during grasp 1.



(c) Area plot of predicted likelihood for combined neural network during grasp 1.

Figure 6.16 (Continued): Area plot of predicted likelihood across neural networks during grasp 1.

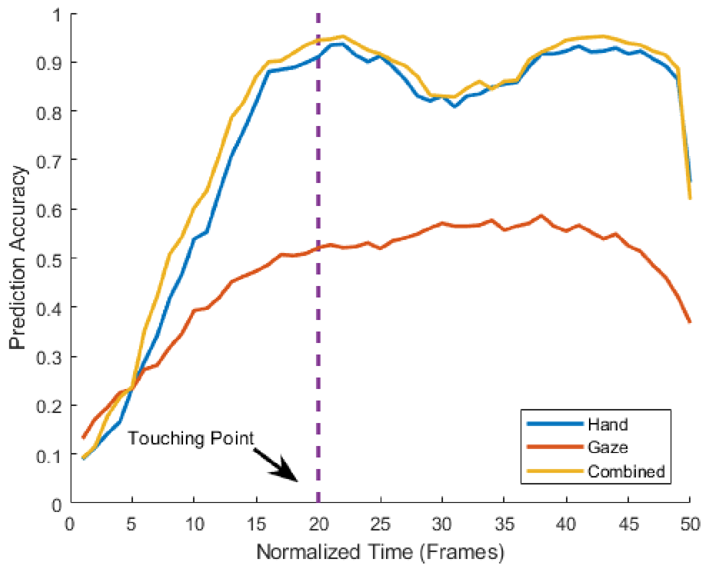


Figure 6.17: Prediction accuracy during grasps across neural networks.

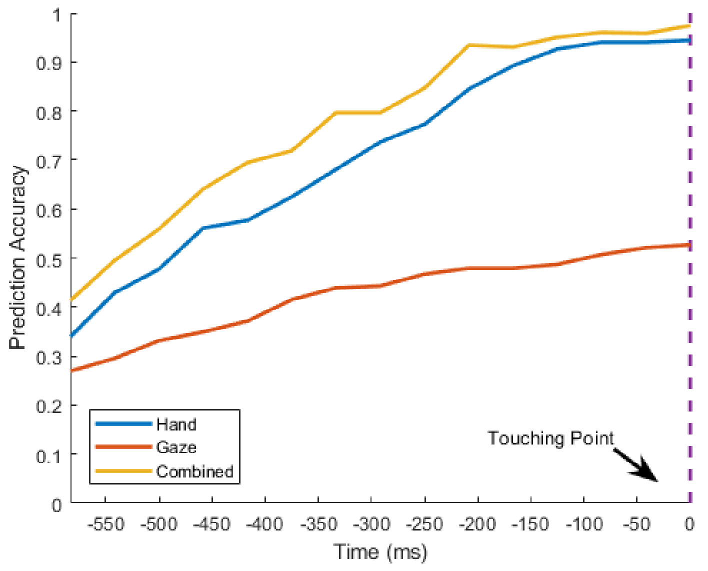


Figure 6.18: Prediction accuracy before touching point across neural networks.

We also looked into the details of how the prediction quality changes over time during all of the grasps. Similar to the previous plots, all grasps in our prediction results were normalized into 50 time steps. The prediction results over time is shown in figure 6.17. We can see that gaze has better accuracy at the beginning of the grasp. Addition of gaze information helps grasp prediction across all phases of the grasp. The dip in the prediction accuracy for the hand image and combined neural networks from time steps 22 through 41 were found to be due to the hand going out of field of view while the bottle is being lifted.

Figure 6.18 shows the prediction accuracy timeline before touching point without time normalization. The plot spans a length of 625 ms which was the length of shortest duration from grasp begin to touching point. This plot gives a sense of when our early prediction starts to become fairly accurate. In this experiment, we observed that the prediction accuracy passes 0.7 around 375 ms before the touching point for the combined network versus around 292 ms before for the hand image network. The prediction accuracy passes 0.9 around 208 ms before the touching point for the combined network versus around 125 ms before for the hand image network.

6.7 Stick Grasping Experiment

6.7.1 Experimental Setup

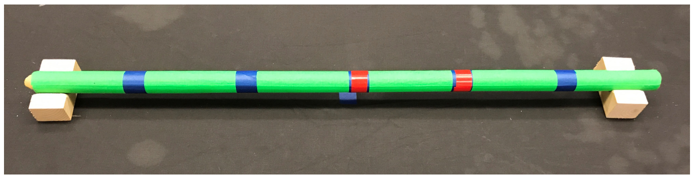


Figure 6.19: Object used in the stick grasping experiment. Asymmetry in tape marking was used for object tracking.

In this experiment, we asked the same five subjects from the controlled grasping experiment to grasp a stick (figure 6.19) at one of the five marked locations in a random order and predicted the locations using the same neural networks. The subjects were asked not to vary their grasps, as the goal of this experiment was to test the effectiveness of gaze for distinguishing grasp locations when similar grasps were used. As in the controlled grasping experiment, the subjects were asked to briefly lift the stick, put it back down at the marked home position, and reset their hand and gaze to their respective home positions. Ground truth labels were manually labeled.

6.7.2 Stick Grasping Experiment Results

Because we only had one stick for this experiment, we performed a 5-way cross-validation across each of the subjects. The error metrics are reported in table 6.4. In this experiment, gaze neural network's prediction accuracy compares closely with that of hand image neural network, because we are only predicting the location of the grasp and not varying the hand shape. Overall best accuracy is achieved by the combined neural network. We found the resulting improvement in the combined neural network over the hand image neural network to be statistically significant with t-test's p-value being less than 0.001. This value was obtained by comparing the distribution of predicted likelihood corresponding to the ground truth grasp types at each frame.

		Hand Image Neural Network	Gaze Neural Network	Combined Neural Network
Subject 1	Error	**0.167**	0.240	0.179
	Grasp Error	0.303	0.323	**0.191**
Subject 2	Error	0.188	0.311	**0.152**
	Grasp Error	0.240	0.449	**0.230**
Subject 3	Error	0.279	**0.167**	0.251
	Grasp Error	0.375	0.257	**0.231**
Subject 4	Error	0.188	0.187	**0.142**
	Grasp Error	0.240	**0.145**	0.220
Subject 5	Error	0.244	0.307	**0.173**
	Grasp Error	0.363	0.402	**0.269**
Mean	Error	0.213	0.240	**0.179**
	Grasp Error	0.304	0.323	**0.228**

Table 6.4: Prediction error across different networks for stick grasping experiment (best results in bold).

6.8 Experiment With Tasks That Require Grasping

6.8.1 Experimental Setup

During this experiment, subjects were asked to perform various tasks with no restrictions on their choice of grasp type or location. Each task involved interactions with multiple bottles in the scene. A different set of five naïve subjects were recruited for this experiment. There were 20 different tasks and the total trial time per subject was around 12 minutes for a total of 90,029 frames (roughly 62 minutes). Examples of tasks includes rotating bottles in place, rolling bottles on table, standing bottles upside down, and stacking bottles. The full list of tasks is provided below.

- Pick up to observe bottles.
- Rotate bottles in place.
- Swap bottle locations.
- Roll bottles on table.
- Stand bottles upside down.
- Stack bottles on top of each other.
- Stack bottles upside down on top of each other.

- Sort bottles from shortest to tallest.
- Sort bottles from tallest to shortest.
- Stand bottles upside down with labels facing away.
- Roll one bottle on the table and immediately roll another so that they collide.
- Shake bottles.
- Lean every other bottle on the one to the left of it.
- Lean every other bottle on the one to the right of it.
- Lay all bottles flat on the table.
- Move every other bottle closer.
- Pretend to pour out the contents of each bottle.
- Count the number of grooves on the bottom of each bottle.
- Read labels to determine which name has the most letters.
- Read labels to determine which drink has the most calories.

Two more bottles were added to the experiment in addition to the four shown in figure 6.1. All six bottles were present in the scene simultaneously, and all tasks involve interacting with each bottle at least once. Because the motions were more fluid, grasps were defined as beginning when the hand starts reaching for the object after it had completely moved away from the previous one. Grasps were defined as ending when the hand establishes a firm grasp on the target object. The labeling of ground truth was done manually.

6.8.2 Experiment With Tasks That Require Grasping Results

Grasps were divided manually into grasp types based on hand shape and location on the bottle. After we labeled unique grasps, we were surprised to find 39 different grasps given that they were still interacting with only bottles. Out of the 39 grasps, we filtered out uncommon grasps with less than 75 frames across the entire training set and were left with 23 grasps (figure 6.20).

Next, we tried to predict between the 23 common grasps without success. After examining our results and confusion matrices, we found that many of the grasps were similar grasps at different locations or bottle orientations. We ran a spectral co-clustering [Dhillon,

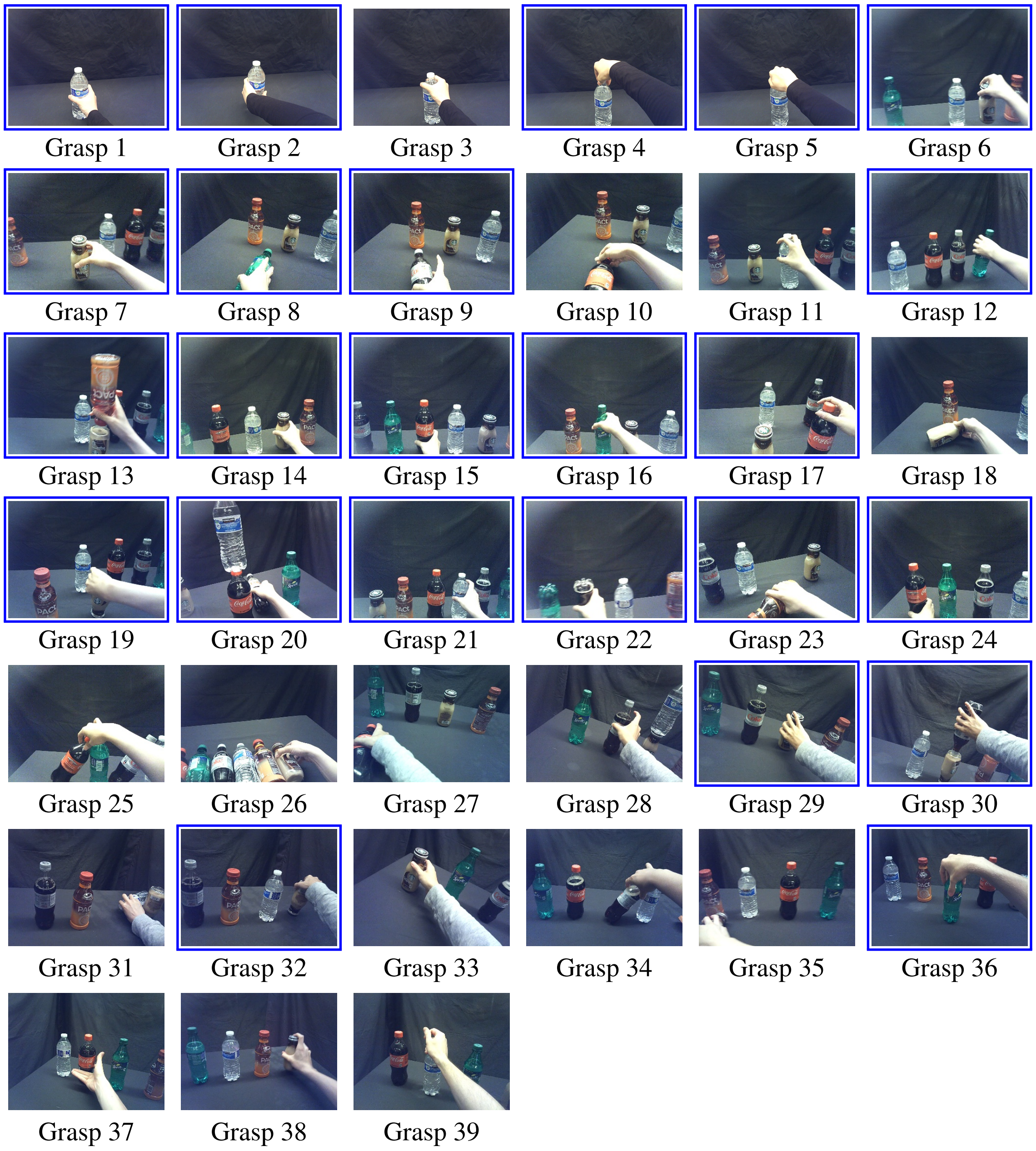
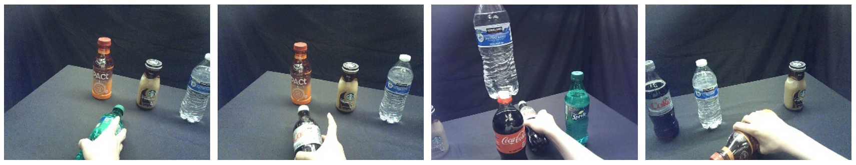


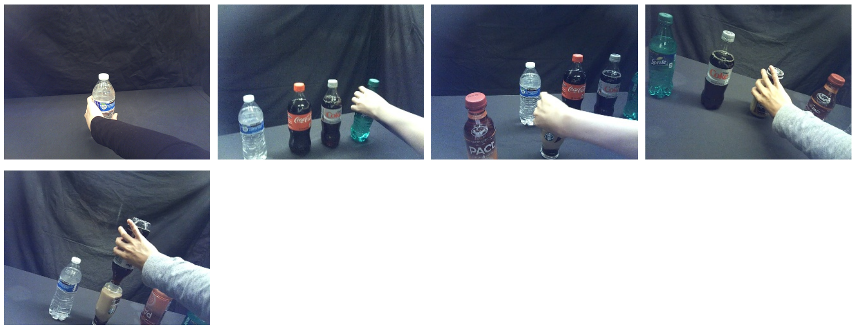
Figure 6.20: Grasps found in the general experiment. The ones highlighted in blue are the grasps we kept after pruning out the outliers.

(a) Body grasps.

(b) Top grasps.



(c) Lying grasps.



(d) Backhand grasps.

Figure 6.21: Grasp groups found by spectral co-clustering.

2001] on our confusion matrices to group the 23 common grasps into four different groups shown in figure 6.21.

We tested the prediction results using the four different groups and a non-grasp classification for a total of 5 possible classifications. We used the same neural networks as in section 6.5, but with a gaze position sequence length of 13 in order to better match the shorter duration of grasps, and learning rate of $1e{-}4$ for combined neural network.

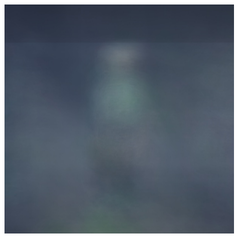
		Hand Image Neural Network	Gaze Neural Network	Combined Neural Network
Subject 1	Error	0.249	0.559	**0.226**
	Grasp Error	**0.515**	0.632	0.567
Subject 2	Error	**0.199**	0.521	0.217
	Grasp Error	0.393	0.659	**0.365**
Subject 3	Error	**0.204**	0.522	0.209
	Grasp Error	0.529	0.705	**0.495**
Subject 4	Error	0.207	0.551	**0.202**
	Grasp Error	**0.532**	0.661	0.626
Subject 5	Error	**0.162**	0.521	0.170
	Grasp Error	0.565	0.625	**0.562**
Mean	Error	**0.204**	0.535	0.205
	Grasp Error	**0.507**	0.656	0.523

Table 6.5: Prediction error across different networks for general experiment (best results in bold).

Table 6.5 shows prediction results across the four grasp groups using the modified neural network. Even though all networks perform better than random chance error which is 0.8, there were no improvements for this experiment when gaze information was added. When we performed a qualitative gaze analysis on the four grasp groups (similar to figures 6.4 to 6.8), we found no distinct gaze patterns which explains why gaze is not improving prediction results (figure 6.22). We attribute the lack of distinct gaze pattern to the subjects being more focused on the tasks than their grasps. Depending on the task, subjects' gaze was fixated on key task-related locations such as the contact point between two bottles (figures 6.23a and 6.23b). We believe this behavior is related to the inspection actions found in chapter 5. For some other tasks such as shaking the bottle where maintaining a firm grasp on the bottle is crucial, we did find that subjects fixate on the grasp location (figure 6.23c). It seems that during certain tasks, observing the completion of task takes priority over maintaining a firm grasp on the object which can also be checked through peripheral vision, proprioception, and tactile feedback.

(a) Average gaze for body grasps.

(b) Average gaze for top grasps.


(c) Average gaze for lying grasps.

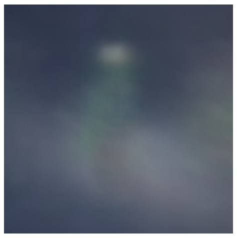
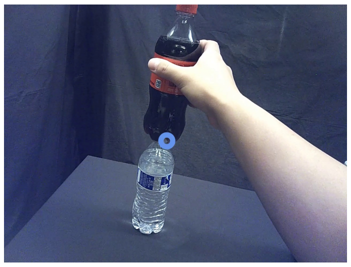
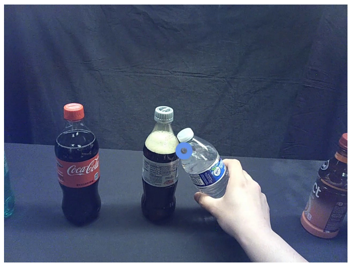

(d) Average gaze for backhand grasps.

Figure 6.22: Average gaze during tasks that require grasping.


(a) Subject fixates in between the two bottles while stacking them.


(b) Subject fixates in between the two bottles while leaning one on another.

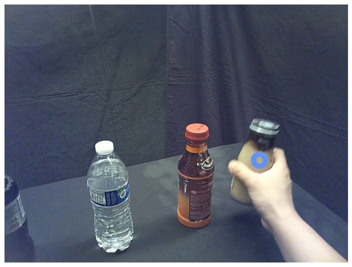

(c) Subject fixates on the grasp location while shaking the bottle.

Figure 6.23: Gaze behavior during tasks that require grasping.

6.9 Discussion

Across the three experiments, we have observed cases in where gaze improves prediction of grasp types, gaze improves prediction of grasp locations, and gaze has no effect in improving prediction results. Our neural network structure assumes that gaze position with respect to the target object has a direct relation to the grasp type or location to be used. This assumption held for the first two experiments where subjects were asked to do simple tasks such as grasping a bottle using a certain grasp or grasping a stick at a certain location. On the other hand, we observed in the third experiment that gaze is not directly related to grasps during tasks that are more complex such as stacking a bottle on top of another. During these tasks, inspecting the result of the manipulation took priority over fixating at the grasp targets. However, gaze still holds valuable information about people's intentions and should not be dismissed. For these more complex tasks, we believe gaze should be used to extract contextual information rather than being fed in directly to predict grasps. Then, the contextual information should be used to predict grasps that are relevant to that context.

Our work shows that adding gaze not only improves grasp type prediction results as in Rogez et al. [Rogez et al., 2015], but also retains the property that prediction is able to be made before the hand reaches the object as in Fermüller et al. [Fermüller et al., 2016].

Our results are already promising for predicting grasps in a controlled environment, but become even better when we think about how we would actually use them, for example, in virtual reality applications. Eye tracking brings numerous benefits to VR such as foveated rendering, object selection using gaze, and virtual social interactions. Because of these benefits, eye tracking is an obvious integration into future VR headsets. Companies such as FOVE, Inc. are already manufacturing VR headsets with eye tracking. There is even a study showing it is possible to integrate eye tracking into Google Cardboard [Greenwald et al., 2016].

One of the main applications of our method for virtual reality would be the prediction of hand motions. By predicting user's hand motions and rendering them before we can get actual data, we would be able to reduce delays in tracking and networked interactions, and even improve hand tracking by ruling out unlikely poses ahead of time. For hand motion prediction, the most uncertainty lies in predicting when a grasp will begin, and predicting what kind of grasps will occur while the hand is still reaching for the object. Figure 6.24 shows the predicted likelihood of a grasp across different neural networks. We can see here that gaze neural network has a very precise transition in the grasp likelihood when grasp actually begins. By contrast, the hand image neural network is relatively unsure about when a grasp will begin until a few frames after the grasp has begun. The combined network inherits the benefits of using gaze information and performs similarly well for predicting when grasps will begin. We similarly observed in figure 6.16 that the combined network yields the good qualities of both inputs. By the time the hand reaches the touching point, hand information is able to distinguish grasp 1 from 2, and gaze information is able to distinguish grasp 3 from 1.

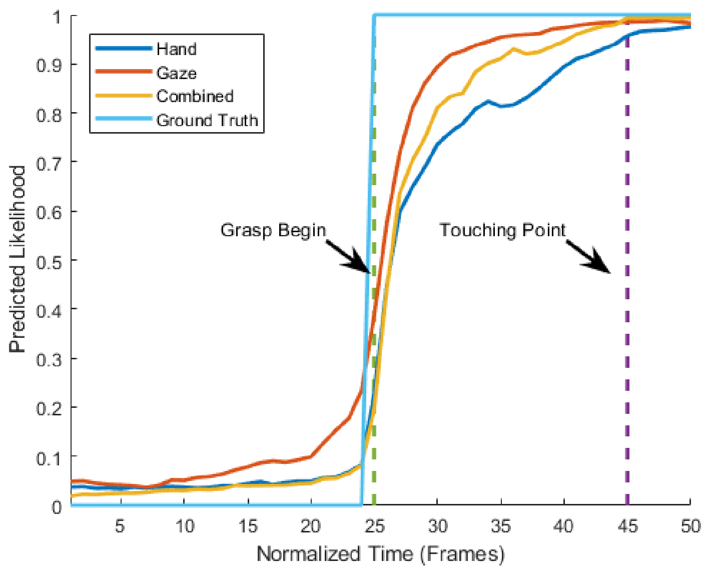


Figure 6.24: Predicted likelihood of a grasp across different networks.

6.10 Limitations and Future Work

In our work, we have successfully demonstrated the significance of gaze information when predicting grasp types in a controlled experiment. We further demonstrated that gaze is especially useful for distinguishing between grasp locations when using the same grasps in our stick grasping experiment. However, we found that further work is needed to utilize our findings in a more general setting. In particular, we observed that gaze behavior resembles inspection actions during certain tasks. During these tasks, gaze often points towards key locations of the task which may be decoupled from the grasp location. This makes it hard to predict grasp type directly from gaze. In these cases, we will need to find ways to exploit the contextual information available in gaze, then predict grasps that are relevant in this context.

Overall, we are excited to present the possibilities of improving hand motion prediction using eye tracking which will be readily available in future generation of virtual reality headsets. We hope our initial findings spark interest to further explore the hand motion cues available from eye tracking.

Chapter 7

Grasp Type Prediction in Virtual Reality

Following our success in grasp type prediction experiments in a laboratory setting (chapter 6), we have implemented a similar experiment in virtual reality (VR) to show the helpfulness of gaze information in a setup more relevant to real applications.

In the VR experiment, FOVE head mounted display ([Hom]) was used to display the virtual environment and to track the user's eye movements. In order to track the user's hand, a Leap Motion ([Lea]) was attached to the FOVE headset. To minimize interference between infrared lights between Leap Motion and FOVE's positional tracking system, the Leap Motion was mounted underneath the FOVE headset (figure 7.2). Leap Motion SDK's rigged hand was used (figure 7.1).

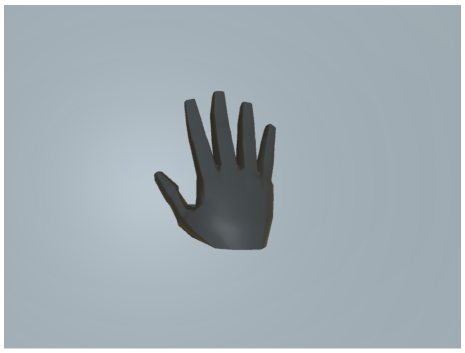


Figure 7.1: Hand model in virtual reality experiment.

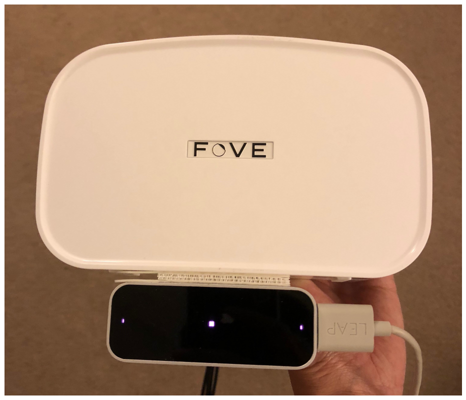


Figure 7.2: Leap Motion was attached underneath the FOVE headset.

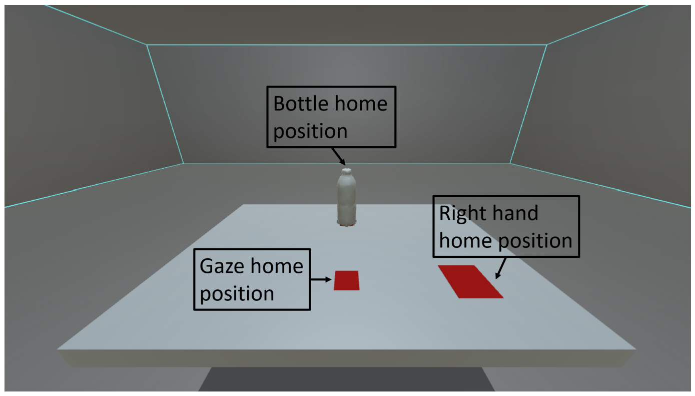
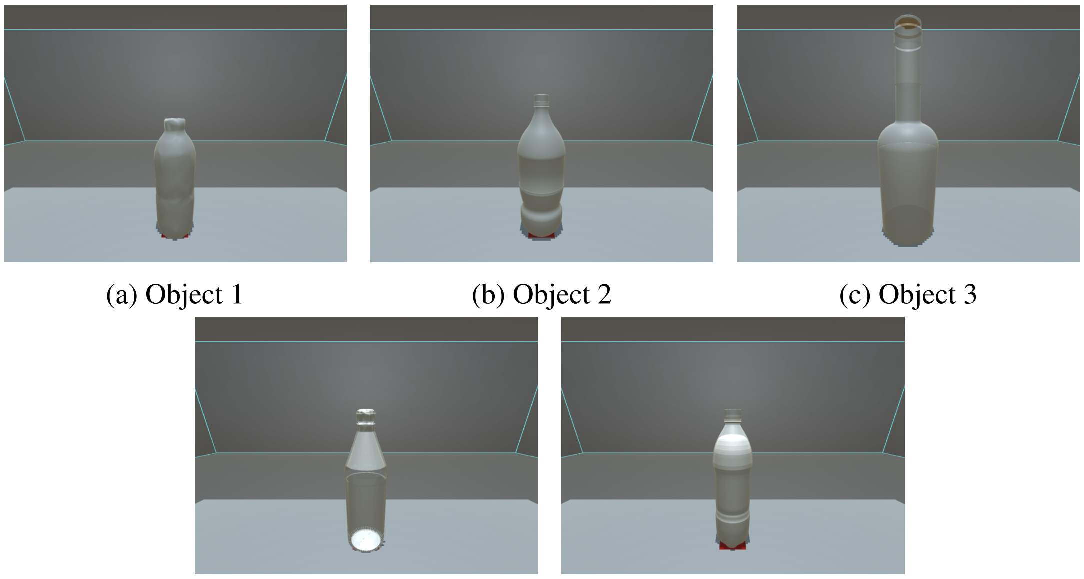


Figure 7.3: Home positions for grasp type prediction in virtual reality experiment.

7.1 Experimental Setup

The experimental setup was kept similar to the experiment in section 6.2. As in the previous experiment, there were designated home positions for the object, gaze, and hand (figure 7.3). The subjects were required to rest their right hand and gaze at their respective home positions before the start of each grasp. Then, they were asked to perform a random grasp out of the five predefined grasps in figure 6.2. This set of five grasps was repeated five times for the same object, for a total of 25 grasps. There were five different objects in the experiment (figure 7.4).

One difference from the real-life experiment was that the subjects were instructed to pick up the bottle briefly and let go in midair without worrying about resetting the bottle in the original position. It was possible to reset the bottle programmatically in virtual reality



(a) Object 1 (b) Object 2 (c) Object 3

(d) Object 4 (e) Object 5

Figure 7.4: Objects used in the VR experiment.

so that subjects can focus more on the grasping behavior.

A total of 15 subjects were recruited for the VR experiment. After manual inspection, some of the grasps were removed according to the rejection criteria below.

- Subject took more than two tries to grasp the object.
- Hand tracking was unstable and yielded an orientation jump larger than 45 degrees.
- Subject made a mistake and performed the wrong grasp.
- Bottle starting location was glitched and not reset correctly.

Total data size after removing invalid grasps was 319,985 frames captured at 30 fps. Breakdown of number of frames obtained for each grasp is shown in table 7.1. An example of a subject performing a body normal grasp is shown in figure 7.5.

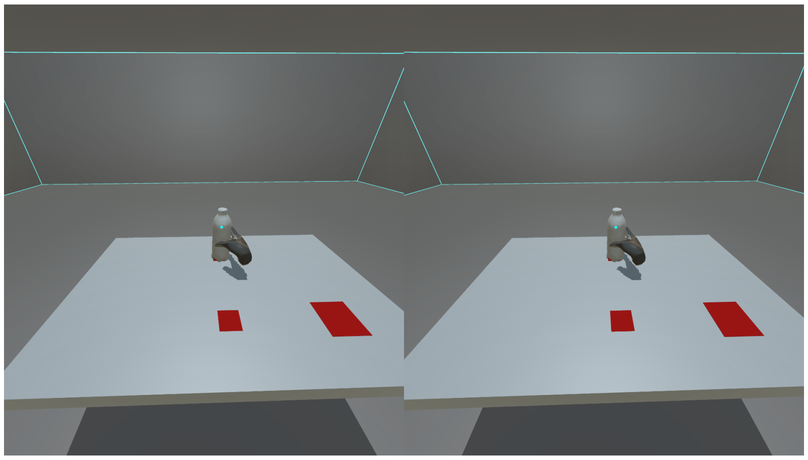


Figure 7.5: Sample of body normal grasp being performed by a subject.

Grasp 1	Grasp 2	Grasp 3	Grasp 4	Grasp 5	Non-Grasp	Total
29258	27318	30411	33067	31107	168824	319985

Table 7.1: Number of frames obtained for each grasp in the VR experiment.

7.2 Methods

We used neural networks to predict the grasp type as in section 6.5. However, in the VR setup, we are able to obtain the hand pose directly from Leap Motion. Therefore, we used the positions of various locations in the hand directly rather than using a hand image input which indirectly provides the same information. This hand position vector of size 24 includes wrist position, palm position, palm position offset by a unit vector in the direction the palm is facing, and fingertip positions of all five fingers. The gaze input was obtained in a similar manner to section 6.5, except it was obtained as a 3D position rather than 2D.

The reduced dimension of the hand feature also allowed us to utilize long short-term memory (LSTM) units for the entire input. A diagram for the overall neural network is shown in figure 7.6.

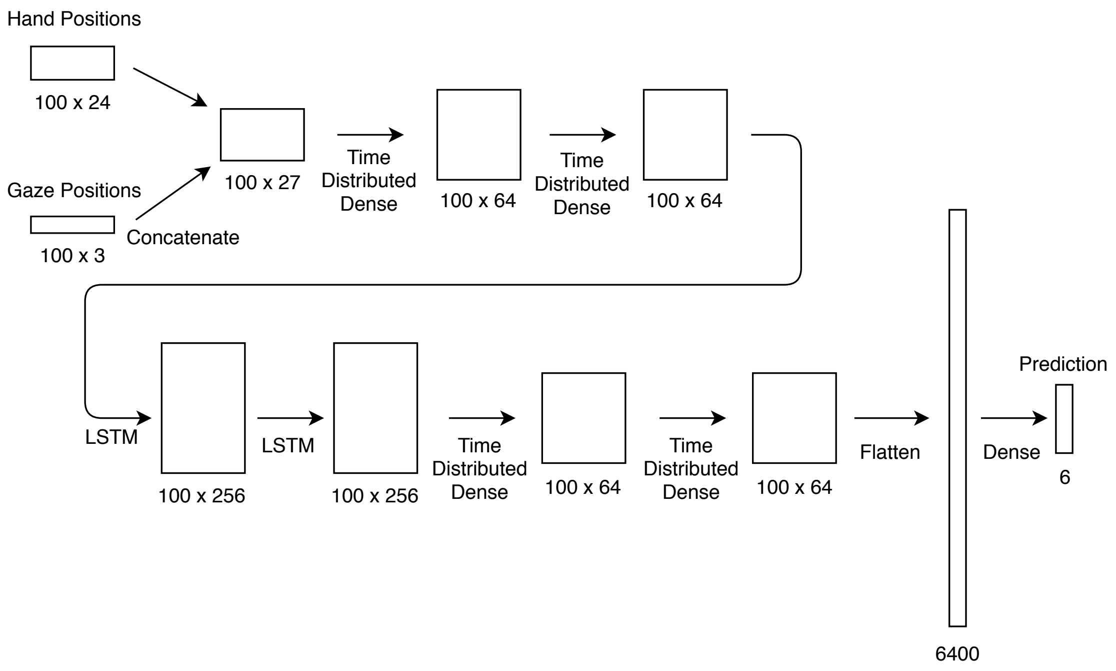


Figure 7.6: VR prediction neural network. The numbers denote the dimensions of the vectors going through each layer.

7.3 Results

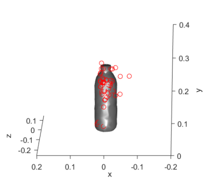
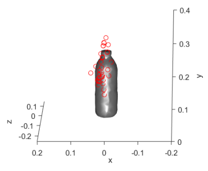
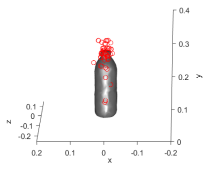
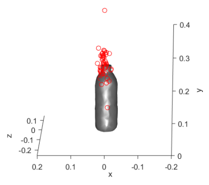
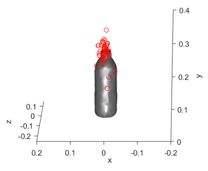
Qualitatively, we observed a similar behavior as in section 6.4. For each different grasp, most subjects fixated on the target location of the grasp before the hand reached the object. Figure 7.7 shows the subjects' fixation on characteristic locations one third of the way into the grasp. Across all the bottles, we can observe that the subjects fixate on the target location of the grasp (body for body grasps, neck for neck normal grasp, and top for top grasps), and these locations vary over the different bottle shapes. One particularly interesting pattern was for figure 7.7m. Because this object has an elongated neck, the subject's gaze pattern for neck normal grasp (grasp 3) was spread all over the neck.

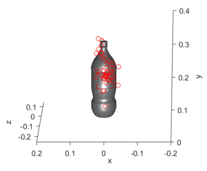
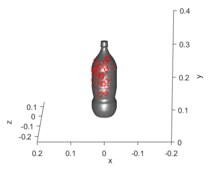
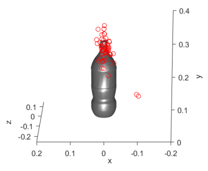
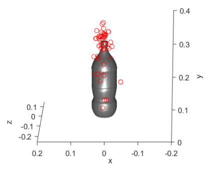
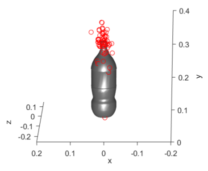
We performed a leave-one-subject-out cross validation for five of the subjects recruited in the experiment to verify that gaze information still lowers prediction error in VR environment. The prediction error is reported in table 7.2. It is interesting to note that subjects 2, 3, and 4 had lowest grasp error with hand position neural network, but lowest overall error with combined neural network. This implies that for these particular subjects, combined neural network was better at detecting when a grasp was not happening than at detecting which grasp was actually happening. However, combined neural network has the best overall performance across all five subjects. We found the resulting improvement in the combined neural network over the hand position neural network to be statistically significant with t-test's p-value being less than 0.001. This value was obtained by comparing the distribution of predicted likelihood corresponding to the ground truth grasp types at each frame.

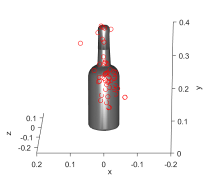
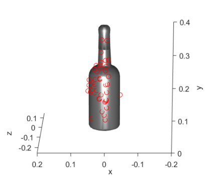
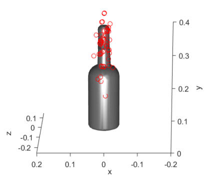
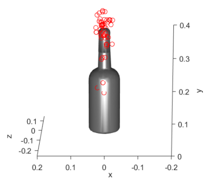
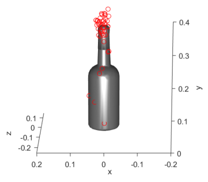
7.4 Discussion

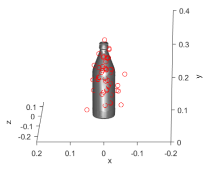
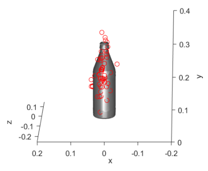
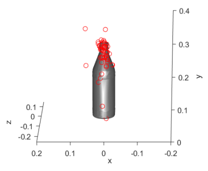
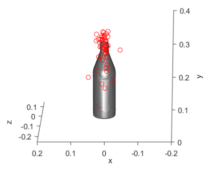
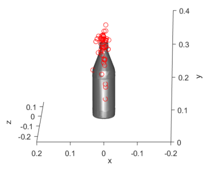
For the VR experiment, it was more important to provide feedback to the user that they have successfully grasped the object than in the real-life experiment, because they do not receive a tactile feedback. Users rely more heavily on their visual feedback to observe the task completion. Therefore, we have decided to make the bottle fade away upon the completion of a grasp. Also, unlike in the real-life experiment where someone must physically reset the objects to their original position, in the VR experiment, it was possible to reset the objects programmatically so that the users can focus more on the grasping tasks.

One particular challenge in our setup was that the Leap Motion device does not provide perfect hand tracking, especially when there were self-occlusions among the fingers and the palm. Besides surprising our users when this happened, the tracking errors meant that some grasps had to be manually filtered out according to the rejection criteria outlined in


(a) Grasp 1

(b) Grasp 2

(c) Grasp 3

(d) Grasp 4

(e) Grasp 5


(f) Grasp 1

(g) Grasp 2

(h) Grasp 3

(i) Grasp 4

(j) Grasp 5


(k) Grasp 1

(l) Grasp 2

(m) Grasp 3

(n) Grasp 4

(o) Grasp 5


(p) Grasp 1

(q) Grasp 2

(r) Grasp 3

(s) Grasp 4

(t) Grasp 5

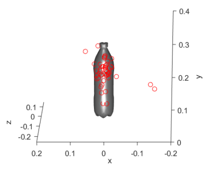
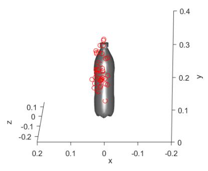
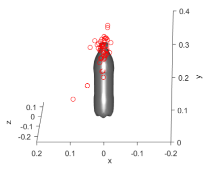
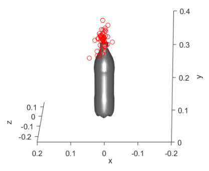
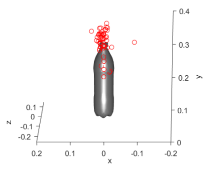

(u) Grasp 1

(v) Grasp 2

(w) Grasp 3

(x) Grasp 4

(y) Grasp 5

Figure 7.7: Average gaze for virtual reality experiment.

		Hand Position Neural Network	Gaze Neural Network	Combined Neural Network
Subject 1	Error	0.310	0.430	**0.235**
	Grasp Error	0.493	0.691	**0.328**
Subject 2	Error	0.309	0.425	**0.308**
	Grasp Error	**0.468**	0.644	0.483
Subject 3	Error	0.271	0.404	**0.251**
	Grasp Error	**0.431**	0.632	0.502
Subject 4	Error	0.345	0.389	**0.248**
	Grasp Error	**0.344**	0.670	0.374
Subject 5	Error	0.333	0.416	**0.285**
	Grasp Error	0.605	0.724	**0.513**
Mean	Error	0.314	0.413	**0.266**
	Grasp Error	0.468	0.672	**0.440**

Table 7.2: Prediction error in leave-one-subject-out cross validation (best results in bold).

section 7.1. We saw many instances where users took two or more tries to successfully grasp the object in VR. In the real-life experiment, subjects were always able to grasp the bottle in their first try.

Albeit with these difficulties, we found no significant deviation of gaze behavior in VR from the real-life experiment. Our results confirm that gaze is still helpful for improving grasp type prediction accuracy in VR.

Chapter 8

Grasp Type Prediction User Experience Study in Virtual Reality

In the virtual reality grasp type prediction experiment, we noticed many users having trouble grasping bottles in the desired manner due to instability in hand tracking and physics simulation. As a proof of concept to see if we can remedy the situation using our prediction results, we conducted a user study to compare and contrast bottle grasping behavior with and without grasp assistance based on prediction.

8.1 Experimental Setup

Scene layout was kept the same as in chapter 7. However, we conducted two versions of the grasping study per subject: one with grasp assistance based on prediction (blue set) and one without (red set). In the blue set, subject's hand tracking was replaced with pre-animated hand motion based on the predicted grasp type in order to mitigate the effect of unstable hand tracking. The order in which a subject experiences grasp assistance and order of objects within a version was randomized in order to study the learning effect of how much faster the subjects learn to successfully grasp the objects. After completing each version of experiment, the subjects were asked to fill out a survey which asks them about their perception of that version. The questionnaire is shown in figure 8.1. A total of ten users participated in the study.

In order to keep the focus of the study on the effects of being able to use clean pre-animated hand poses based on accurate predictions, we have simplified the possible grasp types to three grasps that were able to be distinguished clearly. We used body normal grasp (grasp 1), body backhand grasp (grasp 2), and top precision grasp (grasp 4) in the user study.

8.2 Results

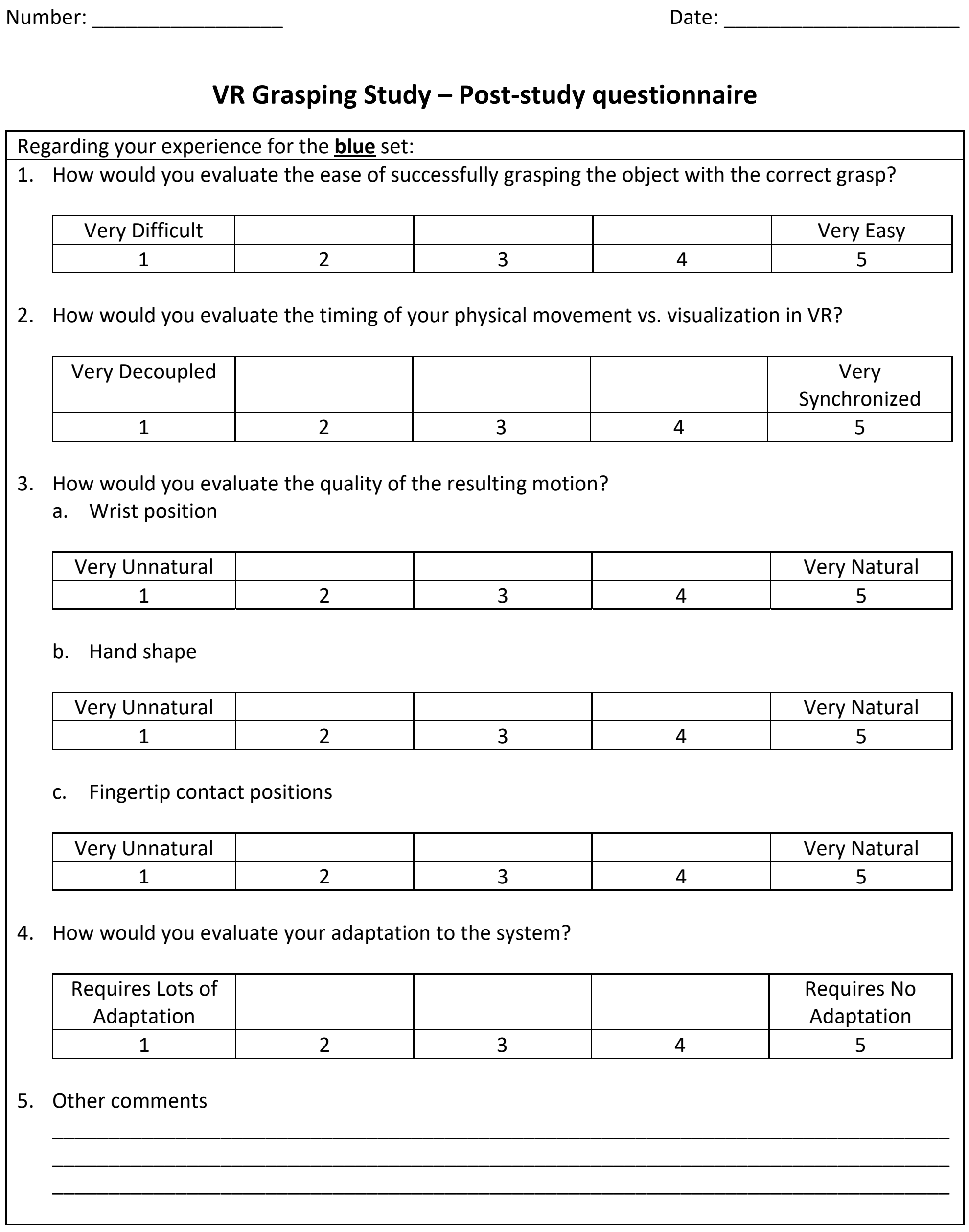
The red set was the vanilla system that had the same setup as the one in chapter 7. It had occasional tracking error from the Leap Motion device when hand parts were self-occluded. When users noticed the tracking error, they adapted to the system by adjusting their hand pose slightly or by trying again. The blue set featured grasp assistance based on grasp type prediction. Because the blue set used pre-animated grasps, the users had to adapt to these animated grasps that were indirectly driven by their actions. Also, sometimes the prediction was incorrect, in which case, the users had to adapt to the system by adjusting their hand pose slightly or by trying again.

In order to compare how quickly users adapted to both systems, we have measured the time it takes for users to complete the grasp using each of the system. The order in which the sets and the five objects within each set were presented to the users were randomized. The plots of average grasp times over trials are shown in figure 8.2. We observed learning effects in both sets with the blue sets mean grasp time decreasing more rapidly. Subjects start out taking longer than the overall mean grasping time in the beginning, but eventually are able to adapt to both systems and achieve complete grasps faster than the overall mean. The overall mean grasp time was 98.58 frames for the red set and 84.81 frames for the blue set (at 30 fps). This is equivalent to a difference of 459 ms.

We also report the aggregated survey results in table 8.1. The aggregate results show that users preferred the vanilla system, although they were able to grasp faster with prediction.

Figure 8.3 shows area plot of predicted likelihood for different grasps during the user study over a normalized average grasp time of 91.69 frames where the touching point is at 91.69 frames. We can see that for all of the grasps, the prediction likelihood increases before the hand reaches the bottle and converges into the desired grasp. Out of the three grasps, grasp 2 had the lowest likelihood levels.

The overall prediction accuracy at the touching point was 0.876. Breakdown of prediction accuracy per subject is shown in figure 8.4. Among the five subjects that had higher prediction accuracy, the mean accuracy was 0.949. An analysis of normalized confusion matrix of subjects with low prediction accuracy (figure 8.5a) illustrates that these subjects had trouble getting the system to register body backhand grasp (grasp 2) and top precision



Number: ________________ Date: ____________________

VR Grasping Study – Post-study questionnaire

Regarding your experience for the **blue** set:

1. How would you evaluate the ease of successfully grasping the object with the correct grasp?

Very Difficult				Very Easy
1	2	3	4	5

2. How would you evaluate the timing of your physical movement vs. visualization in VR?

Very Decoupled				Very Synchronized
1	2	3	4	5

3. How would you evaluate the quality of the resulting motion?
   a. Wrist position

Very Unnatural				Very Natural
1	2	3	4	5

   b. Hand shape

Very Unnatural				Very Natural
1	2	3	4	5

   c. Fingertip contact positions

Very Unnatural				Very Natural
1	2	3	4	5

4. How would you evaluate your adaptation to the system?

Requires Lots of Adaptation				Requires No Adaptation
1	2	3	4	5

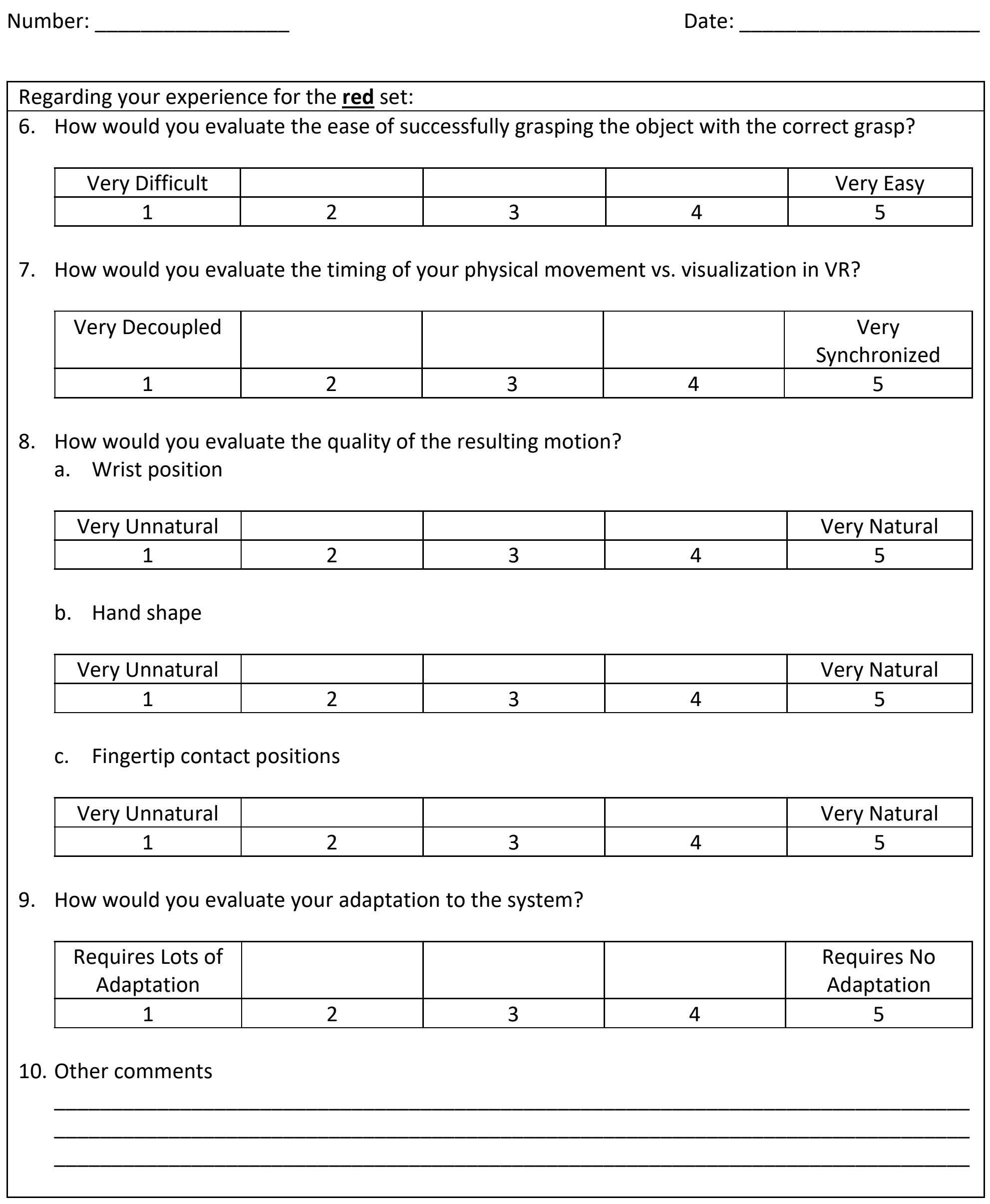
5. Other comments

____

____

____

Figure 8.1: Questionnaire provided to the subjects of virtual reality user study experiment.



Number: ________________ Date: ____________________

Regarding your experience for the **red** set:

6. How would you evaluate the ease of successfully grasping the object with the correct grasp?

Very Difficult				Very Easy
1	2	3	4	5

7. How would you evaluate the timing of your physical movement vs. visualization in VR?

Very Decoupled				Very Synchronized
1	2	3	4	5

8. How would you evaluate the quality of the resulting motion?
   a. Wrist position

Very Unnatural				Very Natural
1	2	3	4	5

   b. Hand shape

Very Unnatural				Very Natural
1	2	3	4	5

   c. Fingertip contact positions

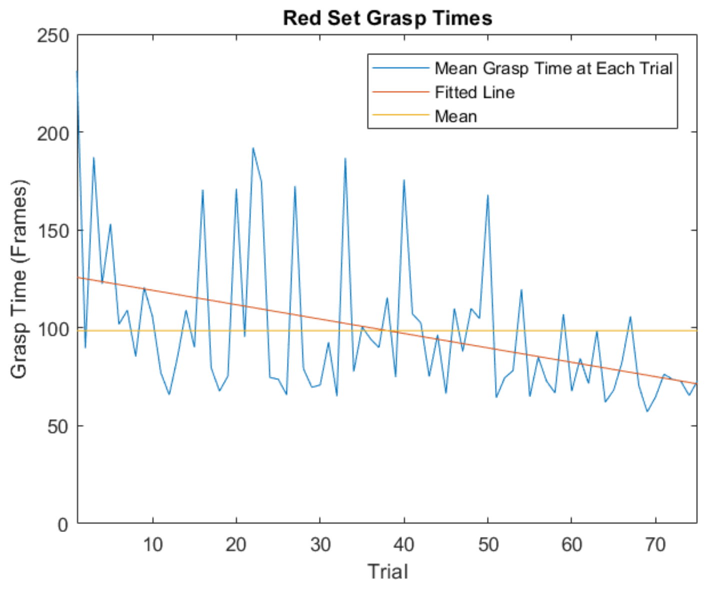
Very Unnatural				Very Natural
1	2	3	4	5

9. How would you evaluate your adaptation to the system?

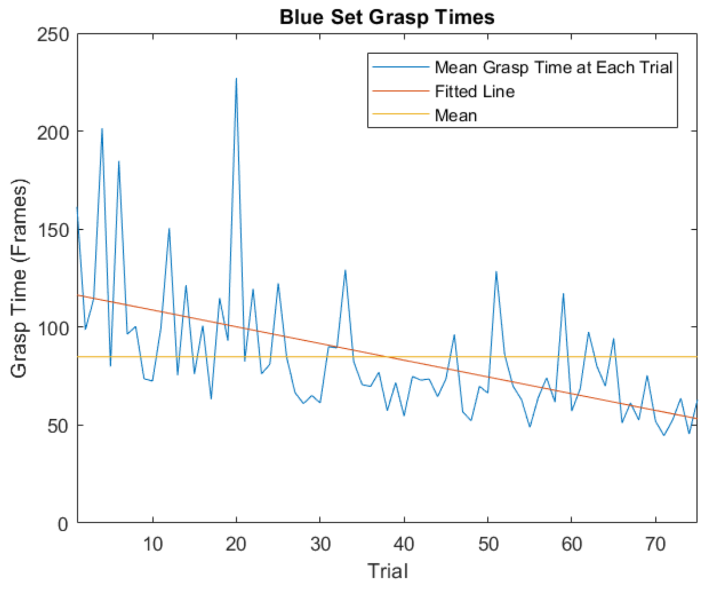
Requires Lots of Adaptation				Requires No Adaptation
1	2	3	4	5

10. Other comments
____
____
____

Figure 8.1 (Continued): Questionnaire provided to the subjects of virtual reality user study experiment.

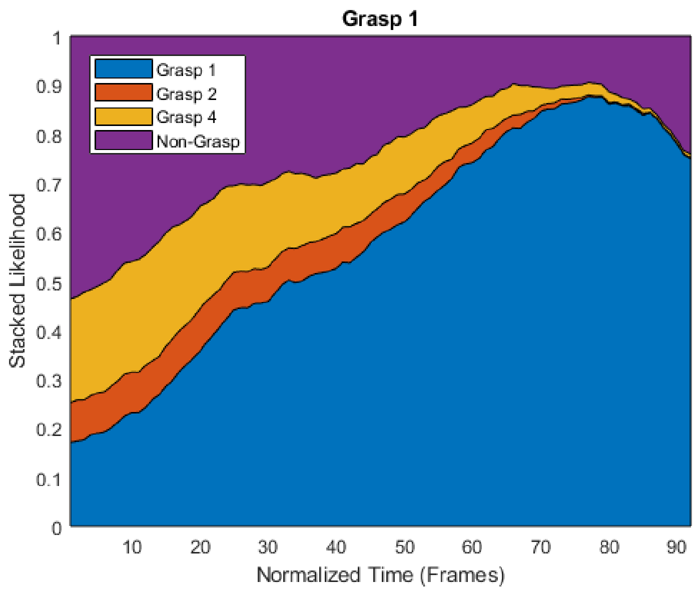


(a) Grasp time over trials for red set.

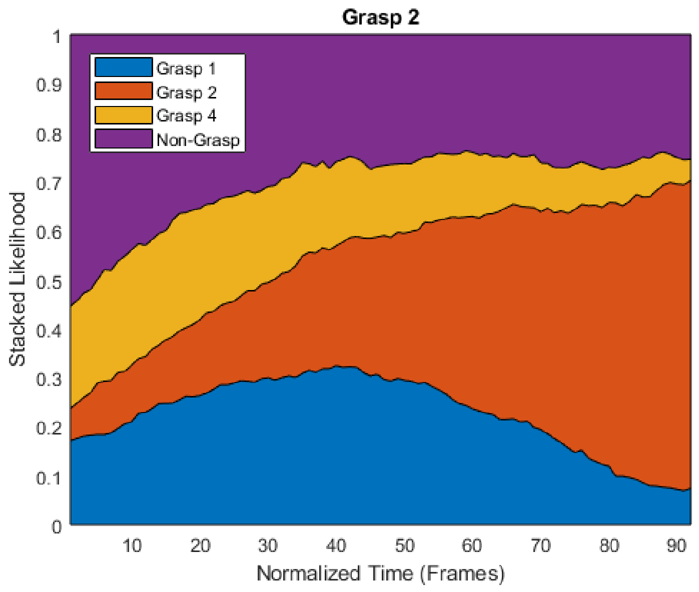


(b) Grasp time over trials for blue set.

Figure 8.2: Grasp times for the two versions of experiment.

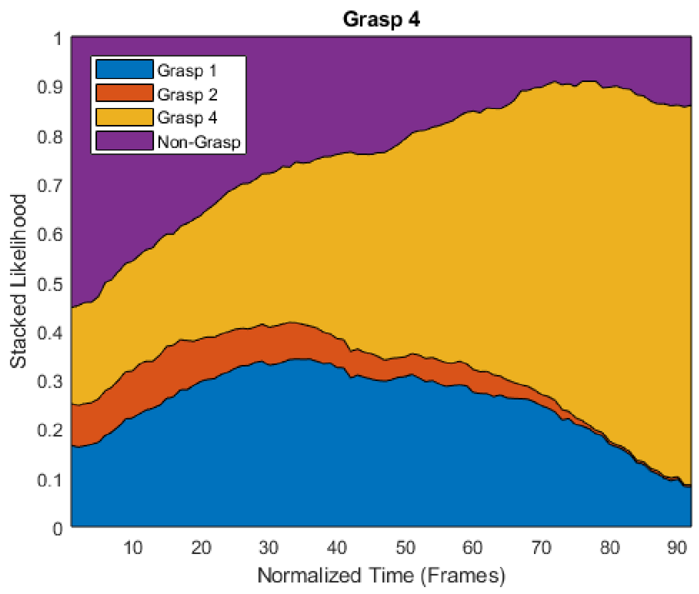


(a) Area plot of predicted likelihood during grasp 1.



(b) Area plot of predicted likelihood during grasp 2.

Figure 8.3: Area plot of predicted likelihood in the user study.



(c) Area plot of predicted likelihood during grasp 4.

Figure 8.3 (Continued): Area plot of predicted likelihood in the user study.

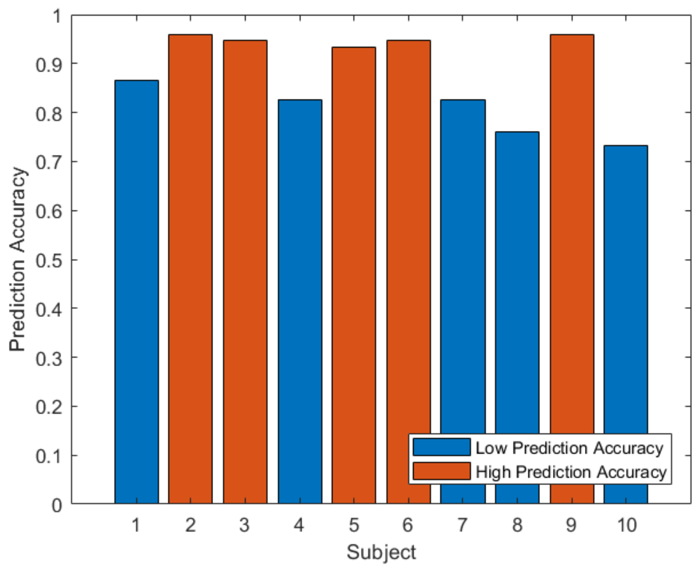
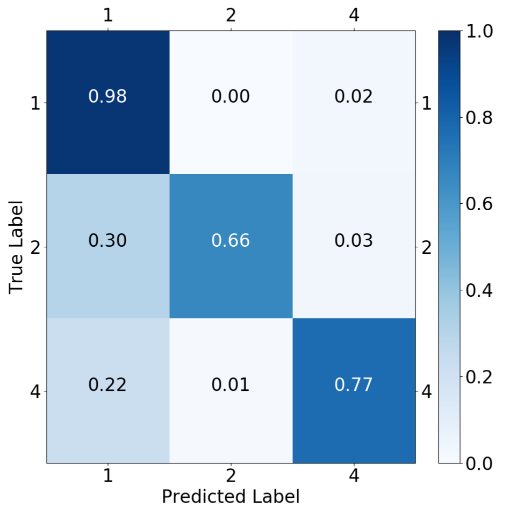
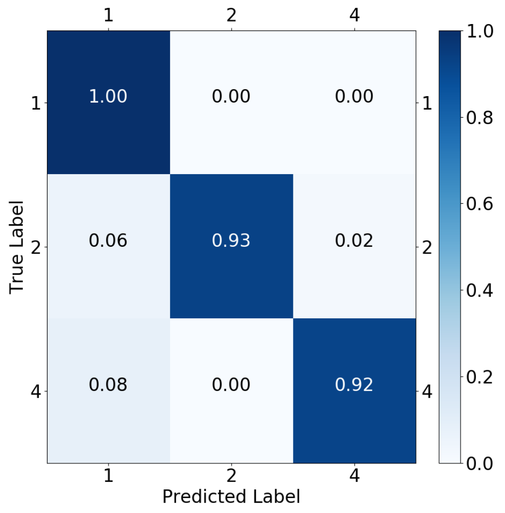


Figure 8.4: User study prediction accuracy per subject.



True \ Predicted	1	2	4
1	0.98	0.00	0.02
2	0.30	0.66	0.03
4	0.22	0.01	0.77

(a) Normalized confusion matrix for subjects with low prediction accuracy.



True \ Predicted	1	2	4
1	1.00	0.00	0.00
2	0.06	0.93	0.02
4	0.08	0.00	0.92

(b) Normalized confusion matrix for subjects with high prediction accuracy.

Figure 8.5: Normalized confusion matrices for the user study. Label numbers correspond to grasp types.

grasp (grasp 4). On the other hand, the subjects with high prediction accuracy did not have much trouble performing these grasps (figure 8.5b). In both cases, most of the wrong predictions were predicted as body normal grasp (grasp 1).

8.3 Discussion

In this user study, we showed a proof of concept for potential uses of grasp type prediction. We found that users were able to adapt to the prediction based grasping system over time, and achieved a lower mean grasp time compared to the system which directly uses hand tracking.

However, there are areas to be improved in the future. Although users were able to adapt to the prediction system, they rated it lower than the vanilla system. This seems to be because the prediction system did not always yield the correct prediction. The overall prediction accuracy was 0.876, which could be improved in the future.

Also, pre-animated versions of grasps sometimes diverged from user's actions. In the future, if an online pose generation method such as work by Zhao et al. [Zhao et al., 2013]

Set	Ease of successfully grasping	Timing of physical movement vs. visualization in VR	Quality of wrist position	Quality of hand shape	Quality of fingertip contact positions	Ease of adaptation to the system
Red	**3.8**	**4.5**	**3.9**	**3.7**	**3.9**	**4.2**
Blue	3.3	3.4	3.3	3.4	3.2	3.7

Table 8.1: User study survey results (better results are in bold).

is adopted, we would be able to generate clean motions that closely matches the timing of user's actions with a prior on the predicted grasp type.

Chapter 9

Conclusion

We have detailed two frameworks that address various challenges of hand tracking and take us closer towards being able to use our hands directly to interact with virtual objects. The hand pose estimation framework showed that quadratic encoding model can significantly ease the problem of hand tracking by being able to estimate the whole hand pose when fingertips are reliably tracked. The hand pose prediction framework showed that use of eye tracking improves grasp type prediction which can then be used to generate hand pose.

Along the way, we shed light on some important properties of hand motion. First, chapter 4 showed hand motion is nonlinear, but because the range of natural human finger motions is quite limited, an extremely simple nonlinear function such as a quadratic function can approximate the inverse transformation from the fingertip positions to the joint angles. This nature of reduced dimensionality of hand motion space is the same principle that guided various works in section 3.1.3, but our work showed that depending on the desired motion space and precision, an extremely simple and fast model such as a quadratic function can be viable. Next, chapter 5 confirmed findings in previous biological studies that gaze leads hand movements, and it often gives cue for position and timing of the action. We also observed that tools held effectively become an extension of the hand and shifts the fixation of the gaze toward the tool tip. However, gaze behavior during inspection actions were not directly related to hand movements, and these actions should be treated differently when inferring hand motion from gaze data. Then, in chapters 6 and 7, we successfully augmented gaze information to improve grasp type prediction accuracy in both real-life and virtual reality (VR) settings. We were able to show that for the five different grasp types we experimented with, gaze patterns aggregated over multiple subjects converge to target locations of the grasp. This gaze pattern was shown to improve grasp type prediction results in early stages of grasp where hand information alone was too ambiguous to distinguish. Finally, we concluded with a user study in chapter 8 which show potential for using grasp type prediction in VR applications.

We hope our findings contribute a step towards a future of immersive virtual environments where users not only perceive the virtual environment directly through a head-mounted display, but also convey their intent directly using full hand movement (figure 1.2).

Chapter 10

Future Work

There are many potential extensions to our work that and some of our thoughts are presented here. For example, it could be possible to combine the proposed method for estimating hand pose with the method for prediction. By tracking and predicting only parts of the hand that are reliably tracked and estimating the entire hand after prediction has been performed, prediction can be performed in a lower dimensional space. This not only reduces computation, but also enables prediction when not all of the hand can be tracked.

Another extension of our work could be to expand the space of hand manipulation actions beyond 5 discrete grasps. In section 6.8, we found 39 different grasps used for interacting with bottles. This shows that the space of hand poses is really a continuous space which may be possible to constrain based on the gaze. We can use generative methods such as conditional variational autoencoders [Walker et al., 2016, Fragkiadaki et al., 2017] to learn to predict in the continuous domain.

Even though we have some methods for estimating and predicting hand motion, we still need a good physics engine to simulate predictable behavior for virtual interactions. Predictable behavior is important, because users expect to see small changes in initial condition yield predictable changes in the output [Chung and Pollard, 2016]. In the future, it would be interesting to see how well predictability of a physics engine affects virtual interactions with hands.

Chapter 11

Bibliography

3Gear Systems. `http://www.threegear.com`. Accessed: 2014-01-21. 4.5.2

Home - fove eye tracking virtual reality headset. `https://www.getfove.com/`. (Accessed on 07/24/2018). 7

Leap Motion. `https://www.leapmotion.com`. Accessed: 2014-01-21. 4.5.2, 7

TUIO. `http://tuio.org`. Accessed: 2014-01-21. 4.4

Ijaz Akhter, Tomas Simon, Sohaib Khan, Iain Matthews, and Yaser Sheikh. Bilinear spatiotemporal basis models. *ACM Transactions on Graphics*, 31(2):17:1–17:12, April 2012. doi: 10.1145/2159516.2159523. 3.2.11

Irene Albrecht, Jörg Haber, and Hans-Peter Seidel. Construction and animation of anatomically based human hand models. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 98–109. Eurographics Association, 2003. 3.1.1

Y. Bai and C. K. Liu. Dexterous manipulation using both palm and fingers. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1560–1565, May 2014. doi: 10.1109/ICRA.2014.6907059. 3.1.1

Yunfei Bai, Wenhao Yu, and C. Karen Liu. Dexterous manipulation of cloth. In *Proceedings of the 37th Annual Conference of the European Association for Computer Graphics*, EG '16, pages 523–532, Goslar Germany, Germany, 2016. Eurographics Association. doi: 10.1111/cgf.12852. URL `https://doi.org/10.1111/cgf.12852`. 3.1.1

Luca Ballan, Aparna Taneja, Juergen Gall, Luc Van Gool, and Marc Pollefeys. Motion capture of hands in action using discriminative salient points. In *European Conference on Computer Vision (ECCV)*, Firenze, October 2012. 3.1.2

Gedas Bertasius, Hyun Soo Park, Stella X. Yu, and Jianbo Shi. First person action-object detection with egonet. *Robotics: Science and Systems*, XIII, 2017. URL `http://arxiv.org/abs/1603.04908`. 3.2.5

Matthew Brand and Aaron Hertzmann. Style machines. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 183–192, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co. ISBN 1-58113-208-5. doi: 10.1145/344779.344865. URL `http://dx.doi.org/10.1145/344779.344865`. 3.2.11

Alan Bränzel, Christian Holz, Daniel Hoffmann, Dominik Schmidt, Marius Knaust, Patrick Lühne, René Meusel, Stephan Richter, and Patrick Baudisch. Gravityspace: Tracking users and their poses in a smart room using a pressure-sensing floor. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 725–734, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-1899-0. doi: 10.1145/2470654.2470757. URL `http://doi.acm.org/10.1145/2470654.2470757`. 3.1.3

Frank Broz, Illah Nourbakhsh, and Reid Simmons. Planning for humanrobot interaction in socially situated tasks. *International Journal of Social Robotics*, 5(2):193–214, 2013. ISSN 1875-4791. doi: 10.1007/s12369-013-0185-z. URL `http://dx.doi.org/10.1007/s12369-013-0185-z`. 3.2.6

Melissa C Bulloch, Steven L Prime, and Jonathan J Marotta. Anticipatory gaze strategies when grasping moving objects. *Experimental brain research*, 233(12):3413–3423, 2015. 3.2.1

Minjie Cai, Kris M Kitani, and Yoichi Sato. A scalable approach for understanding the visual structures of hand grasps. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 1360–1366. IEEE, 2015. 3.2.10

Minjie Cai, Kris M Kitani, and Yoichi Sato. Understanding hand-object manipulation with grasp types and object attributes. *Proceedings of Robotics: Science and Systems, Ann Arbor, Michigan*, 2016. 3.2.10

Jinxiang Chai and Jessica K. Hodgins. Performance animation from low-dimensional control signals. In *ACM SIGGRAPH 2005 Papers*, SIGGRAPH '05, pages 686–696, New York, NY, USA, 2005. ACM. doi: 10.1145/1186822.1073248. URL `http://doi.acm.org/10.1145/1186822.1073248`. 3.1.3

A. Chan, R. Lau, and L. Li. Hand motion prediction for distributed virtual environments. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):146–159, Jan 2008. ISSN 1077-2626. doi: 10.1109/TVCG.2007.1056. 3.2.3

Lillian Y Chang, Nancy S Pollard, Tom M Mitchell, and Eric P Xing. Feature selection for grasp recognition from optical markers. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 2944–2950. IEEE, 2007. 3.1.3

Se-Joon Chung and Nancy Pollard. Predictable behavior during contact simulation: a comparison of selected physics engines. *Computer Animation and Virtual Worlds*, 27 (3-4):262–270, 2016. 10

Inderjit S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '01, pages 269–274, New York, NY, USA, 2001. ACM. ISBN 1-58113-391-X. doi: 10.1145/502512.502550. URL `http://doi.acm.org/10.1145/502512.502550`. 6.8.2

Anca Dragan and Siddhartha Srinivasa. A policy blending formalism for shared control. *International Journal of Robotics Research*, May 2013. 3.2.6

Andrew T. Duchowski, Sophie Jörg, Tyler N. Allen, Ioannis Giannopoulos, and Krzysztof Krejtz. Eye movement synthesis. In *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*, ETRA '16, pages 147–154, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4125-7. doi: 10.1145/2857491.2857528. URL `http://doi.acm.org/10.1145/2857491.2857528`. 3.2.2

George ElKoura and Karan Singh. Handrix: animating the human hand. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 110–119. Eurographics Association, 2003. 3.1.3

Chenyou Fan, Jangwon Lee, and Michael S. Ryoo. Forecasting hand and object locations in future frames. *CoRR*, abs/1705.07328, 2017. URL `http://arxiv.org/abs/1705.07328`. 3.2.3

Alireza Fathi, Yin Li, and James Rehg. Learning to recognize daily actions using gaze. *Computer Vision–ECCV 2012*, pages 314–327, 2012. 3.2.5

Cornelia Fermüller, Fang Wang, Yezhou Yang, Konstantinos Zampogiannis, Yi Zhang, Francisco Barranco, and Michael Pfeiffer. Prediction of manipulation actions. *International Journal of Computer Vision*, pages 1–17, 2016. 3.2.4, 6.9

J Randall Flanagan, Miles C Bowman, and Roland S Johansson. Control strategies in object manipulation tasks. *Current Opinion in Neurobiology*, 16(6):650 – 659, 2006. ISSN 0959-4388. doi: http://dx.doi.org/10.1016/j.conb.2006.10.005. URL `http://www.sciencedirect.com/science/article/pii/S0959438806001450`. Motor systems / Neurobiology of behaviour. 3.2.1, 5

AmaliaF. Foka and PanosE. Trahanias. Probabilistic autonomous robot navigation in dynamic environments with human motion prediction. *International Journal of Social Robotics*, 2(1):79–94, 2010. ISSN 1875-4791. doi: 10.1007/s12369-009-0037-z. URL `http://dx.doi.org/10.1007/s12369-009-0037-z`. 3.2.6

Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *International Conference on Computer Vision (ICCV)*, 2015. 3.2.11

Katerina Fragkiadaki, Jonathan Huang, Alex Alemi, Sudheendra Vijayanarasimhan, Susanna Ricco, and Rahul Sukthankar. Motion prediction under multimodality with conditional stochastic networks. *CoRR*, abs/1705.02082, 2017. URL `http://arxiv.org/abs/1705.02082`. 10

C. Garre, F. Hernndez, A. Gracia, and M. A. Otaduy. Interactive simulation of a deformable hand for haptic rendering. In *2011 IEEE World Haptics Conference*, pages 239–244, June 2011. doi: 10.1109/WHC.2011.5945492. 3.1.1

Liuhao Ge, Hui Liang, Junsong Yuan, and Daniel Thalmann. Robust 3d hand pose estimation in single depth images: From single-view cnn to multi-view cnns. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 3.1.2

J-P Gourret, Nadia Magnenat Thalmann, and Daniel Thalmann. Simulation of object and human skin formations in a grasping task. In *ACM Siggraph Computer Graphics*, volume 23, pages 21–30. ACM, 1989. 3.1.1

Scott W Greenwald, Luke Loreti, Markus Funk, Ronen Zilberman, and Pattie Maes. Eye gaze tracking with google cardboard using purkinje images. In *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology*, pages 19–22. ACM, 2016. 6.9

Henning Hamer, Juergen Gall, Raquel Urtasun, and Luc Van Gool. Data-driven animation of hand-object interactions. In *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, pages 360–367. IEEE, 2011. 3.1.3

Kris Hauser. Recognition, prediction, and planning for assisted teleoperation of freeform tasks. *Autonomous Robots*, 35(4):241–254, 2013. ISSN 0929-5593. doi: 10.1007/s10514-013-9350-3. URL `http://dx.doi.org/10.1007/s10514-013-9350-3`. 3.2.6

Mary M Hayhoe, Anurag Shrivastava, Ryan Mruczek, and Jeff B Pelz. Visual memory and motor planning in a natural task. *Journal of Vision*, 3(1):49–63, 2003. 3.2.1, 5.1

Ludovic Hoyet, Kenneth Ryall, Rachel McDonnell, and Carol O'Sullivan. Sleight of hand: perception of finger motion from reduced marker sets. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, pages 79–86. ACM, 2012. 3.1.3

Eugene Hsu, Kari Pulli, and Jovan Popović. Style translation for human motion. In *ACM SIGGRAPH 2005 Papers*, SIGGRAPH '05, pages 1082–1089, New York, NY, USA, 2005. ACM. doi: 10.1145/1186822.1073315. URL `http://doi.acm.org/10.1145/1186822.1073315`. 3.2.11

Haoda Huang, Ling Zhao, KangKang Yin, Yue Qi, Yizhou Yu, and Xin Tong. Controllable hand deformation from sparse examples with rich details. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 73–82. ACM, 2011. 3.1.1

Leslie Ikemoto, Okan Arikan, and David Forsyth. Generalizing motion edits with gaussian processes. *ACM Trans. Graph.*, 28(1):1:1–1:12, February 2009. ISSN 0730-0301. doi: 10.1145/1477926.1477927. URL `http://doi.acm.org/10.1145/1477926.1477927`. 4.5.1

Ashesh Jain, Avi Singh, Hema Swetha Koppula, Shane Soh, and Ashutosh Saxena. Recurrent neural networks for driver activity anticipation via sensory-fusion architecture. *CoRR*, abs/1509.05016, 2015. URL `http://arxiv.org/abs/1509.05016`. 3.2.6

Roland S. Johansson, Gran Westling, Anders Bckstrm, and J. Randall Flanagan. Eye-hand coordination in object manipulation. *JOURNAL OF NEUROSCIENCE*, 21(17): 6917–6932, 2001. 3.2.1, 5, 5.1

Chris Kang, Nkenge Wheatland, Michael Neff, and Victor Zordan. Automatic hand-over animation for free-hand motions from low resolution input. In Marcelo Kallmann and Kostas Bekris, editors, *Motion in Games*, volume 7660 of *Lecture Notes in Computer Science*, pages 244–253. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-34709-2. doi: 10.1007/978-3-642-34710-8_23. URL `http://dx.doi.org/10.1007/978-3-642-34710-8_23`. 3.1.3

Cem Keskin, Furkan Kiraç, Yunus Emre Kara, and Lale Akarun. Hand pose estimation and hand shape classification using multi-layered randomized decision forests. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part VI*, ECCV'12, pages 852–863, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 978-3-642-33782-6. doi: 10.1007/978-3-642-33783-3_61. URL `http://dx.doi.org/10.1007/978-3-642-33783-3_61`. 3.1.2

Junggon Kim, Nancy S Pollard, and Christopher G Atkeson. Quadratic encoding of optimized humanoid walking. In *IEEE/RAS International Conference on Humanoid Robot (ICHR)*, 2013. 3.1.4

H Kjellstrom, Javier Romero, and Danica Kragic. Visual recognition of grasps for human-to-robot mapping. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 3192–3199. IEEE, 2008. 3.1.2

Hema S Koppula and Ashutosh Saxena. Anticipating human activities using object affordances for reactive robotic response. *IEEE transactions on pattern analysis and machine intelligence*, 38(1):14–29, 2016. 3.2.6

Paul Kry, Adeline Pihuit, Adrien Bernhardt, and Marie-Paule Cani. Handnavigator: Hands-on interaction for desktop virtual reality. In *ACM symposium on Virtual Reality Software and Technology (VRST)*. ACM, October 2008. URL `http://www-evasion.imag.fr/Publications/2008/KPBC08`. 3.1.1

Paul G. Kry and Dinesh K. Pai. Interaction capture and synthesis. *ACM Trans. Graph.*, 25(3):872–880, July 2006. ISSN 0730-0301. doi: 10.1145/1141911.1141969. URL `http://doi.acm.org/10.1145/1141911.1141969`. 3.1.1

Paul G Kry, Doug L James, and Dinesh K Pai. Eigenskin: real time large deformation character skinning in hardware. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 153–159. ACM, 2002. 3.1.1

V. Kumar, Y. Tassa, T. Erez, and E. Todorov. Real-time behaviour synthesis for dynamic hand-manipulation. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6808–6815, May 2014. doi: 10.1109/ICRA.2014.6907864. 3.1.1

Michael Land, Neil Mennie, and Jennifer Rusted. The roles of vision and eye movements in the control of activities of daily living. *Perception*, 28(11):1311–1328, 1999. doi: 10.1068/p2935. URL `https://doi.org/10.1068/p2935`. PMID: 10755142. 3.2.1

Michael F. Land. Eye movements and the control of actions in everyday life. *Progress in Retinal and Eye Research*, 25(3):296 – 324, 2006. ISSN 1350-9462. doi: http://dx.doi.org/10.1016/j.preteyeres.2006.01.002. URL `http://www.sciencedirect.com/science/article/pii/S1350946206000036`. 3.2.1

Cheng Li and Kris M Kitani. Pixel-level hand detection in ego-centric videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3570–3577, 2013. 6.5.1

Yin Li, Alireza Fathi, and James M. Rehg. Learning to predict gaze in egocentric video. In *Proceedings of the 2013 IEEE International Conference on Computer Vision*, ICCV '13, pages 3216–3223, Washington, DC, USA, 2013. IEEE Computer Society. ISBN 978-1-4799-2840-8. doi: 10.1109/ICCV.2013.399. URL `http://dx.doi.org/10.1109/ICCV.2013.399`. 3.2.8

Yin Li, Zhefan Ye, and James M Rehg. Delving into egocentric actions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 287–295, 2015. 3.2.5

C Karen Liu. Synthesis of interactive hand manipulation. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 163–171. Eurographics Association, 2008. 3.1.1

C. Karen Liu. Dextrous manipulation from a grasping pose. *ACM Trans. Graph.*, 28 (3):59:1–59:6, July 2009. ISSN 0730-0301. doi: 10.1145/1531326.1531365. URL `http://doi.acm.org/10.1145/1531326.1531365`. 3.1.1

Guodong Liu, Jingdan Zhang, Wei Wang, and Leonard McMillan. Human motion estimation from a reduced marker set. In *Proceedings of the 2006 symposium on Interactive 3D graphics and games*, pages 35–42. ACM, 2006. 3.1.3

Libin Liu, Michiel Van De Panne, and KangKang Yin. Guided learning of control graphs for physics-based characters. *ACM Transactions on Graphics (TOG)*, 35(3):29, 2016. 3.2.12

M. Ma, H. Fan, and K. M. Kitani. Going deeper into first-person activity recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1894–1903, June 2016. doi: 10.1109/CVPR.2016.209. 3.2.5

Jim Mainprice, Rafi Hayne, and Dmitry Berenson. Predicting human reaching motion in collaborative tasks using inverse optimal control and iterative re-planning. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 885–892. IEEE, 2015. 3.2.12

Jim Mainprice, Rafi Hayne, and Dmitry Berenson. Goal set inverse optimal control and iterative re-planning for predicting human reaching motions in shared workspaces. *arXiv preprint arXiv:1606.02111*, 2016. 3.2.12

A. Majkowska, V. B. Zordan, and P. Faloutsos. Automatic splicing for hand and body animations. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '06, pages 309–316, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association. ISBN 3-905673-34-7. URL `http://dl.acm.org/citation.cfm?id=1218064.1218106`. 3.1.2

Kenji Matsuo, Kentaro Yamada, Satoshi Ueno, and Sei Naito. An attention-based activity recognition for egocentric video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 551–556, 2014. 3.2.5

Igor Mordatch, Zoran Popović, and Emanuel Todorov. Contact-invariant optimization for hand manipulation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '12, pages 137–144, Aire-la-Ville, Switzerland, Switzerland, 2012. Eurographics Association. ISBN 978-3-905674-37-8. URL `http://dl.acm.org/citation.cfm?id=2422356.2422377`. 3.1.1

Sara Mulatto, Alessandro Formaglio, Monica Malvezzi, and Domenico Prattichizzo. Using postural synergies to animate a low-dimensional hand avatar in haptic simulation. *IEEE Trans. Haptics*, 6(1):106–116, January 2013. ISSN 1939-1412. doi: 10.1109/TOH.2012.13. URL `http://dx.doi.org/10.1109/TOH.2012.13`. 3.1.3, 4.3.3, 4.3.3, 4.5.1

I. Oikonomidis, N. Kyriazis, and A.A. Argyros. Full dof tracking of a hand interacting with an object by modeling occlusions and physical constraints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2088–2095, 2011a. doi: 10.1109/ICCV.2011.6126483. 3.1.2

Iason Oikonomidis, Nikolaos Kyriazis, and Antonis A Argyros. Efficient model-based 3d tracking of hand articulations using kinect. In *BmVC*, volume 1, page 3, 2011b. 3.1.2

Jiazhi Ou, Lui Min Oh, Susan R. Fussell, Tal Blum, and Jie Yang. Predicting visual focus of attention from intention in remote collaborative tasks. *IEEE Transactions on Multimedia*, 10(6):1034–1045, 2008. doi: 10.1109/TMM.2008.2001363. URL `http://dx.doi.org/10.1109/TMM.2008.2001363`. 3.2.8

Vladimir Pavlovic, James M. Rehg, and John MacCormick. Learning switching linear models of human motion. In T.K. Leen, T.G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 981–987. MIT Press, 2001. URL `http://papers.nips.cc/paper/1892-learning-switching-linear-models-of-human-motion.pdf`. 3.2.11

Claudia Pérez-D'Arpino and Julie A Shah. Fast motion prediction for collaborative robotics. In *IJCAI*, pages 3988–3989, 2016. 3.2.3

Ken Pfeuffer, Benedikt Mayer, Diako Mardanbegi, and Hans Gellersen. Gaze + pinch interaction in virtual reality. In *Proceedings of the 5th Symposium on Spatial User Interaction*, SUI '17, pages 99–108, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-5486-8. doi: 10.1145/3131277.3132180. URL `http://doi.acm.org/10.1145/3131277.3132180`. 3.2.7

Lord Kenneth Pinpin, Roland S Johansson, Cecilia Laschi, and Paolo Dario. Gaze interface: Utilizing human predictive gaze movements for controlling a hbs. In *Biomedical Robotics and Biomechatronics, 2008. BioRob 2008. 2nd IEEE RAS & EMBS International Conference on*, pages 158–162. IEEE, 2008. 3.2.7

Lord Kenneth Pinpin, Daniel Fernando Tello Gamarra, Roland Johansson, Cecilia Laschi, and Paolo Dario. Utilizing gaze behavior for inferring task transitions using abstract hidden markov models. *Inteligencia Artificial*, 19(58):1–16, 2016. 3.2.5

Nancy S. Pollard and Victor Brian Zordan. Physically based grasping control from example. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '05, pages 311–318, New York, NY, USA, 2005. ACM. ISBN 1-59593-198-8. doi: 10.1145/1073368.1073413. URL `http://doi.acm.org/10.1145/1073368.1073413`. 3.1.1

Qeexo Inc. Qeexo. `http://www.qeexo.com/`. 4.5.2

Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. *arXiv preprint arXiv:1612.08242*, 2016. 6.5.2

Hans Rijpkema and Michael Girard. Computer animation of knowledge-based human grasping. *ACM SIGGRAPH Computer Graphics*, 25(4):339–348, 1991. 3.1.1

Gregory Rogez, James S. Supancic, and Deva Ramanan. Understanding everyday hands in action from rgb-d images. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pages 3889–3897, Washington, DC, USA, 2015. IEEE Computer Society. ISBN 978-1-4673-8391-2. doi: 10.1109/ICCV.2015.443. URL `http://dx.doi.org/10.1109/ICCV.2015.443`. 3.2.4, 6.5.1, 6.9

J. Romero, H. Kjellstrom, and D. Kragic. Hands in action: real-time 3d reconstruction of hands in interaction with objects. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 458–463, 2010. doi: 10.1109/ROBOT.2010.5509753. 3.1.2

Charles Rose, Michael F. Cohen, and Bobby Bodenheimer. Verbs and adverbs: Multidimensional motion interpolation. *IEEE Comput. Graph. Appl.*, 18(5):32–40, September 1998. ISSN 0272-1716. doi: 10.1109/38.708559. URL `http://dx.doi.org/10.1109/38.708559`. 4.5.1

Matthias Schröder, Jonathan Maycock, and Mario Botsch. Reduced marker layouts for optical motion capture of hands. In *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games*, MIG '15, pages 7–16, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3991-9. doi: 10.1145/2822013.2822026. URL `http://doi.acm.org/10.1145/2822013.2822026`. 3.1.3

Toby Sharp, Cem Keskin, Duncan Robertson, Jonathan Taylor, Jamie Shotton, David Kim, Christoph Rhemann, Ido Leichter, Alon Vinnikov, Yichen Wei, Daniel Freedman, Pushmeet Kohli, Eyal Krupka, Andrew Fitzgibbon, and Shahram Izadi. Accurate, robust, and flexible real-time hand tracking. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 3633–3642, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3145-6. doi: 10.1145/2702123.2702179. URL `http://doi.acm.org/10.1145/2702123.2702179`. 3.1.2

T. Simon, H. Joo, I. Matthews, and Y. Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 00, pages 4645–4653, July 2017. doi: 10.1109/CVPR.2017.494. URL `doi.ieeecomputersociety.org/10.1109/CVPR.2017.494`. 3.1.2

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6.5.1

Ronit Slyper and Jessica K. Hodgins. Action capture with accelerometers. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '08, pages 193–199, Aire-la-Ville, Switzerland, Switzerland, 2008. Eurographics Association. ISBN 978-3-905674-10-1. URL `http://dl.acm.org/citation.cfm?id=1632592.1632620`. 3.1.3

Sony Mobile, Inc. Floating touch. `http://developer.sonymobile.com/knowledge-base/technologies/floating-touch/`. 4.5.2

David J. Sturman and David Zeltzer. A survey of glove-based input. *IEEE Comput. Graph. Appl.*, 14(1):30–39, January 1994. ISSN 0272-1716. doi: 10.1109/38.250916. URL `http://dx.doi.org/10.1109/38.250916`. 3.1.2

Shinjiro Sueda, Andrew Kaufman, and Dinesh K. Pai. Musculotendon simulation for hand animation. *ACM Trans. Graph.*, 27(3):83:1–83:8, August 2008. ISSN 0730-0301. doi: 10.1145/1360612.1360682. URL `http://doi.acm.org/10.1145/1360612.1360682`. 3.1.1

Xiao Sun, Yichen Wei, Shuang Liang, Xiaoou Tang, and Jian Sun. Cascaded hand pose regression. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 3.1.2

Ilya Sutskever, Geoffrey E. Hinton, and Graham W. Taylor. The recurrent temporal restricted boltzmann machine. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1601–1608. Curran Associates, Inc., 2009. URL `http://papers.nips.cc/paper/3567-the-recurrent-temporal-restricted-boltzmann-machine.pdf`. 3.2.11

Jie Tan, Yuting Gu, C Karen Liu, and Greg Turk. Learning bicycle stunts. *ACM Transactions on Graphics (TOG)*, 33(4):50, 2014. 3.2.12

Vildan Tanriverdi and Robert JK Jacob. Interacting with eye movements in virtual environments. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 265–272. ACM, 2000. 3.2.7

Graham W. Taylor and Geoffrey E. Hinton. Factored conditional restricted boltzmann machines for modeling motion style. In *Proceedings of the 26th Annual International*

Conference on Machine Learning, ICML '09, pages 1025–1032, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-516-1. doi: 10.1145/1553374.1553505. URL `http://doi.acm.org/10.1145/1553374.1553505`. 3.2.11

Graham W. Taylor, Geoffrey E. Hinton, and Sam T. Roweis. Modeling human motion using binary latent variables. In B. Schölkopf, J.C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 1345–1352. MIT Press, 2007. URL `http://papers.nips.cc/paper/3078-modeling-human-motion-using-binary-latent-variables.pdf`. 3.2.11

G.W. Taylor, L. Sigal, D.J. Fleet, and G.E. Hinton. Dynamical binary latent variable models for 3d human pose tracking. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 631–638, June 2010. doi: 10.1109/CVPR.2010.5540157. 3.2.11

Yue Peng Toh, Shan Huang, J. Lin, M. Bajzek, G. Zeglin, and N.S. Pollard. Dexterous telemanipulation with a multi-touch interface. In *Humanoid Robots (Humanoids), 2012 12th IEEE-RAS International Conference on*, pages 270–277, Nov 2012. doi: 10.1109/HUMANOIDS.2012.6651531. 4.5.1

Jonatan Tompson, Murphy Stein, Yann LeCun, and Ken Perlin. Real-time continuous pose recovery of human hands using convolutional networks. *ACM Transaction on Graphics*, 2014. to appear. `http://youtu.be/J4c_x1QnW0A`(Video). 3.1.2

Winnie Tsang, Karan Singh, and Eugene Fiume. Helping hand: An anatomically accurate inverse dynamics solution for unconstrained hand motion. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '05, pages 319–328, New York, NY, USA, 2005. ACM. ISBN 1-59593-198-8. doi: 10.1145/1073368.1073414. URL `http://doi.acm.org/10.1145/1073368.1073414`. 3.1.1, 3.2.2

Raquel Urtasun, David J. Fleet, Andreas Geiger, Jovan Popović, Trevor J. Darrell, and Neil D. Lawrence. Topologically-constrained latent variable models. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 1080–1087, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-205-4. doi: 10.1145/1390156.1390292. URL `http://doi.acm.org/10.1145/1390156.1390292`. 3.2.11

Jacob Walker, Carl Doersch, Abhinav Gupta, and Martial Hebert. An uncertain future: Forecasting from static images using variational autoencoders. *CoRR*, abs/1606.07873, 2016. URL `http://arxiv.org/abs/1606.07873`. 10

Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Multifactor gaussian process models for style-content separation. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 975–982, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-793-3. doi: 10.1145/1273496.1273619. URL `http://doi.acm.org/10.1145/1273496.1273619`. 3.2.11

J.M. Wang, D.J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(2):283–298, Feb 2008. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.1167. 3.2.11

Robert Wang, Sylvain Paris, and Jovan Popović. 6d hands: Markerless hand-tracking for computer aided design. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 549–558, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0716-1. doi: 10.1145/2047196.2047269. URL `http://doi.acm.org/10.1145/2047196.2047269`. 3.1.2

Robert Y Wang and Jovan Popović. Real-time hand-tracking with a color glove. In *ACM Transactions on Graphics (TOG)*, volume 28, page 63, 2009. 3.1.2, 4.5.2

Nkenge Wheatland, Sophie Jörg, and Victor Zordan. Automatic hand-over animation using principle component analysis. In *Proceedings of Motion on Games*, MIG '13, pages 175:197–175:202, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-2546-2. doi: 10.1145/2522628.2522656. URL `http://doi.acm.org/10.1145/2522628.2522656`. 3.1.3

Chi Xu and Li Cheng. Efficient hand pose estimation from a single depth image. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 3456–3462, Dec 2013. doi: 10.1109/ICCV.2013.429. 3.1.2

Katsu Yamane, James J. Kuffner, and Jessica K. Hodgins. Synthesizing animations of human manipulation tasks. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, pages 532–539, New York, NY, USA, 2004. ACM. doi: 10.1145/1186562.1015756. URL `http://doi.acm.org/10.1145/1186562.1015756`. 3.2.2

Yuting Ye and C Karen Liu. Synthesis of detailed hand manipulations using contact sampling. *ACM Transactions on Graphics (TOG)*, 31(4):41, 2012. 3.1.3

Sang Hoon Yeo, Martin Lesmana, Debanga R. Neog, and Dinesh K. Pai. Eyecatch: Simulating visuomotor coordination for object interception. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 31(4), 2012. 3.2.2

KangKang Yin and Dinesh K. Pai. Footsee: An interactive animation system. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 329–338, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association. ISBN 1-58113-659-5. URL `http://dl.acm.org/citation.cfm?id=846276.846323`. 3.1.3

Wentao Yu, R. Alqasemi, R. Dubey, and N. Pernalete. Telemanipulation assistance based on motion intention recognition. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 1121–1126, April 2005. doi: 10.1109/ROBOT.2005.1570266. 3.2.6

Wenping Zhao, Jinxiang Chai, and Ying-Qing Xu. Combining marker-based mocap and rgb-d camera for acquiring high-fidelity hand motion data. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '12, pages 33–42, Aire-la-Ville, Switzerland, Switzerland, 2012. Eurographics Association. ISBN 978-3-905674-37-8. URL `http://dl.acm.org/citation.cfm?id=2422356.2422363`. 3.1.2

Wenping Zhao, Jianjie Zhang, Jianyuan Min, and Jinxiang Chai. Robust realtime physics-based motion control for human grasping. *ACM Trans. Graph.*, 32(6):207:1–207:12, November 2013. ISSN 0730-0301. doi: 10.1145/2508363.2508412. URL `http://doi.acm.org/10.1145/2508363.2508412`. 3.1.1, 3.1.2, 8.3

Thomas G. Zimmerman, Jaron Lanier, Chuck Blanchard, Steve Bryson, and Young Harvill. A hand gesture interface device. In *Proceedings of the SIGCHI/GI Conference on Human Factors in Computing Systems and Graphics Interface*, CHI '87, pages 189–192, New York, NY, USA, 1987. ACM. ISBN 0-89791-213-6. doi: 10.1145/29933.275628. URL `http://doi.acm.org/10.1145/29933.275628`. 3.1.2