Towards New Interfaces for Pedagogy

by

Murphy Martin Stein

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science

New York University

January 2014

Professor Ken Perlin

Dedication

To my wife, Alexandra, who continually shows me how to live with intention and love.

Acknowledgements

I need to acknowledge many people whole helped me complete this thesis.

First among them, I need to thank my parents. For mother Aleen, I am so grateful to you for teaching me how to make productive analogies which gives me intuition for so many foreign ideas. For my father, Bob, I need to thank you for teaching me how to program in HyperCard when I was seven, and for demonstrating the value in being visionary.

Second, I have to thank my former boss, Michael Artin, who was my first real math teacher, when I was already 26, and who convinced me to do my graduate study at the Courant Instititue.

Third, I have to thank my advisor Ken Perlin, who taught me so many wonderful things, including computer graphics and image processing, and perhaps most importantly, who modeled his method of invention for me so that I can now imitate it productively.

Fourth, I need to thank my many other collaborators, including Jan Plass who provided the scientific foundation for my games research; Elizabeth Heyward for organizing all our games experiments with live kids; Alec Jacobson who was the best co-game designer I've ever worked with; Jonathan Tompson who showed me the value of a good question so many times; Professor Steve Feiner for pioneering work that demonstrated the vast possibilities of Augmented Reality which was both influential and inspirational to my own work; Xiao Xiao at MIT, who taught me how to do user studies, and was a glorious collaborator; Clement Farabet whose alacrity has been an inspiration to me for many years now; Nathan Silberman for many helpful discussions about machine learning; Qingnan Zhou whose careful thinking was useful on so many occasions; Charley

Hendee for many wonderful conversations and outfits; Adam Gashlin for taking games seriously; Professor Rob Fergus for introducing me to computational photography; Professor Yann LeCun for introducing me to Convolutional Neural Networks, teaching me many important algorithms, and introducing me to GPU programming; Professor Chris Bregler for many helpful discussions about human pose estimation; Professor Margaret Wright for teaching me about the big ideas in optimization and giving me pseudo-code for the Broyden-Fletcher-Goldfarb-Shanno (BFGS) Quasi-Newton algorithm; and Professor Hiroshi Ishii at MIT whose insistence on making beautiful interfaces has been another huge inspiration to me.

I also need to acknowledge the generous support from several sources for funding my research: Microsoft Inc. funded my research on games for learning; Google Inc., Microsoft Inc. and SAP all provided partial funding for my research on augmented reality presentations (ARCADE), and the department of Computer Science at NYU for all other funding.

Abstract

Developing technology to help people teach and learn is an important topic in Human Computer Interaction (HCI). In this thesis we present three studies on this topic. In the first study, we demonstrate new games for learning mathematics and discuss the evidence for key design decisions from user studies. In the second study, we develop a real-time video compositing system for distance education and share evidence for its potential value compared to standard techniques from two user studies. In the third study, we demonstrate our markerless hand tracking interface for real-time 3D manipulation and explain its advantages compared to other state-of-the-art methods.

Table of Contents

Dedication ii

Acknowledgements iii

Abstract v

List of Figures ix

List of Tables xi

List of Appendices xii

Introduction 1

I Designing Games to Teach Mathematics **5**

0.1 Motivation 6

0.2 Past Work 7

0.3 Our contribution 7

1 FactorReactor **9**

1.1 Introduction 9

1.2 Gameplay 11

1.3 Modes of Play 14

1.4 Study 15

2 SuperTransformations **28**

2.1 Introduction 28

2.2 Gameplay 29

2.3 Level Design 30

2.4 Mathematical Content 30

3 Noobs vs. Leets **35**

3.1 Storyline 35

3.2 Mathematical Content 35

3.3 Target Audience 37

3.4 Level Design 37

3.5 Study 39

II Augmented Reality Communication **46**

4 Augmented Reality Communication **47**

4.1 Introduction 47

4.2 Related Work 49

4.3 Setup 52

4.4 Implementation 54

4.5 Applications 62

4.6 Evaluation 65

4.7 Discussion and Future Work 73

4.8 Conclusion 75

III Real-Time Hand Pose Estimation 76

5 Real-time Hand Pose Estimation 77

5.1 Introduction 77

5.2 Related Work 80

5.3 Binary Classification 82

5.4 Dataset Creation 85

5.5 Feature Detection 90

5.6 Pose Recovery 96

5.7 Results 98

5.8 Future Work 103

5.9 Conclusion 104

IV Back Matter 105

Appendices 106

Bibliography 121

List of Figures

1.1 FactorReactor: Comparison of three experimental versions 23

1.2 FactorReactor: Screengrab of Practice mode 24

1.3 FactorReactor: Controls for XBox 360 Controller 24

1.4 FactorReactor: Number of problems solved by game version. 25

1.5 FactorReactor: Mastery goal orientation of players by game version. . . 25

1.6 FactorReactor: Situational interest of players by game vesion. 26

1.7 FactorReactor: Player enjoyment by game version. 26

1.8 FactorReactor: Player intention to replay game. 27

1.9 FactorReactor: Player intention to recommend game to someone else. . 27

2.1 SuperTransformations level 1, rendered. 32

2.2 SuperTransformations, level 1 description in level file. 32

2.3 SuperTransformations: Step-by-step solution of level 1. 33

3.1 Noobs: Character types 38

3.2 Noobs: Screenshots from two versions of the game 45

4.1 ARCADE: Live screengraphs 48

4.2 ARCADE setup showing presenter and audience views. 53
4.3 ARCADE: Presenter‘s dashboard 53
4.4 ARCADE: Gesture grammar 60
4.5 ARCADE: Screenshot of manipulation of 3d model 63
4.6 ARCADE: Screenshot of interaction with topographic map 64
4.7 ARCADE: Screenshot of tangible interaction capabilities 65
4.8 ARCADE: Study of sorting task timings 69

5.1 Hand Pose: Pipeline overview 79
5.2 Hand Pose: Example decision forest classification results 84
5.3 Hand Pose: Linear blend skinning model 87
5.4 Hand Pose: Inferred feature locations 91
5.5 Hand Pose: ConvNet architecture 92
5.6 Hand Pose: Multi-resolution pyramid input 93
5.7 Hand Pose: High-resolution feature detector bank 94
5.8 Hand Pose: Neural network for heat-map generation 95
5.9 Hand Pose: Decision forest binary classification error 99
5.10 Hand Pose: Dataset generation vs. ground truth 100
5.11 Hand Pose: Example test set images for ConvNet 101
5.12 Hand Pose: ConvNet learning curve 101

List of Tables

2.1 SuperTransformations: Screenshots 34

3.1 Noobs: Mathematical rules 44

4.1 ARCADE: Hand-tracking pipeline stages 54

4.2 ARCADE: Screengrabs of ball sorting task 67

4.3 ARCADE: Comparison of presentation formats 71

4.4 ARCADE: Presentation formats preferences 72

5.1 Hand Pose: Heat-Map of UV Error by Feature Type 102

List of Appendices

Appendix A Games: Evaluation Materials 106

Appendix B Hand Pose Inference: A Post-Mortem 117

Introduction

At this moment, you are reading a sentence I wrote, and which I transmitted to you via computer. Evidently, we have used computers to communicate. It seems obvious today to point out that computers are an important communication channel. And yet, this simple fact obscures a miraculous theory — information theory — whose careful application over sixty years has enabled the reliable transmission of information across noisy channels, such as electrical wires and free-air. We owe the theory to Claude Shannon who developed his “Mathematical Theory of Communication” in 1948, but I would argue we owe at least as much to the many people who have understood the theory and applied it, and to the billions of people who have come to rely on it.

Now this theory and so many others have rendered computation a fundamental resource, which is as essential to human productivity today as the telephone has been for over 100 years, and writing has been for several thousand years. As a computer scientist, I am interested in understanding what properties this resource has. At the same time, I want to exploit these properties to solve practical problems. This also makes me a computer engineer.

The practical problem that interests me, and which has been guiding my PhD research, is “How can we use computers to augment human intellect?” This is the way that Douglas Engelbart phrased it in a report to the Air Force in 1962 [25], before they gave him money to start the Augmentation Research Center at SRI International, where he and others developed many of the ideas for personal computers. By its nature, this problem concerns humans as well as computers, and so it has come to be studied by computer scientists in the field called “Human Computer Interaction” (HCI). However, it is also studied in several other fields, which are usually considered distinct from HCI.

For example, I have been lucky enough to collaborate with experts in Educational Psychology, Cognitive Science, and Game Design, as well as from other parts of Computer Science, including Computer Graphics, Computer Vision, and Machine Learning. I have also collaborated with K-12 teachers and their pupils, who use computers on a daily basis, continually experimenting with many "solutions" to this problem.

Before discussing my thesis, I would like to take a little time to explain my perspective on this vast problem. My way into this problem was through developing computer software for teachers and students — so-called "educational technology". I worked for many years at Scholastic Inc., which is the publisher of children's books such as "Harry Potter", and which is also the market leader in selling reading and math software for "the intervention market" — a nicer term that replaced "remedial training". My job was to design adaptive algorithms for online tests that could quickly score ability from as few as 20 items. I needed to learn about Bayesian probability, and I got my first taste of mathematical optimization, in the form of"Item Response Theory" (IRT), which we used for our tests. This is the same theory that drives other adaptive tests like the Graduate Record Examination (GRE). I learned recently that one math test I developed has been taken over a million times.

I never liked taking tests, and I was never very good at them, so originally I felt awkward developing tests for the mass market. However, my feelings changed when I learned that teachers depend on Scholastic's rapid assessments to retarget their lessons for different student needs. I began to see computerized testing as a very natural way to efficiently measure specific pieces of knowledge. The idea of administering tests was still troubling to me though, because they were being overemphasized in U.S. schools as a result of the No Child Left Behind Act. On the one hand tests can provide useful indications of what you know, and what you are ready to know; on the other hand, taking too many tests takes time away from instruction and learning.

I already knew about several impressive efforts to use computers for instruction — so-called "Computer Aided Instruction (CAI)" — specifically for learning rather than testing. For example, Patrick Suppes at Stanford developed a number of computer-aided instructional programs in the 1960s for teaching arithmetic, logic, and even Russian [75]. At its height these programs were used by more than 7,000 students in three states in one semester. And the PLATO Learning System was a landmark effort begun out of University of Illinois at Urbana-Champaign in the 1960s to develop both the hardare and software for teaching diverse subjects, including mathematics, physics, chemistry, and history [10]. This system has gone through several iterations, and is still in use today as a commercial product. Then there was the LOGO programming system developed by Seymour Papert and his colleagues at MIT in the 1970s to teach mathematical thinking and programming [63], which at its height was used by at least 1 million students. By the mid 1980s, Papert's students had developed lesson plans to teach sophisticated subjects including hyperbolic geometry, and special relativity [2]. And probably the first use of computers for instruction was to make analog flight simulators to train pilots in the United States and Royal Air Forces in the 1930s [83].

There are two important principles underlying all these educational technologies. The first common principle is that humans learn from feedback, and computers can be programmed to vary feedback from human input [25]. The second principle is that certain "human factors" [83, 59] make information easier or harder to process. For example, showing a series of diagrams of a cockpit to a rookie pilot on a computer might be less useful than letting them sit inside a simulated cockpit and giving him the opportunity to *use* the controls to fly a plane. Human factors are merely aspects of human behavior or biology which are very difficult to change, or may be unchangeable.

What kinds of human factors are there, and which ones should people pay attention to when they design computer software for learning? I will attempt to ouline a few of them that have been central to my thinking and my PhD research. One important human factor is that *personal motivation* strongly influences a student's ability to learn [24]. And importantly, motivation can be influenced, and is not static [22]. A second important human factor is that we process information better when multiple sensory modalities are used simultaneously [67]. This means that finding ways to supplement pictures with audio, or spoken explanations with body language can improve learning. A third human factor is that we have a limited ability to extrapolate from our experiences to new situations [71, 13]. This means that it is important to consider how information is going to be used, not just how it is presented.

In this thesis, I will discuss several experiments I conducted to explore ways to augment the human intellect by addressing specific human factors. The first set of experiments are three different games I developed to make learning elementary mathematics fun and engaging. This work attempts to enhance student motivation, and thereby increase engagement and learning. The second set of experiments are about creating better presentation tools for teachers, and in particular giving them the ability to control real-time special effects in real-time video of themselves. This work attempts to leverage our ability to learn from multiple streams of information, particularly from audio and body language, and also to give teachers "power-ups" in the classroom to make their ideas more compelling. The third set of experiments arose naturally out of the second; it attempted to answer the question "How can we track the full pose of the human hand from a single commodity depth camera?" This is the most technical of my experiments and it attempts to open new arenas for students and teachers who want to directly manipulate pixels in space, such as for real-time special effects.

Part I

Designing Games to Teach Mathematics

0.1 Motivation

No country has figured out how to teach all of its citizens proficiency in mathematics reliably or cheaply. For example, on international tests of math skills, such as the PISA, or TIMSS, even the very best performing countries, such as Singapore, South Korea, and China, produce fewer than 75% of students at the proficient level [55]. In the United States, the situation is typically much worse than in these countries. For example, the National Assessment of Yearly Progress (NAEP) is a national test administered every year to 4th and 8th grade students. It is colloquially referred to as "The Nation's Report Card" because it has been used in policy-making decisions, and by news organizations, to gauge American reading and math skills since 1969. In 2013, $350,000$ students took the exam. Less than 40% of those students had proficient scores in mathematics [23]. Year to year these problems remain, and so educators are in search of solutions.

An old question that has seen a resurgence of interest from educators and a small group of researchers, is "How can computer games be used to supplement or even replace existing learning materials" [74, 30]. The corresponding research hypothesis is that a properly designed game should engage students to learn through gameplay, and that the knowledge they gain can be transferred outside the game. Although commercial games for learning were developed as early as the 1960s for mathematics and the sciences [10], many of these games capitalized on student's *extrinsic interest* in accruing points [50], rather than their *intrinsic interest* in learning the subject matter itself, and in getting good at the gameplay mechanics [18]. Therefore, an exciting new area of research in games for learning is asking how to capitalize on students intrinsic interest through the careful design of game mechanics [69]. The goal of this kind of research is to identify game mechanics that enable players to gain real-world world knowledge merely by getting good at the game. My work in games for learning has focused in this research area. Specifically I have looked at which game mechanics and game design choices benefit learning elementary mathematics.

0.2 Past Work

Past work in developing games for learning have looked at a few design factors that could enhance learning or motivation. For example, Cordova & Lepper [18] showed that personalization and customization positively enhance student motivation for learning arithmetic, and that increased motivation led to gains in arithmetic ability. *Personalization* refers to putting aspects of the player's own life inside the game, such as their name or things they like. *Customization* refers to letting students choose aspects of the game, such as which character they play, or which background story the game uses. More recently, Andersen et al. [4] looked at how varying design aesthetics such as more or less music, and different amounts of animation, led to increases in gameplay.

0.3 Our contribution

Working with cognitive scientists and educational psychologists from the Games for Learning Institute, we have created three games, and varied some of the gameplay mechanics so that we can empirically measure the importance of various design decisions. The contribution of this work is three-fold. Firstly, for teaching arithmetic, and and certain topics from elementary euclidean geometry, we can demonstrate significantly different learning and engagement outcomes dependent on certain design choices. Second, we share the details of our methodology, including empirical validation studies and testing protocols, so that other researchers can replicate and extend our results. Third, and finally, the game mechanics we chose can serve as an example to other learning game designers for what has already been tried to teach these math subjects.

Chapter 1

FactorReactor

1.1 Introduction

Factor Reactor was the first game I made for research into Games for Learning. In this section, I will describe the history of the game, and how it evolved. I will focus on aspects of the design process that may be helpful for future collaborative research into games for learning.

Originally, my advisor Ken Perlin and I conceived of FactorReactor to provide practice at finding prime factorizations. We designed it for the Microsoft Zune Personal Music Player, because we had a grant from Microsoft to develop games, and they had given us some Zunes for experimentation. The first version of FactorReactor was very simple. It showed a single whole number and a ring of factors surrounding it, and a line of disappearing horizontal dots representing a timer. The object of the game was to divide the central number down to 1 before a time limit ran out.

The second version of FactorReactor added a two-player mode that was the same as the single-player mode, but two players could race against each other. The two-player mode required two Zunes and it used the built-in Bluetooth to display real-time stats about the other player's progress through the race.

The third version of FactorReactor grew into a game about all of arithmetic. Ken Perlin was the judge of a contest to make games for Smart Technologies SmartBoards. One game he liked allowed students to solve entire arithmetic problems by dragging operands and operators into "number sentences" that a T-rex would eat if they were in fact equalites. I really liked that one game could investigate a large set of problems, and so I expanded the scope of FactorReactor, while keeping its existing look-and-feel. Based on the need to select operators, and the increased complexity of the gameplay, I decided I needed more screen real estate, and decided to develop for the Microsoft XBox/PC instead of the Zune. I added some "polish", such as graphics and sound effects and little animations, and I also added a new game mechanic to incentivize players to make as few moves as possible. The game mechanic I came up with was to give the user a supply of "rings", and to charge 1 ring for each operator/operand "move". This makes it unwise to iteratively hack away at the central number using random moves, or, say, repeatedly subtracting or adding 1.

I playtested the third version with 10-15 middle schoolers in an after-school computer class on Staten Island. A few surprising things happened. First, students clearly enjoyed the game, and a crowd of students developed to cheer and jeer whoever was playing. Second, despite the fact that I'd played the game hundreds of times before, many of the students were much better than I was, indicating that the problems were not hard enough. Third, it became clear that merely operating on the central number was too limiting, and players wanted more flexibility. Based on these observations, I made a fourth version.

The fourth version implemented a "switch feature" that allowed players to switch the central number and a factor. This greatly increased the number of arithmetic expressions that could be created. I also implemented a procedural level generation system to invent harder problems on the fly.

At this point, Jan Plass expressed an interested in running an experiment to test the difference between the solo and competive versions and a third collaborative version, in order to investigate how these different modes, or game mechanics, affected learning outcomes. Importantly, since the competitive mode at that point was a straightforward application of a “split screen” game-mechanic, the solo mode and the competitive mode were ideally similar. In order to faithfully test the most rudimentary nature of these different versions, Jan and I designed the collaborative mode so that it was also very similar to the solo and competitive modes.

1.2 Gameplay

FactorReactor is a fast-paced arithmetic practice game designed for players aged 8-13. It is designed for play on a PC with an Xbox Controller. Initially just a single-player version (”solo mode”) was developed, and then competitive, and collaborative versions were adapted. We will describe the rules and controls in terms of solo mode, and then describe the variations for the multi-player modes, which are very similar.

1.2.1 Rules

The objective of FactorReactor is to score as many points as possible. Players score points by combining whole numbers together with operators to solve arithmetic problems, called ”goals” (see Figure 1.1). Each goal contains one or more ”rings”, which are captured when the player reaches the goal, and which in turn, save you from dying.

At the center of the player's screen is a large circular object called the ”Reactor” which has an opening at one end, facing the current goal. The player's present answer sits at the center of the Reactor, and one or more operands surround it in a circle. The player makes a move by selecting an operator, and an operand, and then ”firing” on the core number to change it. Each time the player ”fires” it costs one ring. When the center number equals the goal, the player captures the goal and its rings. Each goal has a 2-4 rings depending on its level of difficulty. When the player completes all the goals they advance to the next level. If they ever run out of rings the game ends. When starting from the first level, or restarting after running out of rings, there are 10 free rings with which to start. The number of rings remaining in the player's cache is indicated at the topright of their screen under their current score.

In each level, the Reactor starts with three to seven operands and up to four operators. The operands are arranged in a circle surrounding the core number. The operators are plus, minus, times, and divide, but on some levels one or more of these may be disabled to increase difficulty. All four operators are shown at the bottom right of the game screen, with disabled ones grayed out. Any combination of available operator and operand can be used an unlimited number of times per goal, level, and game. Every time a operator and operand pair is used to change the central number, exactly one ring is used up. Use of the division operator has special behavior compared to the other operators in that, if dividing the core number by the chosen operand will create a non-integer core number then the core number does not change; it glows red instead, and no ring is used.

When the level starts, all the goals surround the whole reactor in a large circle. A level may have from five to ten goals, depending on the level design. When the core number equals the selected goal, an animation occurs that shows the goal entering the reactor, and releasing its rings. These rings then join the other rings, circling around inside. If there are more goals remaining, the reactor rotates around to the next goal in clockwise order. At anytime during gameplay, the player may also choose which goal is selected by using the controller.

To increase the number of arithmetic expressions available to the player, FactorReactor allows a move called *Swap* that swaps the selected operand with the current number core. Swap is always available, and requires no rings. If the player ever swaps the core number with an operand, the core number becomes that operand and the old core number becomes a regular operand for use to solve this or any other goal. A meaningful animation showing these two numbers switch places accompanies the swap move.

During gameplay, the player has the following moves available to them in FactorReactor:

1. Select another operator
2. Select another operand
3. Swap selected operand and core number
4. Select another goal
5. Apply current operand and operator to core number

These moves can be applied in any order, and an unlimited number of times, apart from the last one, which requires sufficient rings to present in the reactor.

1.2.2 Controls

Players control FactorReactor using the XBox 360 controller shown in Figure 1.3. The control mappings for all possible moves are shown on the figure. Other buttons and pads on the controller are not used.

1.2.3 Scoring

Each time the player completes a goal, their score increases proportional to the difficulty of the goal D, and the number of rings used, R. Precisely, their score S increases according to $S \leftarrow S + 1000\frac{D}{R}$. D is equal to the minimum number of moves required, and is typically in the range $D \in [3, 5]$. Therefore, $\frac{D}{R}$ is usually less than 1 for skilled players, but is typically greater than 1 for less skilled players.

1.2.4 Procedural Level Generation

Levels consist of a set of goals, operands, operators, and a starting central number. Level content can be manually, or procedurally generated using a mixture of human input and pseudo-randomness. For the study of different game versions, described below, levels were created, manually. For casual gameplay, procedural level generation was used. Creating a procedurally generated level requires the level designer to specify four pieces of information: 1) the starting central number, 2) the operands, 3) the operators, 4) the number of goals to create for a given number of moves. Then, a tree is constructed in a breadth-first manner, starting with the central number, and iteratively adding children to each lead node. Each child represents one possible operator and operand applied to the parent. Then to find a goal that is reachable in at most n moves, the level generator picks any node at level n at random.

1.3 Modes of Play

By definition, a game must be "played", and so every game designer must decide how many players there are going to be, and how they will interact. These choices characterize part of the social aspect of game design, of which there are many varieties.

For example, solitaire is a game for just one person, but in poker two or more players compete against each other. In team sports, such as football, baseball, and water polo, players have specialized roles and compete as a team against other teams. In games such as chess, tennis, battleship, or Dungeons and Dragons, players compete against each other, alternating turns. In contrast, in soccer, basketball, boxing, and many others, players compete against each other synchronously in a kind of race. These well-known examples show that there are many ways to vary the social aspect of play. Variations include, Single or multi-player, cooperative or competitive, and Turn-based or racing.

When designing a game to teach arithmetic, an obvious question is how does the social aspect of gameplay affect motivation, and ultimately, how does it affect learning outcomes? In order to answer this question, we implemented three versions of Factor Reactor. There is a "solo" version for playing alone. There is a split-screen "racing" mode for two players to compete against each other. And finally, there is a two-player "team" mode for cooperative, or *collaborative* problem solving. Importantly, we made these versions as similar to each other as possible in order to make a fair comparison between design choices. This means that each version lacks certain game mechanics that might make it more interesting or fun in its own right, but would necessarily make it harder to compare against the other two versions.

1.4 Study

In this section, we cite some of the results from a study led by Jan Plass, Elizabeth Heyward, Bruce Homer, Paul O'Keefe from the CREATE lab to measure the empirical differences between these game versions. My role in this work was to design the game and the in-game data collection system, and I also helped carry out the experiment with students. The results were published in [66] and are summarized here with permission of my co-authors.

1.4.1 Participants

We examined the effect of manipulating the social aspect of FactorReactor in a study with adolescents from New York City. In total 58 students were recruited from public middle schools and after school programs. Participant ages ranged from 11 through 14, and $n = 34$ were female. Participants were randomly assigned to the three study conditions, solo ($n = 16$), race ($n = 20$), or team ($n = 22$).

1.4.2 Procedure

Experimenters travelled to student schools over the course of several months. At each school, participants were instructed that they would be participating in a voluntary study to help game designers develop a new game about math. They were also told they would be asked to complete a short exam before and after playing the game and to complete a brief survey. They were assured that none of their results, or comments would be shared with their parents, teachers, or school officials and would be used solely by the game designers to inform games research. Finally they were told that the entire study would take less than an hour and they would be separated into groups or working alone.

Prior to beginning the experiment, participants were randomly assigned to one of three conditions: Solo-Mode, Race-Mode, or Team-Mode. Then they took a pre-test of their arithmetic ability for three minutes.

Next, all participants were shown a two minute narrated tutorial video explaining how to use the XBox controller to make moves such as changing the operator, operand, swapping, choosing a different goal, and applying a transformation to the central number. At this point, they were asked if they had questions about the rules of the game, and these were answered. The experimenters never expressed tips or strategies for how to win at the game.

Prior to being separated into their assigned experimental groups, students played a practice version of the game for 5 minutes, to familiarize themselves with the game rules and how to use the controller.

Next, each student played an in-game pre-test consisting of a fixed set of game problems for 3 minutes.

Next, students were paired up for the race and team conditions, or they remained at their current computer if they were assigned to Solo-mode. Experimenters instructed players in each condition as follows. Solo-mode participants were told, "When playing the game, get the best score you can." Race-mode participants were told, "When playing the game, compete against each other for the better score." And Team-Mode participants were told, "When playing the game, work together to get the best score." All participants were given exactly 15 minutes to play.

After 15 minutes, they were brought to separate workstations, for the remainder of the experiment. In order, they completed a final in-game post-test, a paper and pencil post-test, and filled out some surveys. Both post-tests consisted of *exactly* the same set of problems as their counterpart pre-test. The paper-and-pencil tests had similar a set of problems to the in-game tests, but were not identical.

1.4.3 Measures

In order to measure objectively student learning outcomes, students completed identical arithmetic fluency tests before and after playing FactorReactor. The test is a modified version of *Woodcock Johnson III Math Fluency subtest* [52]. It was modified by randomizing problem presentation, as well as adding more problems for division, subtraction, and addition. The test consists of 160 problems.

Since the problems in FactorReactor are clearly arithmetic in nature, it was possible to assess students ability to measure if students ability to solve problems posed in the style of FactorReactor changed as a result of the different mode they played. Therefore, we created an in-game pre-test and post-test, consisting of 100 separate goals or problems.

In order to assess how mode of play affected achievement goal orientation, the achievement goal orientation subscale based on the Patterns of Adaptive Learning Scales, which we modified to have simplified language for middle schoolers. The 14-item goal achievement scale asked students to indicate their level agreement using a 7-point Likert scale in response to items such as “One of my goals was to learn as much as I could about the game” (mastery), “One of my goals was to show others that the game was easy for me” (performance-approach), and “It was important to me that my performance on the game didnt make me look stupid” (performance-avoidance).

In order to assess how mode of play affected students situational interest, we administered the Situational Interest Survey, which was modified from the work of Garcia et al. [49] to include questions pertaining to interest in playing games. This survey asked participants to use a 7-point Likert scale to indicate their level of agreement with twelve statements, such as “The game was exciting,” “The things I learned from the game are important to me,” and “I thought the game was interesting.”

In order to assess how mode of play affected students individual interest in math, we administered the Individual Interest Survey. The 10-item survey asked students to indicate their level agreement using a 5-point Likert scale in response to items such as “I like math" and "Math is exciting to me." Two questions regarding videogame and Xbox controllers were also included in this survey.

Lastly, students filled out a survey about Factor Reactor itself in order to assess general reactions to the game, including questions about how fun the game was, difficulty, amount learned, what topics were taught, and suggestions for modifications. Responses were either on a 5-point Likert scale or were open-ended.

1.4.4 Key Results

For each result below, the threshold for significance was assumed to be a p-value below .05, which is the standard level for educational psychology studies.

Math Fluency. Playing FactorReactor for just 15 minutes led to an increase in math fluency, regardless of which game version students played. A paired t-test comparing pre- and post-test scores on the math fluency test was conducted, but one student was omitted since he did not complete the post-test. The results suggest that the post-test score ($M = 66.86, SD = 26.42$) increased significantly from the pre-test score ($M = 70.42, SD = 26.67$), with $t(56) = -2.59, p < .01$.

Game Performance. The number of problems students solved during the in-game post-test was analyzed across game versions. Students playing the race version answered significantly more problems correctly ($M = 12.13, STD = 4.45, p < 0.05$) than students in the solo version (M = 8.29, STD = 4.45). Students playing the team version ($M = 9.87, STD = 4.43$) did not complete or answer correctly significantly more than those in the solo version. Significance was determined by fitting a hierarchical linear model (HLM) to the post-test scores, using a linear combination of pre-test score, game version, and controller-experience as input variables. These results are pictured in Figure 1.4.

Achievement Orientation. Students mastery goal orientation was evaluated post gameplay using a survey on a 7-point Likert scale. An ANOVA of these survey results suggest that students playing the solo version had signficantly lower mastery goal orientation scores ($M = 4.98, STD = 1.10$) than students playing the race ($M = 6.12, STD = 1.22, p < 0.01$) version or team version ($M = 5.71, STD = 1.10, p < 0.05$). A similar ANOVA to analyze the performance-approach and performance-approach scores of students, revealed no statistically significant difference among game versions. These results are pictured in Figure 1.5.

Situational Interest. After playing with FactorReactor, students filled out a situational interest survey (see Appendix.) An ANOVA was performed to analyze the survey responses of each student, grouped by game version. The results suggest that students playing the race version ($M = 5.67, STD = .99$) or the team version ($M = 5.82, STD = .77$) reported statistically significantly more situational interest compared to students playing the solo version ($M = 4.93, STD = 1.19$). A follow-up analysis comparing the difference between the race and team versions did not yield statistical significance in terms of situational interest. These results are pictured in Figure 1.6.

Game Enjoyment. Student enjoyment was measured using a survey. An ANOVA of the results suggest that students playing the game solo enjoyed the game significantly less ($M = 3.53, STD = 0.87$) than students playing the race version ($M = 4.21, STD = .66, p = .03$) or the team version ($M = 4.43, STD = 0.57, p < .001$). These results are pictured in Figure 1.7.

Reengagement Intention. Students intention to play the game again at a later time if given the chance was meaured using a survey. An ANOVA was performed to test the responses by students from different game versions. The analysis suggests that students playing the team version reported higher reengagement intention ($M = 4.21, STD = .71, p = .03$) than in the solo condition ($M = 3.58, STD = 1.12$). The reengagement intention of the race condition players ($M = 4.06, STD = 0.90$) was not statistically significant, compared to the solo condition players. These results are pictured in Figure 1.8.

Recommendation Intention. Student intention to recommend the game to someone else in the future was measured using a survey. Intentions were significantly higher for players of the team version ($M = 4.32, STD = .67, p.01$) than for players of the solo version ($M = 3.42, STD = 1.39$). Responses for students playing the race version were not significantly higher compared with the solo version. Significance was analyzed by performing an ANOVA. These results are pictured in Figure 1.9.

1.4.5 Discussion & Future Work

Students playing *any* version of the game for 15 minutes showed improvement on a standard measure of arithmetic fluency. However, we could not determine that the improvement was due to playing the game, or merely from the so-called "warm-up effect," which explains minor increases in student test performance when similar tests are given in rapid succession. Also, we could not determine from these studies how long these gains persist in students and can be utilized in other activities. Follow-up studies are needed to elaborate these results, which we have not yet conducted.

According to the results from all the surveys, students tended to prefer one or both of the two-player versions compared to the solo version. And the version with the preponderance of positive outcomes was the team version. This is a surprising result, since the two-player versions were designed last, and were therefore most constrained, since they needed to be similar to the solo version. Had the game designer (myself) been free to make all versions better, independently, and without regard to keeping them similar to the solo version, even stronger positive effects might well have been observed for the team versions.

Anecdotally, we observed that some students who complained that they were bad at math in regular school, really enjoyed playing FactorReactor. In some cases, we learned from their teachers that these students were actually struggling in math and that the teacher was surprised the student was able to complete some of the goals asked of them. This got us interested in using FactorReactor with special needs students, and we're currently running a study to see how well this population learns from FactorReactor.

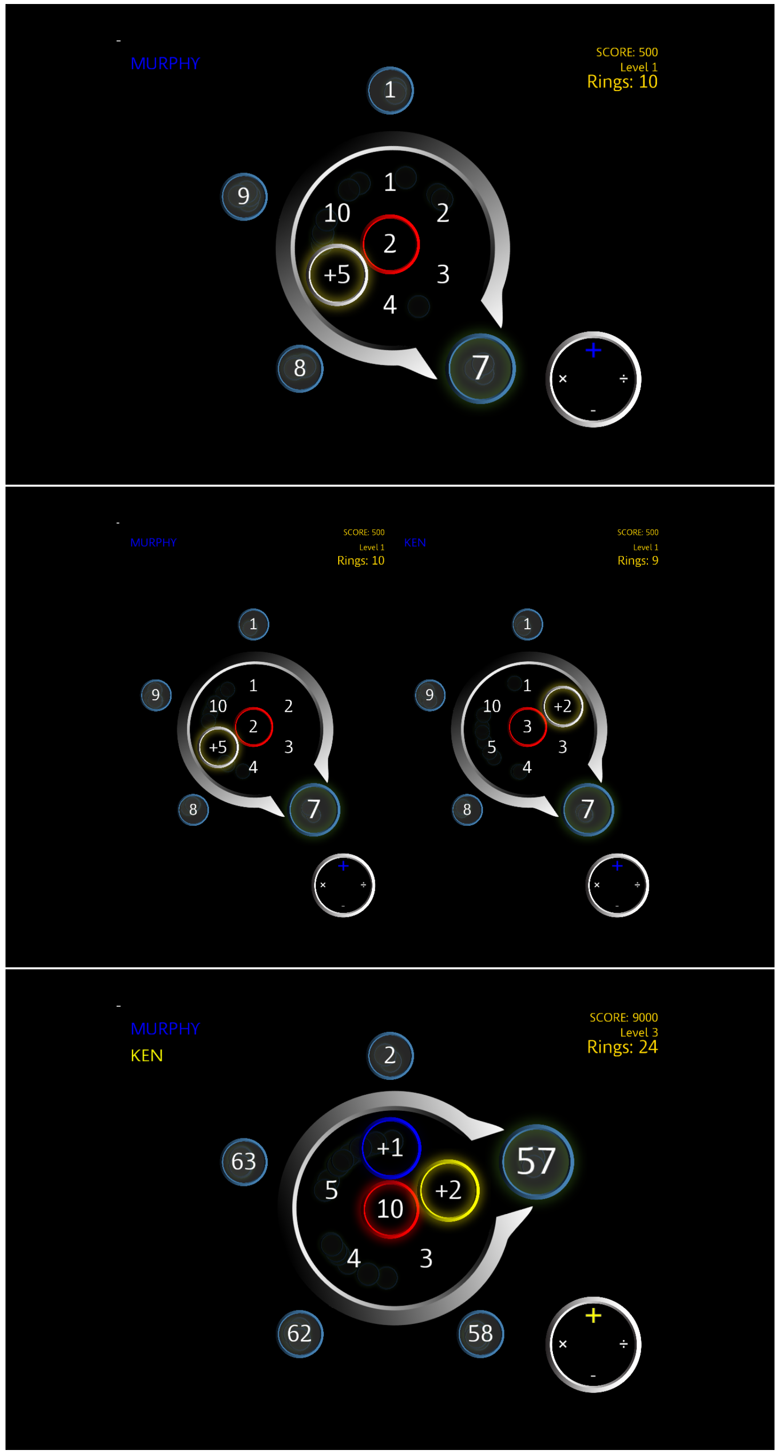


Figure 1.1: Screengrabs of the three experimental versions of Factor Reactor. Solo mode (top), Competitive mode (middle), and Team mode (bottom).

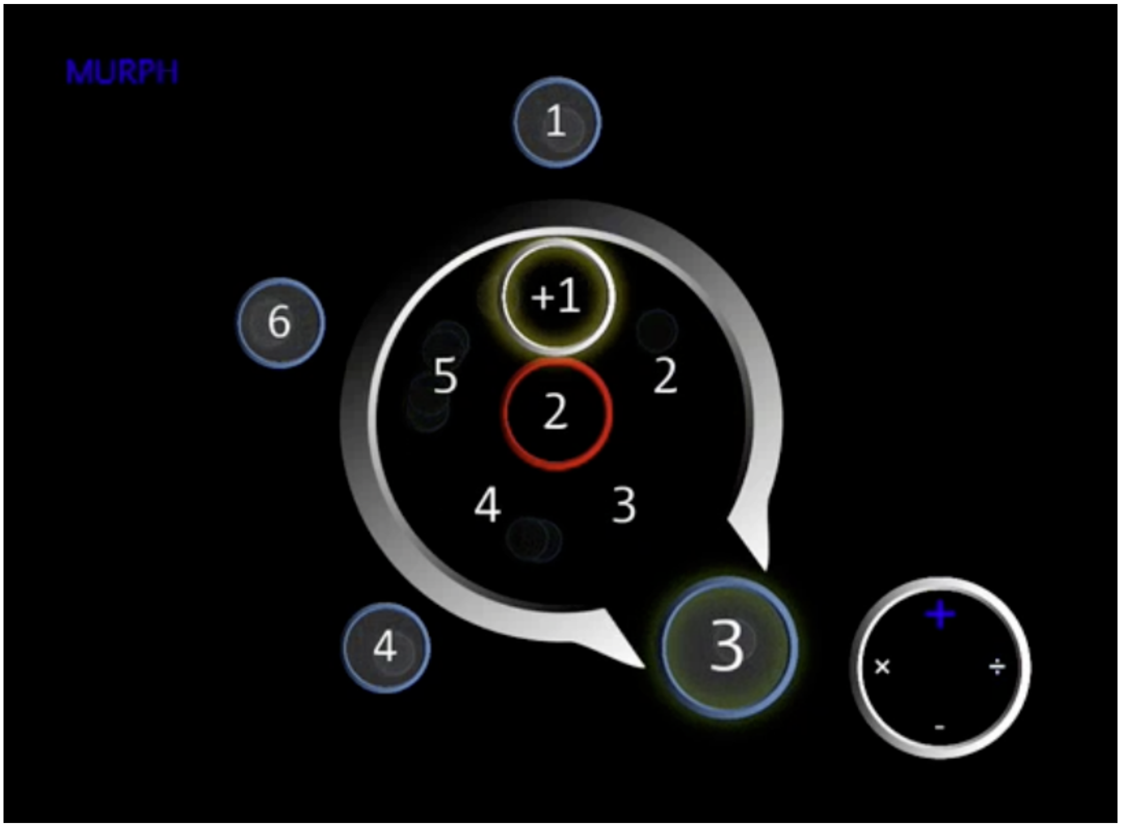


Figure 1.2: Screenshot of FactorReactor in solo-mode during practice

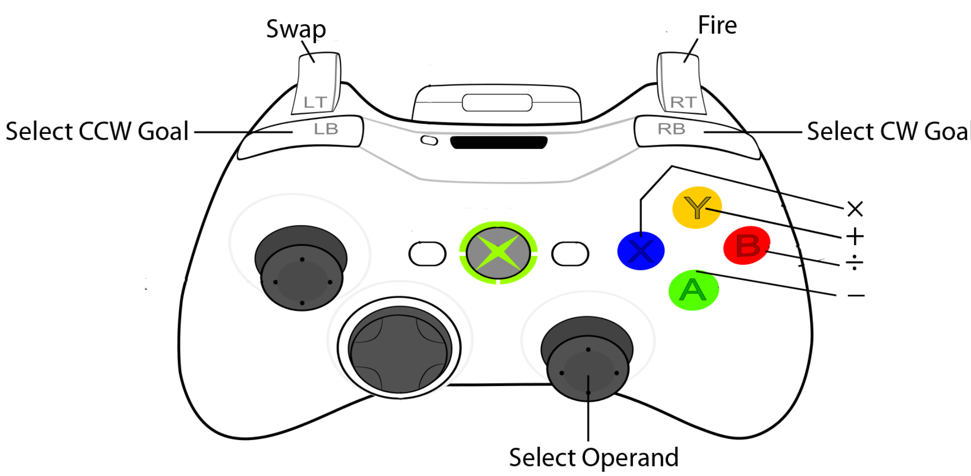


Figure 1.3: FactorReactor: Controls for XBox 360 Controller

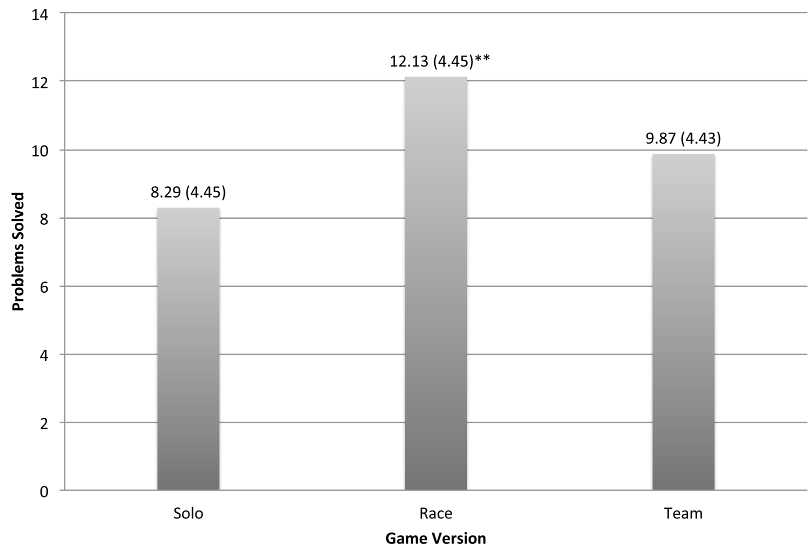


Figure 1.4: Number of problems solved by individuals or pairs during gameplay. Values above bars represent means (and standard deviations) by game version. P value reflects comparsion with solo mode, ** $p < .01$
.

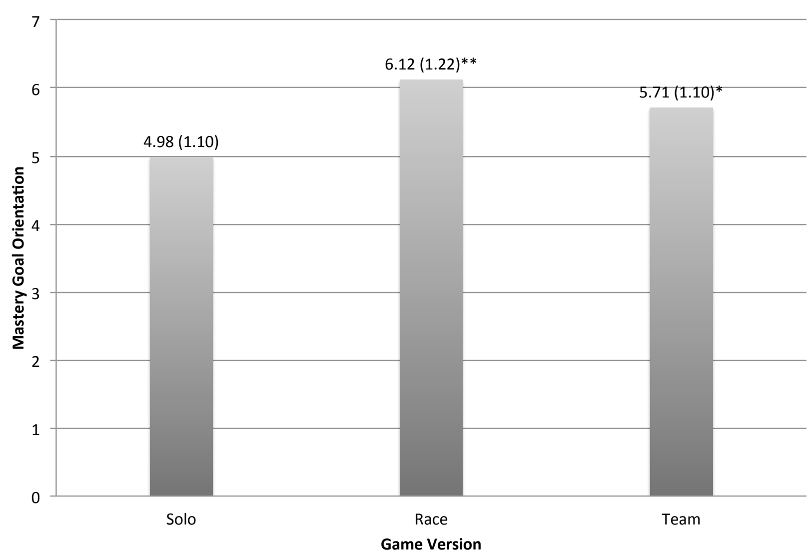


Figure 1.5: Mastery goal orientation scores, as measured from post-game survey. Values above bars represent means (and standard deviations) by game version. The *p-value reflects comparsion with solo mode,* * $p < .05$, ** $p < .01$
.

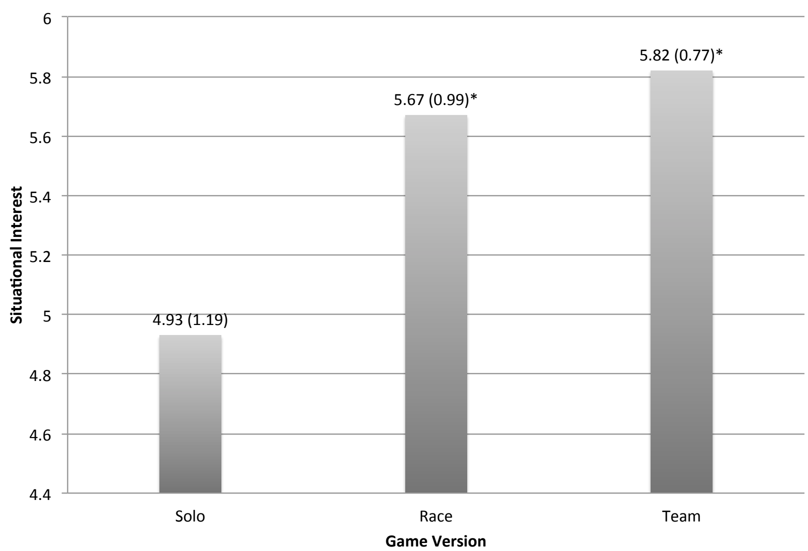


Figure 1.6: Situational interest scores, as measured from post-game survey. Values above bars represent means (and standard deviations) by game version. P value reflects comparsion with solo mode, * $p < .05$.

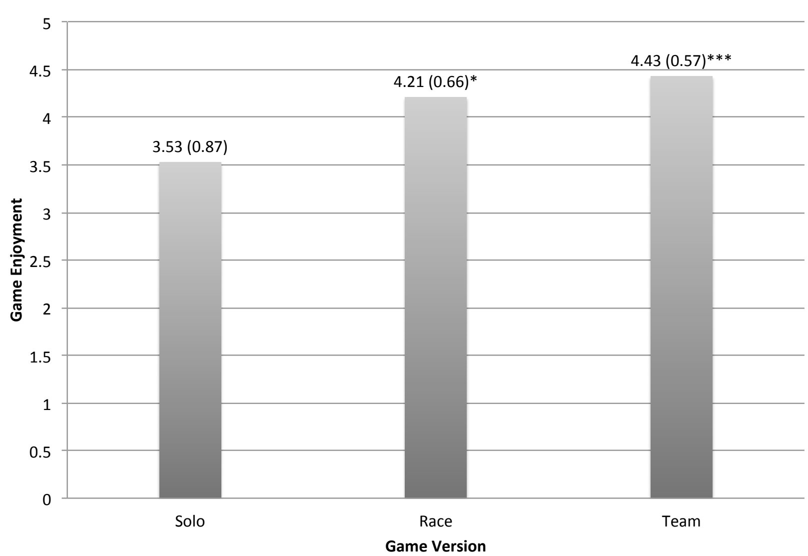


Figure 1.7: Game enjoyment scores reported by participants from post-game survey. Values above bars represent means (and standard deviations) by game version. P value reflects comparsion with solo mode, * $p < .05$, *** $p < .001$

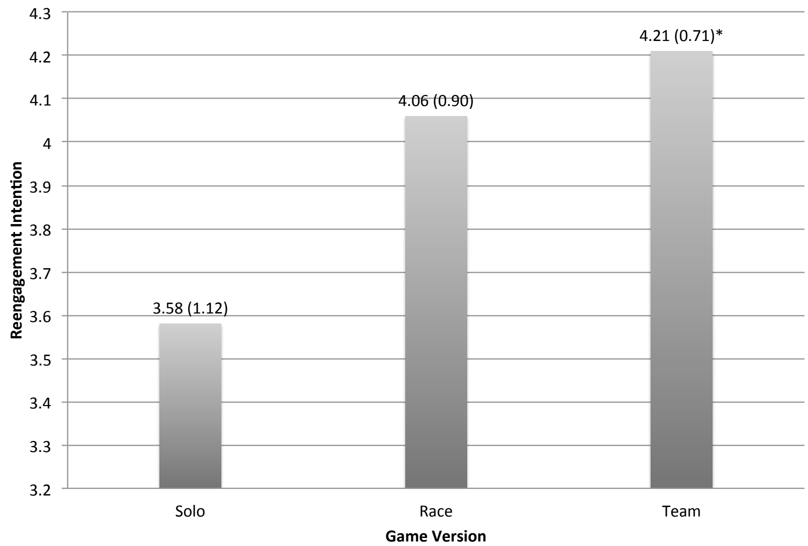


Figure 1.8: Participants' individual self-reported intention to reengage with FactorReactor in the future. Scores measured from post-game survey. Values above bars represent means (and standard deviations) by game version. P value reflects comparsion with solo mode, * $p < .05$.

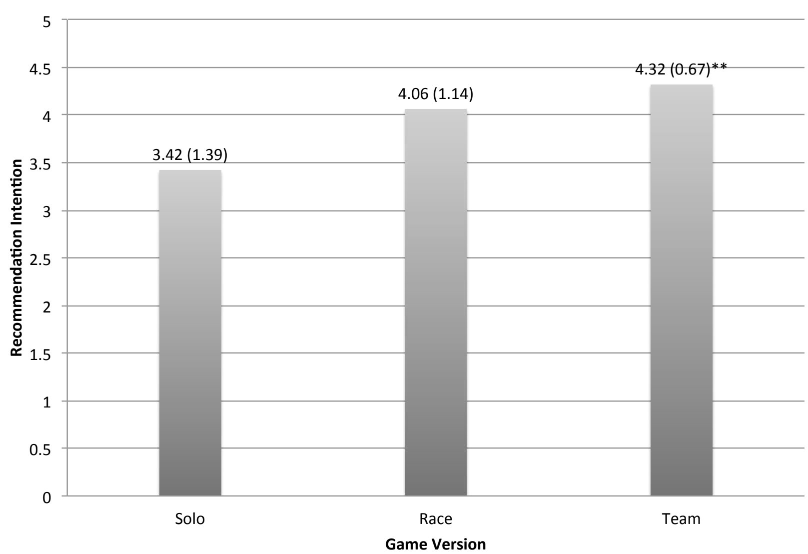


Figure 1.9: Participants individual self-reported intention to recommend FactorReactor to someone else in the future. Scores measured from post-game survey. Values above bars represent means (and standard deviations) by game condition. P value reflects comparsion with solo mode, ** $p < .01$.

Chapter 2

SuperTransformations

2.1 Introduction

Alec Jacobson, a fellow PhD student at the time, and I took Ken Perlin's *Games* class in the fall semester of 2010. During that class, Alec became interested in making a game to teach students how to solve certain 2D geometry problems from the New York State Regent's exam. The exam problems in which he was interested required students to determine if two geometric shapes were equivalent after performing one or more rigid 2D transformations. The possible transformations were rotation, reflection, and translation. In response to these questions, Alec thought of a game in which a missing piece of ground could be filled by another piece only if it was rigidly transformed in the right way. I liked his game, and we ended up working together to build SuperTransformations. We submitted it to the first annual Games For Learning Competition, and it won the Grand Prize. Later, we tested the game with two classes of eigth grade students at a local middle school, totaling about 80 participants in all. As it happened, they were about to cover this material in class, and playng SuperTransformations served as an informal preview of their classwork. We conducted interviews about their gameplay experience, and also asked them to complete pre- and post-tests to gauge the learning benefit of playing SuperTransformations.

2.2 Gameplay

SuperTransformations is based on the idea of a ”platformer” in which the main character is seen from the side and needs to ”jump” from platform to platform to reach his goal. In our game, the main character is named “The Dude” and he walks along platforms, or jumps between them, trying to get to the exit door. The exit door is asymmetric and its outline is the same as the outline of the Dude. He can only fit through the door if his shape perfectly aligns with the shape of the door. Since the door is asymmetric, The Dude might have to rotate or reflect to get through it.

The Dude’s path to the door is usually blocked by several obstacles. One common obstacle is a wall, which can be above him or in front of him. By placing an appropriately placed vertical line or horizontal line reflection, the player can reflect the Dude over a wall (see Figure 2.3). This is not physically very realistic, but mathematically, it makes perfect sense. An added game mechanic is that gravity always points along the vector from the Dude’s head towards his feet. This means that when you transform the Dude, the direction of gravity might change if this vector changes.

An additional game mechanic is a worm that excretes transformations in random locations throughout a level. When the Dude encounters a transformation left by the worm, he transforms as he would from a transformation created by the player. The worm’s transformations stick around until the Dude hits them. The worm game mechanic encourages players to solve the level quickly before too many unwanted transformations are left behind by the worm.

2.3 Level Design

There are 13 levels in SuperTransformations, which were designed by hand for increasing difficulty. Each level is described by a simple string consisting of 20 rows $\times$ 30 columns of ASCII character pairs. Each character pair represents a type of block, a location for the door, or the start position for the Dude. In each pair, the first character designates the type of block, and the second character designates its orientation. Figure 2.2 shows the encoding scheme for level 1, and Figure 2.1 shows the level fully rendered. As in other games, the levels increase in difficulty. All 13 levels and the first two introductory screens are pictured in table 2.1.

2.4 Mathematical Content

The explicit mathematical content of SuperTransformations are of two kinds. First, it attempts to familiarize students with the vocabulary of rigid 2D transformations. Second, it attempts to impart an intuition for how combinations of rigid 2D transformations affect a non-symmetric 2D shape.

From a vocabulary standpoint, students are expected to learn the names of common 2D transformations including, translation, rotation, and reflections. Many state test exam items can be answered correctly, simply by knowing the names for each transformation and how to identify them using simple symmetry. In SuperTransformations we familiarize students with vocabulary by displaying the name of the transformation while the animation for that transformation is being applied. So, for example, when the Dude encounters a horizontal reflection and is starting to *be horizontally reflected* the word "reflection" appears in large obvious letters for the duration of that animation.

From, an intuition standpoint, students are expected to figure out what the effect of applying a transformation, or series of transformations, would be to a given object – usually a simple one. In SuperTransformations, players must constantly apply transformations to the Dude to reorient him so that he fits through the exit door. Thus, the thinking behind the chosen game mechanic, is that repeated use of these transformations in order to solve problems in the game will develop a student's intuition for how to use these transformations outside the game.

Typically, levels require from 1-6 transformations to transform the Dude into the correct orientation, in a place near the exit door. Translation is an automatic, and continuous transformation in SuperTransformation in the sense that the Dude walks forward, on his own, constantly. Reflection about the Dude's vertical axis also happens automatically whenever he bumps head-first into a wall. Obviously, the four manual transformations, form a mathematical group with the caveat that the identity element of this group is merely the non-transformation – i.e. not performing any transformation at all.

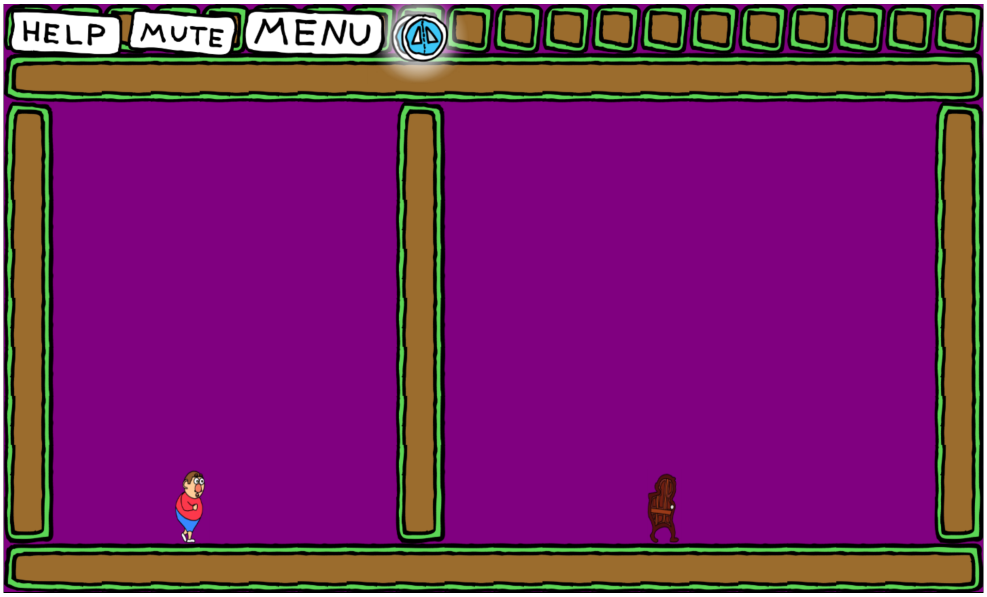


Figure 2.1: SuperTransformations level 1, rendered.

```
B>B#B#B#B#B#B#B#B#B#B#B#B#B#B#B#B#B#B#B<
B>B#B#B#B#B#B#B#B#B#B#B#B#B#B#B#B#B#B#B<
B>              B#                    B<
B>              B#                    B<
B>              B#                    B<
B>              B#                    B<
B>              B#                    B<
B>              B#                    B<
B>              B#                    B<
B>              B#                    B<
B>      P>      B#        @0          B<
B>B#B#B#B#B#B#B#B#B#B#B#B#B#B#B#B#B#B#B<
```

Figure 2.2: SuperTransformations, level 1 description in level file.

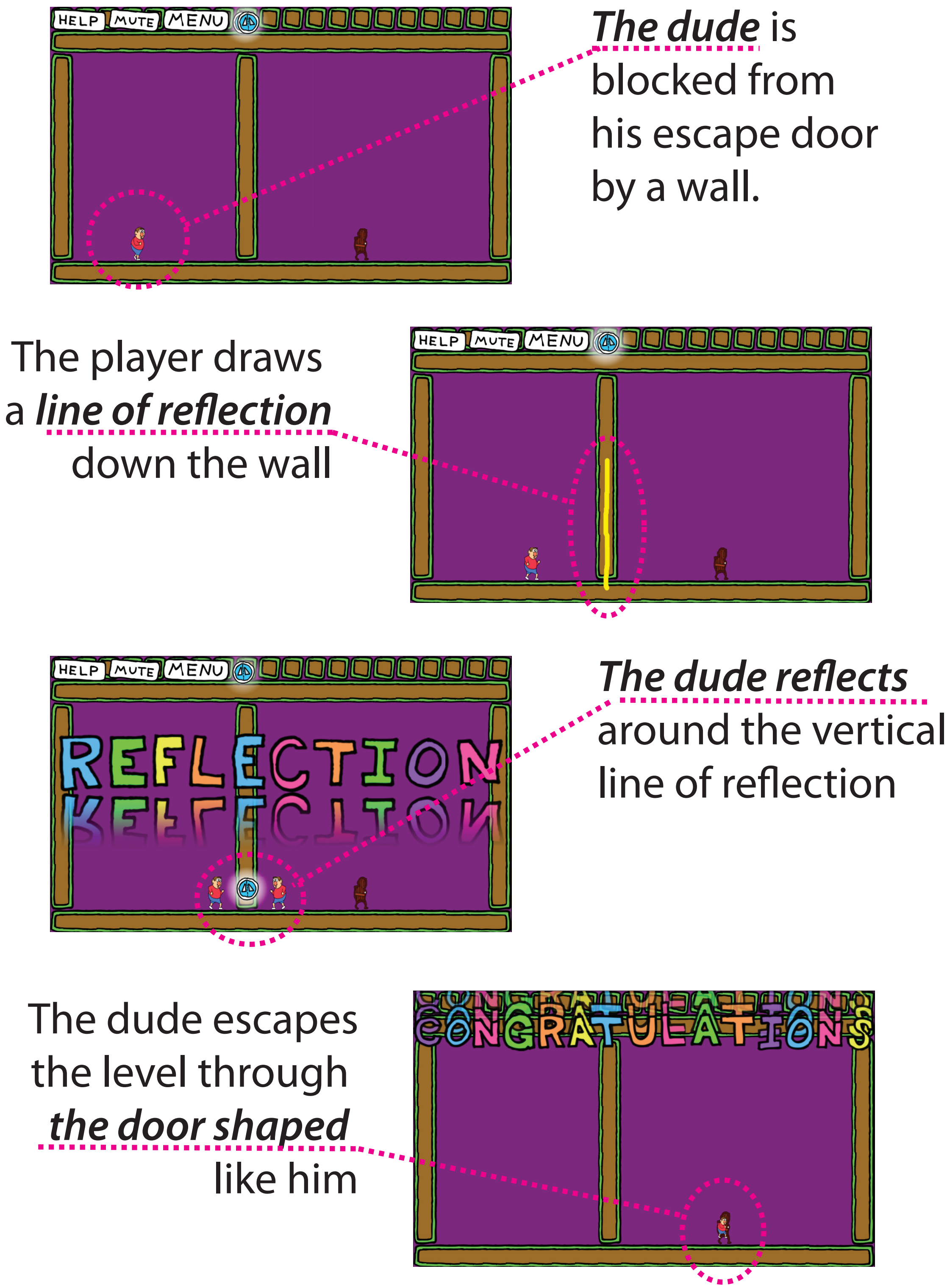
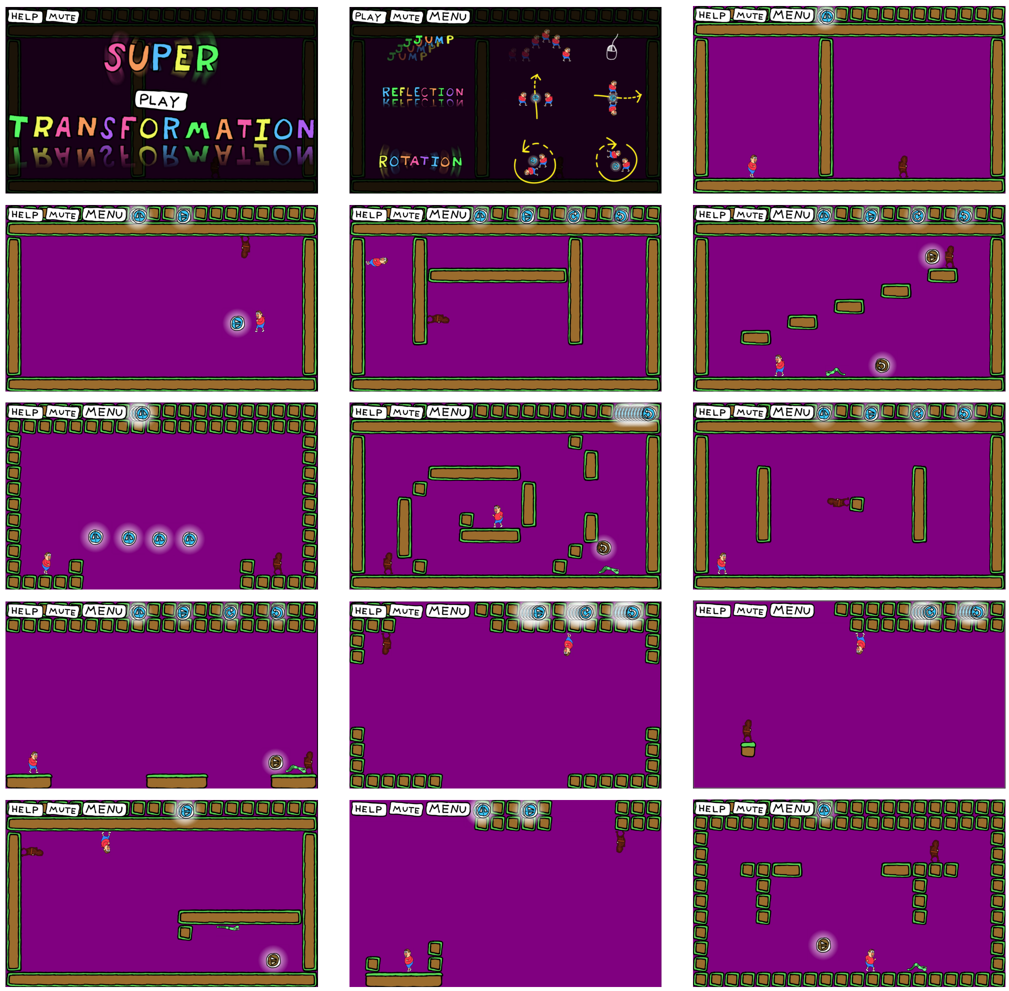


Figure 2.3: Step-by-step solution for the first level.

Table 2.1: Screenshots of SuperTransformations. From top-left: Start Screen, Help Screen, and levels 1 through 13 (the end).



Chapter 3

Noobs vs. Leets

3.1 Storyline

Noobs vs. Leets: The Quadrilateral Battle of Angles and Lines (abbreviated *Noobs*) is a game that focuses on the geometry concept of angles. The basic story is that the "Leets" have captured the unwitting "Noobs" and imprisoned them on different levels of the game world, hiding them at the end of long passageways. As the player, you must help the Noobs escape by finding an escape route. The Leets blocked most of the escape routes, but you can unlock them by applying mathematical truths about angles. In some cases, certain pathways are actively patrolled by Leets, so you must be careful which route you choose.

3.2 Mathematical Content

The educational goal of *Noobs* is to give students experience with some of the most basic definitions or "rules" from Euclidean geometry about angles. These definitions are of two kinds. The first kind consist of definitions about single angles, and there are four of these: *right*, *acute*, *obtuse*, and *straight*. The second kind consist of definitions about two or more angles, and *Noobs* deals with five of these: *complementary*, *supplementary*, *vertical*, *triangle*, and *quadrilateral*. A complete list of the definitions can be found in table 3.1.

Traditionally, these rules or definitions are interesting because they provide the basis for deductive proofs [34]. For example, knowing that a triangle has three interior angles that sum to 180° (i.e. "Triangle Rule") and the values of two of the three angles, you can deduce the value of the third angle. As a second example, knowing that two angles separated by a straight line sum to 180° (i.e. "Supplementary Rule") and the value of just one of the two angles, you can deduce the value of the other angle.

The task of solving for the missing angle entails three basic steps. First, a student must identify the given information and which angle needs to be solved. Second, the student must consider what rules could be applied to generate the missing angle. And third, the student must apply the rule to solve for the missing value. Since knowing which rules would be useful (step one) is arguably a different skill from knowing how to do the arithmetic, we conjectured that teaching students how to do steps one and two only might be a useful teaching strategy for this content.

To test this conjecture, we designed two slightly different versions of *Noobs*. One version, called the *Rule Version* or *Conceptual Version*, requires students to explicitly choose which rule or definition could be used to solve for a missing angle. The second version, called the *Number Version* or *Arithmetic Version* requires students to solve for the missing angle by implicitly applying rules and then explicitly doing the arithmetic in their heads.

3.3 Target Audience

Students typically encounter this material over a two-to-three year period between the ages of seven to fourteen, depending on the country, state, and school curriculum. With the input of twelve and thirteen year-olds at a Staten Island Middle School, we developed the graphics, sound effects, and storyline to appeal to this large age range. For example, a thirteen year-old at this school suggested we call the protagonists “N00bs” and the evil enemy ”L33ts”, which are common terms for novices and expert gamers, respectively, used in chat forums. We took his suggestion, but changed the spelling. A screenshot from an early level of the game showing the game design is pictured in Figure 3.2.

3.4 Level Design

Noobs has 49 levels, broken across five “chapters”. This organizational trope was inspired by Angry Birds [54]. In the spirit of that game, the player starts at the first level on the first chapter, and gradually unlocks higher levels and chapters as they proceed through the game. Each level on any chapter may be replayed once it is unlocked. A level consists of many assets including, vertices, lines, angles, buttons, leets, and in-game tutorials.

3.4.1 Vertices, Lines, and Angles

Each level has a graph structure consisting of vertices, lines, and angles. Angles are uniquely defined by any three vertices connected by lines. An angle may be designated “solved” or “unsolved” by the level designer. Lines are automatically appear “locked” if they are part of any unsolved angle. In each level there are two special vertices. There is

(a) A Noob (b) A Stationary Leet (c) A Moving Leet

Figure 3.1: Good and bad game characters.

a "start" vertex from which the player starts, and a "goal" vertex, where a poor "Noob" is stuck in prison, waiting to be saved.

3.4.2 Buttons

In the "rule version", the allowable buttons are chosen from the nine types of rules mentioned in table 3.1. In the "arithmetic version", the set of buttons is chosen from the numbers in the range [5, 180] in increments of 5. In the first chapter, a mix of both kinds of buttons is used. The chapter has ten levels; the first seven use rule buttons, and the last three use number buttons.

3.4.3 Leets

Starting with *Chapter Two*, most levels have at least one "Leet" sitting on a line to capture unwitting players. There are two kinds of Leets, stationary, and mobile, pictured in Figure 3.1. *Stationary* Leets stay put on the same line and are only dangerous if players happen to choose the wrong pathway to the goal vertex. *Moving* Leets stay put on a line unless the player's Noob is close enough and then they chase after them. Close enough means the player is on a line that is connected to a vertex that the Moving Leet is on. Leets can only move on lines, but they are free to travel on locked or unlocked lines. The purpose of stationary leets is to encourage players to make strategic decisions about which angles to solve, rather than always solving the easy ones.

3.5 Study

Jan Plass, Elizabeth Heyward, and Bruce Homer of the CREATE Lab at NYU, led a study to measure the implications of the use of the learning mechanic of “rule application” in *Noobs*. The study took place from November 2010 through May 2011. The results of the study appeared in *Lecture Notes in Computer Science, 2012*, in a paper by Bruce Homer, Jan Plass, Elizabeth Heyward, Jonathan Frye, Tzuchi Huang, Melissa Biles, myself, and Ken Perlin. My role in this study was to develop the two-games, create the data collection back-end, and also help run the experiment with students. The results are reprinted here with the permission of my co-authors [65].

3.5.1 Participants

The participants were 141 students in sixth through eighth grade from New York City. Sixty five percent (65%) were females. The participants were randomly assigned to play either the rule version or the arithmetic version. Twelve (12) students were excluded from the analysis because they had outlying scores on the arithmetic fluency pretest screener. Another seventeen (17) students were excluded because they completed fewer than 11 levels in the game. (Since both versions of the game are completely identical through level 10, at least eleven levels must be completed for any effect of game version to be detected.) The remaining 112 participants were included in the analysis.

3.5.2 Procedure

Students participated in the study in groups of not more than fourteen. There were three to four adult proctors in the room to assist during each session. When the students arrived, they were told that that they would be doing the following things.

1. Complete a timed arithmetic test (3 mins)
2. Complete a geometry test (10 mins)
3. Play a game (25 mins)
4. Complete another geometry test (10 mins)
5. Complete a few questionaires (untimed)

Each student was given an arithmetic fluency screener to assess their ability to do arithmetic. The screener consists of 160 addition and subtraction problems that use numbers that are similar to the angles found in the "numbers" version of the game. Students were instructed to solve as many problems as possible in three (3) minutes. The test is based on the *Woodcock Johnson III Math Fluency subtest* [52], except problems not pertaining to addition and subtraction were eliminated, and two and three digit addition problems were added. The complete screener is reprinted in the Appendix.

In order to assess the learning outcomes of each game version, participants completed a geometry test *before* and *after* playing the game. The tests consisted of 21 geometry problems, 15 of which were conceptual in nature, and 6 of which were computational. The tests were designed by a former middle school geometry teacher, which he modeled after the New York State End-Of-Year Assessment [58]. The pretest and posttest problems were clones of each other, so that taking each test in succession without intervention would presumably lead to identical test outcomes. The pretest and posttest are reprinted in the Appendix.

After playing the game for 25 minutes and taking the posttest, students completed two individual interest surveys, and one situational interest survey. These are self-report surveys that the students filled out themselves. Each survey consists of a series of statements that students are asked to rate on a 7-point scale from (1) “Not at all true” to (7) “Very true”. The individual interest survey for geometry consisted of 6 statements, including, *“I enjoy doing geometry”*, and *“Geometry is practical for me to know”*. The individual interest survey for games, had 8 statements, including, *“I enjoy playing games”*, and *”Being a good gameplayer is an important part of who I am”*. The situational interest survey consisted of 12 statements, such as *”I liked what I learned in the game”*, and *”The game grabbed my attention”*. The situational interest survey, individual interest in games survey, and individual interest in geometry survey can be found in the Appendix. The situational interest survey was adapted from [49] to include questions pertaining to interest in playing games.

3.5.3 Results

3.5.3.1 Situational Interest

In order to investigate the situational interest in the game, we conducted a between-subjects ANCOVA, controlling for number of levels completed and grade level. This analysis yielded a marginal main effect for condition, $F(1, 82) = 3.64$, $p = .06$, $\eta^2 = .05$, such that the number condition ($M_{adjusted} = 5.45$, $SE = .20$) demonstrated greater situational interest than those in the rules condition, ($M_{adjusted} = 4.90$, $SE = .20$). In educational psychology, the effect is considered “marginal” when there is a $p-value$ of slightly more than .05.

3.5.3.2 Game Performance

In order to examine the effect of condition on game performance, an ANCOVA was performed on the total number of levels completed in the game, with grade level added to the model as a covariate. The analysis yielded effects for grade, $F(1,89) = 9.014$, $p = .004$, $\eta^2 = .10$, as well as condition $F(1,89) = 7.16$, $p = .009$, $\eta^2 = .08$. Participants who were in the 6^{th} grade ($M = 23.84$, $SD = 7.40$) completed significantly fewer levels than those in the 8^{th} grade ($M = 28.52$, $SD = 7.32$). Individuals in the rule group completed more levels ($M_{adjusted} = 28.88$, $SE = 1.06$) than individuals in the number group ($M_{adjusted} = 24.85$, $SE = 1.07$). This suggests that those in the rule group performed better in the game than those in the number group.

3.5.3.3 Geometry Learning Outcomes

A two-way repeated-measures 2 (Time: Pretest vs. Posttest) x 2 (Question Type: Arithmetic vs. Concept) ANCOVA was conducted using the average of correct responses as the dependent variable. Condition was a between-groups factor, and arithmetic fluency pretest and number of levels completed were covariates. A custom RM-ANCOVA model was run with *time*, *question type*, *arithmetic fluency pretest score*, and *levels completed* entered as main effects, and *time by condition*, *time by level*, *condition by level*, and *time by condition by level* as interaction terms.

Main effects were found for time, $F(1,84) = 12.61$, $p = .001$, partial $\eta^2 = .13$ and question type, $F(1,84) = 45.16$, $p < .001$, partial $\eta^2 = .35$. Therefore, there existed overall differences in the responses due to time, such that scores on the outcome measure increased from the pretest ($M_{adjusted} = .50$, $SE = .01$) to the posttest ($M_{adjusted} = .57$, $SE = .01$). There was also a main effect of question type, such that participants in both groups were more successful on the conceptual rule-based questions ($M_{adjusted} = .74$,

$SE = .01$) than on the arithmetic questions ($M_{adjusted} = .35$, $SE = .01$), across the pretest and posttest measures. There were also main effects for both covariates on the outcome measures: arithmetic fluency pretest, $F(1,84) = 23.51$, $p < .001$, partial $\eta^2 = .22$, and levels completed, $F(1,84) = 16.09$, $p < .001$, partial $\eta^2 = .16$.

These main effects were qualified by significant interactions. There was a two-way interaction between time and condition, $F(1,84) = 6.05$, $p = .02$, partial $\eta^2 = .07$, as well as a time and level interaction, $F(1,84) = 4.67$, $p = .03$, partial $\eta^2 = .05$. Of greater theoretical interest is the three-way interaction among time, condition, and level, $F(1,84) = 5.67$, $p = .02$, partial $\eta^2 = .06$. When the results were decomposed by condition, there was simple interaction effect of time by level for the number group, $F(1,41) = 4.49$, $p = .04$, partial $\eta^2 = .10$, but not for the rule group. This interaction was further decomposed by running separate analyses for those who completed less than median number of levels (< 30) or more levels ($\geqslant 30$) for individuals in the numbers condition. For participants in the number condition who completed fewer than 30 levels, there was a significant gain from pretest to posttest, $F(1,24) = 5.55$, $p = .03$, partial $\eta^2 = .19$, whereas for those who completed more than 30 levels, there was no gain from pretest to posttest, $F(1,16) = .174$, *n.s.* The graphic representation of this interaction suggests that for the number group, but not for the rule group, playing more levels diminishes the gain from pretest to posttest. In contrast, for the rule group, the extent of the gain from pretest to posttest was consistent, irrespective of the number of levels completed. The condition by level interaction term did not significantly account for variance in the model.

Right Angle Rule	The angle is exactly 90°
Acute Angle Rule	The angle is less than 90°
Obtuse Angle Rule	The angle is more than 90°
Straight Angle Rule	The angle is exactly 180°
Complementary Rule	A pair of angles separated by a line sum to 90°
Supplementary Rule	A pair of angles separated by a line sum to 180°
Vertical Rule	Opposite angles from two intersecting lines are equal
Triangle Rule	The angles of a triangle sum to 180°
Quadrilateral Rule	The angles of a quadrilateral sum to 360°

Table 3.1: Applicable rules in Noobs vs. Leets

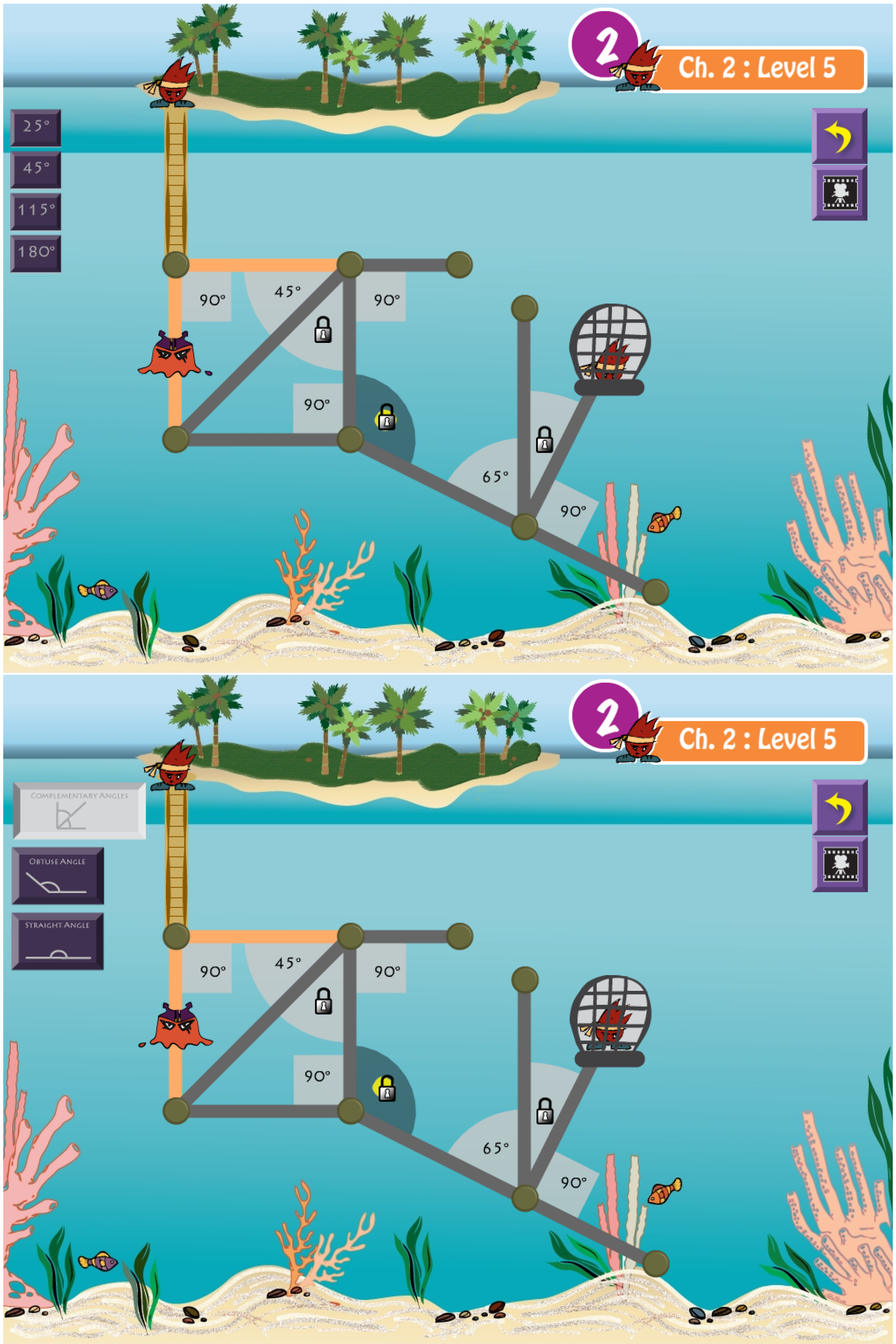


Figure 3.2: Screenshots of Noobs vs. Leets from Level 2.5. The arithmetic version (top) and rules version (bottom).

Part II

Augmented Reality Communication

Chapter 4

Augmented Reality Communication

4.1 Introduction

The human brain contains special-purpose regions for various physical tasks related to embodied communication such as recognition of facial expression, gaze and hand gesturing [11]. Cognitive psychologists have extensively documented the importance of embodiment in the presentation of information. A presenter‘s facial expressions help in capturing and maintaining audience engagement [21] while gaze and gesture help direct audience attention [64].

Presentation tools that augment physical gesture with real-time graphics are widely simulated in feature films from *Minority Report* to *The Hunger Games*, successfully capturing the attention and imagination of audiences worldwide. However, these effects are achieved through extensive post-production and thus remain speculative.

We are interested in enabling those who are experts in content rather than production to deliver richer presentations on their own areas of expertise. For that purpose, we developed ARCADE, a platform for real-time presentations with the following key features. ARCADE is an acronym for ”Augmented Reality Communication And Distance Education”.

1. For the audience, it conveys the illusion that the presenter is directly manipulating 3d virtual objects that appear to be holographically projected around him.

2. For the presenter, it enables delivery of augmented presentations without the need for post-production or a long rehearsal process.

3. The custom software that enables ARCADE is built for the Microsoft Kinect and other off-the-shelf components.

Our description of ARCADE will be organized as follows. First we situate our system in related work among interfaces that emphasize interpersonal communication, presentation styles and employ similar technologies. Second, we describe the setup and the technical framework of ARCADE. Third, we give examples that cover a space of content that benefits from this type of presentation and describe preliminary evaluations of ARCADE’s usability for the presenter and of its legibility for the audience.

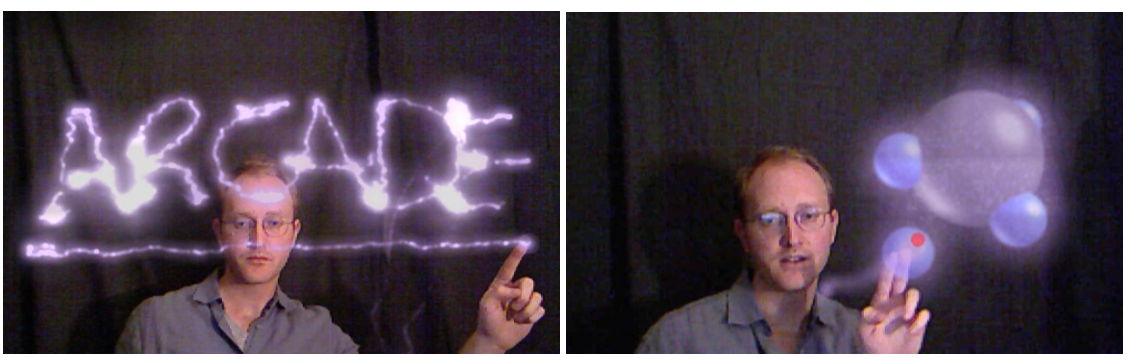


Figure 4.1: Two screengrabs from the audience point of view, demonstrating 3D in-air drawing (left) and manipulation of virtual molecule (right).

4.2 Related Work

ARCADE draws on the work of several fields, including augmented presentation tools, 3D interactive techniques, and markerless hand-tracking. Here we discuss the most relevant prior work that intersects the application of augmented reality and real-time interactive presentations.

Well before the modern computer era, there were attempts to convey the idea of real-time augmentation of gesture. In 1916 Windsor McKay presented live stage performances in which he created the illusion of interacting with, and even feeding an apple to, his animated character Gertie the Dinosaur [16]. Crockett Johnson‘s *Harold and the Purple Crayon* [39] is a well known children‘s book in which the main character draws entire virtual worlds with his magic crayon. Similarly, the children‘s cartoon series *Tennessee Tuxedo and His Tales* [6] features a 3D blackboard in which science lessons begin as drawings and then transform into 3D animated explanations.

4.2.1 Embodied Communication

Pioneering efforts to integrate information in space around a person‘s body to enhance communication include Myron Krueger‘s VideoPlace [44] and Hiroshi Ishii‘s ClearBoard [37]. More recently, Barakonyi et al. [8] demonstrated a video conferencing system using fiducial markers to position virtual information and used eye gaze for selection. E-Chalk [29] focused on improving traditional chalkboard teaching tools by integrating pictures and interactive programs from the web as well as providing archiving tools for saving real-time presentations. The CHARADE [9] system, provided gesture-based control of conventional slides using an instrumented glove.

4.2.2 Video-based Presentation Techniques

Recently, video-based presentations comprising a wide variety of techniques have become available on the web. We briefly discuss important landmarks among these. Vi Hart creates math videos which typically feature her drawing on a notepad, as if the camera were looking over her shoulder while she narrates [33]. Salman Khan has created over 3,000 videos which are among the most watched educational videos on the web using just a drawing application and a screencast of him as he draws and narrates [42]. These techniques are notable because they are cheap to produce, and have been widely emulated. In contrast to these approaches, Hans Rosling uses choreographed movements and post-processing, to demonstrate the changing population statistics of countries over time in space around him [68]. More recently, Marco Tempest has given live augmented reality "technological magic" shows by combining a modest amount of Kinect-based tracking with a reliance on pre-canned animations and carefully choreographed movements. Our work can be viewed as an attempt to free content-creators from this reliance on choreography and pre-canned animations through real-time tracking, while focusing on pedagogy, in the spirit of Hart, Khan, and Rosling.

4.2.3 3D Interactive Techniques

Beginning with Donald Vicker's Sorcerer's Apprentice [77] and VPL's DataGlove [86], there has been considerable research about how to effectively manipulate virtual objects in space [31]. Although our work is focused on demonstration rather than creation, many of the techniques used are similar, and so we recount these briefly here. Sketch Furniture by Front Design [5] is an artistic tool which combines 3D motion capture with rapid prototype printing to create pieces of furniture made from freehand sketches.

This work, like the related works of Takeo Igarashi [45], Steven Feiner et al. [28] and Yee & Lipson [85], focuses on (and are optimized for) interactive object creation and doing useful work in real-time. More recently, Wilson and Benko's LightSpace [82] demonstrated the use of multiple depth cameras and projectors to study the future of interactive physical spaces.

4.2.4 Markerless Hand-Tracking

Real-time markerless hand-tracking using multiple cameras or range data is an area of active research. For example, Schlattman and Klein [70] demonstrate a system capable of 6 degrees of freedom (DOF) per hand, plus recognition of 4 different hand configurations at interactive rates. Their technique computes the visual hull of each hand from three calibrated infrared cameras, and assumes that the hands are confined to an open box covered with infrared absorbing cloth for background subtraction. Wang and Popovic [80] demonstrated a database-driven approach using two Kinects that can detect bi-manual pinch gestures between forefinger and thumb as well as 6DOF for each hand. Recent work by Harrison et al. [32] demonstrated thumb and forefinger tracking using a custom depth sensor and a novel image processing pipeline based on horizontal edges. Their system requires that fingers have a fixed orientation (e.g. horizontal) with respect to the camera image, which is unsuitable for drawing in the air or manipulation of 3D objects. Oikonomidis et al [61] present a system capable of tracking 26-DOF of a single hand at 15fps using a Kinect depth camera. They model the hand as a configuration of joints, and use a GPU-accelerated energy minimization algorithm to find a configuration that minimizes the discrepancies between the model and the Kinect range data. In contrast to the above approaches, our approach runs efficiently at 30fps using commodity hardware, requires no calibration, and reliably tracks the 6DOF position/orientation of each hand, as well as detecting finger pointing and accurately tracking index fingers.

4.3 Setup

The ARCADE platform enables presenters to (1) pre-author content and (2) present this content live or record to video. For presentations, ARCADE runs on a computer with a display for the presenter and a Kinect facing the presenter, either on top or behind the display. The Kinect can be placed up to 50cm from the farthest point that the presenter can reach with their hands outstretched towards the Kinect, and can be moved up to 200 cm away, depending on the constraints of the presentation space, and the desire to increase viewing area around the presenter‘s body. To accommodate this setup during a presentation, the presenter could be seated at a desk or standing at a podium. During live presentations, a standard projector is required to display the audience‘s view of the presentation. For recording videos, no additional display is required.

4.3.1 Interface

The audience‘s view shows a live video feed of the presenter, who appears to be surrounded by interactive 3D objects. The *presenter dashboard* also contains the same live video feed as inset, but is shown in mirror reverse for the presenter to monitor his own actions. Additional information is overlaid on the presenter‘s view to give the presenter feedback on his actions and to guide his controls.

The dashboard also contains panels for control of the presentation. A list of demos at the bottom left allows the presenter to scroll through pre-authored content with keyboard arrows and select one to show. The properties at the top left shows parameters of the

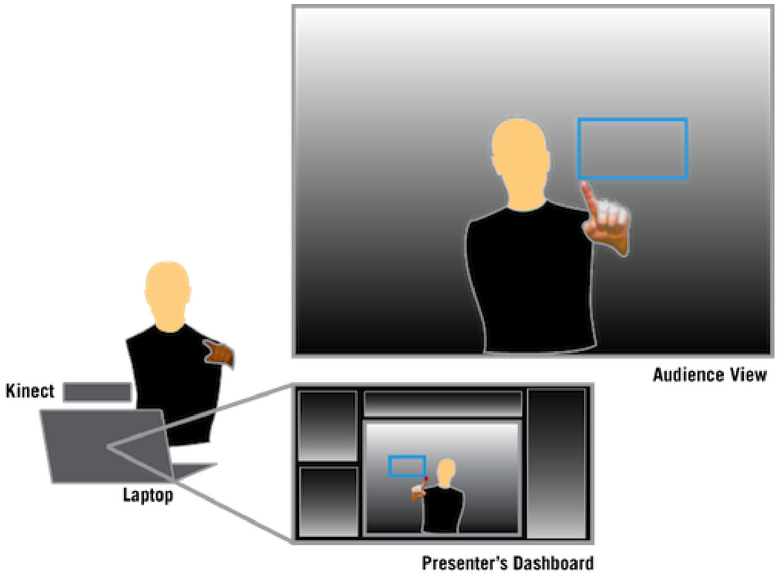


Figure 4.2: ARCADE setup showing presenter and audience views.

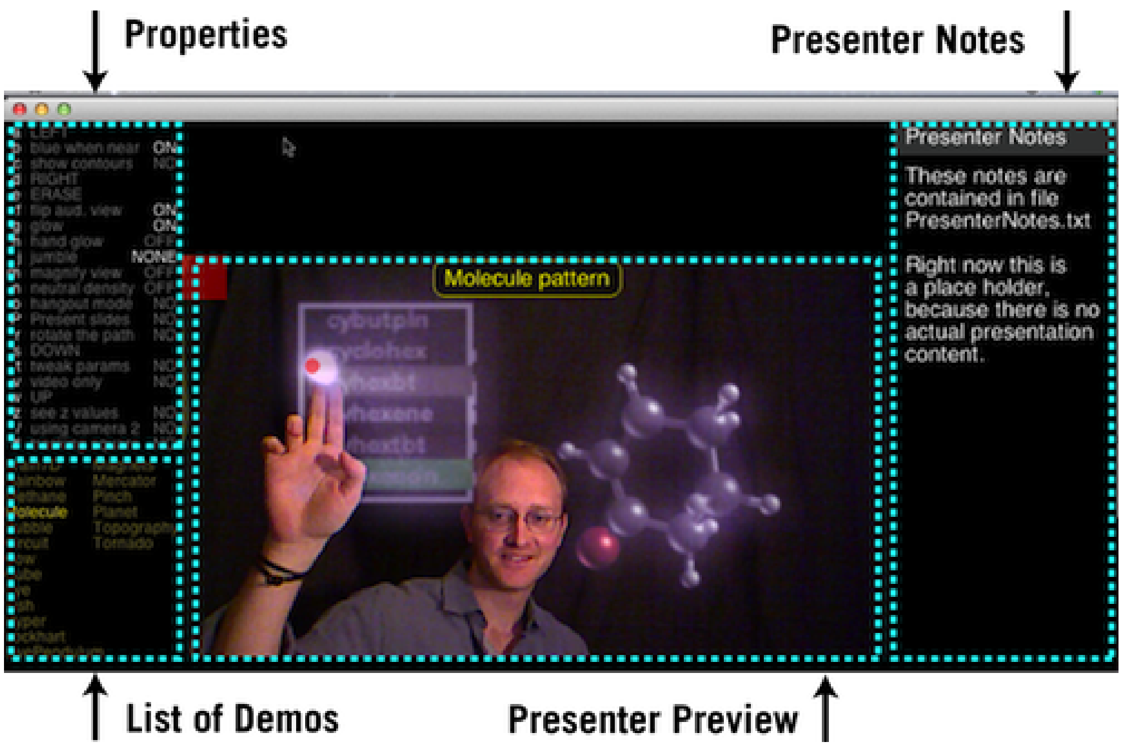


Figure 4.3: Presenter‘s dashboard

presentation, such as turning on or off a digital neutral density filter. The right hand side of the dashboard provides room for the presenter's reference notes (stored in a text file) to be used during the presentation.

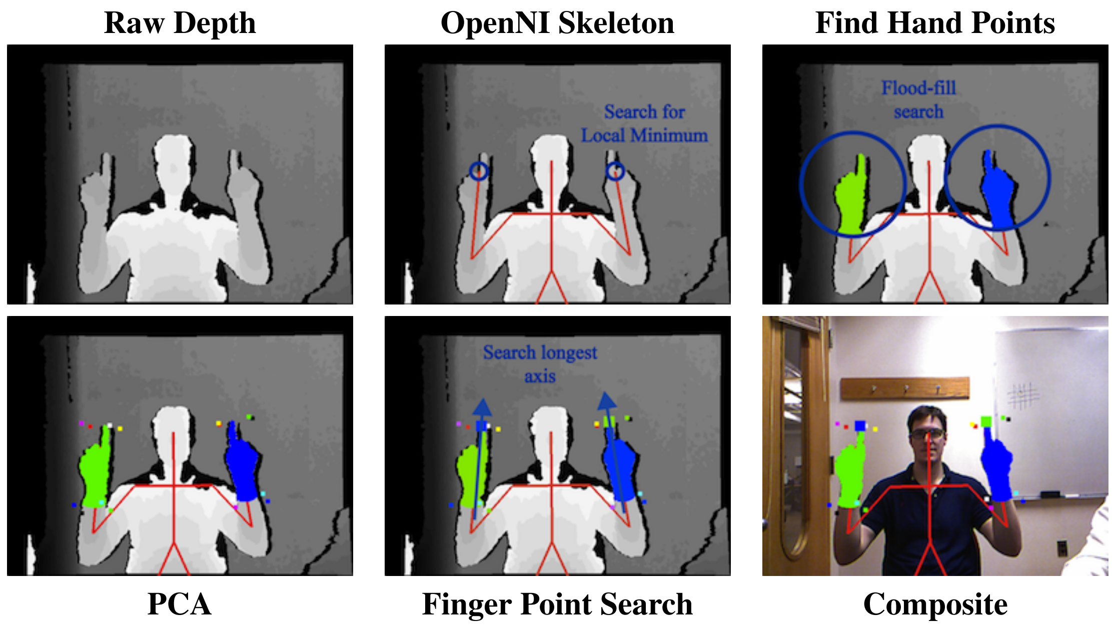


Table 4.1: Six stages of hand-tracking pipeline.

4.4 Implementation

At it's core, ARCADE is a software system that consists of a back-end for processing Kinect sensor data, and a front-end for creating and presenting content. The back-end uses OpenNI [36] to capture the calibrated depth, video, and skeleton frames and combines the depth and skeleton information to create a bounding box around each hand (see Hand Tracking below). The front-end consists of the following four parts

1. a framework for creating interactive content (described in Content Creation)

2. a gesture event system (described in Gesture Grammar)

3. a renderer to overlay visual effects in the projected 3D space of the video

4. a presentation mode which gives the presenter the ability to sequence content.

4.4.1 Hand Tracking

State-of-the-art skeleton tracking algorithms in depth images (OpenNI/NITE, Microsoft SDK) rely on the lower torso to stabilize the upper torso and joints, and exhibit significant instability when the lower torso is occluded. When giving live presentations, presenters often stand at a podium or behind a lectern, which makes tracking the lower body difficult in practice. Since accurate finger positions is a requirement for our system, we created a simple and high frame-rate pipeline to extract a single high-quality 3D finger point as well as an estimated hand orientation for each hand, which serves as a combined 3+3=6DOF input to the system. This section describes the details of the pipeline, as depicted in Table 1.

The Skeleton tracking data includes a 3D point for each hand, which is centered approximately in the middle of the hand whether it is opened or closed. Since this hand point exhibits noise and often lies in regions that are not part of the hand geometry (between the fingers for instance), we need to first find a stable pixel in the depth image that with high confidence lies on the hand. We do so by searching within some small constant radius in the real world coordinate system (10mm), performing the actual search on the projected depth image (with an appropriately scaled radius in pixel coordinates) in order to find the pixel in the point cloud closest to the Kinect. Since other objects rarely occlude the hand when performing gestural interactions with AR, we assume that the hand is closest to the camera within some small radius of the skeleton hand point, which works well in practice.

Using this high confidence hand point, we do a flood fill in the depth image to determine all the points which lie within a fixed radius of the seed point (200mm). Due to noise in the depth image, and the tendency for the hand to self-occlude, we allow the flood-fill to "jump over" small patches of missing depth regions and include pixels which are not part of the same connected component. This is implemented by testing not only the connectedness of the 8 surrounding neighbors of each pixel in the flood fill, but additional pixels at fixed locations further away. The depth points are assigned to either the Left Hand, the Right Hand, or neither.

We then calculate the covariance matrix of the point cloud and then perform Principle Component Analysis (PCA) to find the dimensions of greatest variance. These statistics are then used to fit an oriented bounding box to the hand's point cloud. Since the optimality of the bounding volume is dependent on the internal distribution of the point cloud [20] we can only derive a rough estimate for the orientation of the hand from the orientation of the bounding volume. However, since the direction of the principal component has temporal coherence at 30fps, we find that in practice we can use the estimated orientation over the last 100 frames in order to detect coarse gestures reliably, despite these limitations on accuracy.

Finally, a finger point is found for each hand by projecting the hand points onto the longest axis to find the two extremal points at each end of the bounding volume. Since the eigenvectors of the covariance matrix (calculated from PCA) have sign ambiguity (the negative of the eigenvectors from SVD are also themselves eigenvectors), both extremal points must be considered as finger points. To decide which point is the most probable finger point, we perform a small search in the depth image for the two candidates and count the number of close neighbors to each. The candidate point with fewer close neighbors is then chosen as the finger point since the incorrect point typically lies on the wrist, which is densely surrounded by other range points.

Using the pipeline described above, each hand becomes a 3DOF input, yielding 6DOF in all. Although significantly more than 3DOF is contained in the bounding box fit over each hand, in practice these are much too noisy to depend on for interactions other than coarse mode switching (hand waving gestures for instance) where signal amplitude is high.

4.4.2 Gesture Grammar

Using the 3DOF from each finger point, a large number of appealing control gestures are possible [15]. As our application is a presentation tool, we found it necessary to optimize for gestures that were simultaneously intuitive to the presenter and visually appealing to the audience. In this section, we discuss the grammar vocabulary that resulted from this optimization process. We assume a right-handed coordinate system in which positive Z comes out from the screen towards the presenter, positive Y goes up towards the sky, and positive X is to the right. When referring to the object‘s bounding box, the sides of the bounding box are always axis-aligned, and oriented with respect to the world coordinate system, not the object‘s local coordinate system.

3D Drawing: This is a bimanual interaction allowing the presenter to start and stop drawing a line in 3D space without using external signals such as buttons or keypresses. Drawing begins when the presenter touches the pointer fingers of both hands together. The finger that moves out of this circle first, is the active drawing finger and emits scintillating particles in a tight region around it. Drawing stops by touching both fingers together again. This interaction allows the presenter to draw with either hand, and to stop bringing the inactive finger into contact with the drawing finger, or by making a loop whereby the inactive finger is held (approximately) stationary and the active finger returns to it.

Drag: Drag consists of three phases, 1) grab, 2) move, and 3) release. The grab is initiated by placing either the left or right forefinger inside the object. At this point, the object glows or changes color to signify it is active. It then stays positioned over the user‘s finger as they move it around in 3D. Finally, to release, the user places their other hand inside the object.

Turntable spin (Y Spin): This gesture is akin to turning the object on a turntable and is mathematically equivalent to a rotation about the object‘s central Y axis. It is initiated by entering the object‘s bounding box from either the left or right side, and responds continuously to the horizontal position of the finger. The rotation is scaled so that travelling through the horizontal length of the object corresponds to 180°. The gesture ends when the user‘s finger exits the object‘s original axis-aligned bounding box, which was computed when the gesture began.

X Spin: This gesture is akin to rotating the object along its central X axis. It is identical to the turntable spin, but is initiated by entering the top or bottom faces of the object‘s bounding box, and is driven by the finger‘s vertical position, rather than horizontal position.

Uniform Scale and Z spin: This is a bimanual gesture that is initiated by placing the tracked finger from each hand inside the object. Moving the fingers apart in X or Y creates a proportional, uniform scaling of the object. While moving the fingers around the circle in the Z-plane, effects a rotation around the Z-axis. To release the gesture, the user simply moves either finger at least 5 cm in Z towards or away from the starting depth of that finger. Scaling the object down so that it is invisible is equivalent to a delete operation.

Menu selection and scrolling: This is a single finger interaction which enables the user to select items from a menu. Selecting consists of placing the finger over a menu item. Scrolling is effected by placing the finger at the top or bottom edge of the menu. Selection is effected by moving the finger to the left or right horizontally. Visually, the menu item moves with the finger position and after sufficient travel highlights to show selection.

Delete or Dissolve: When the presenter waves his whole hand behind an object for a few seconds the system detects this as a request to remove that object from view. Visually this gesture looks like erasing on a whiteboard or a blackboard with the fingers splayed outwards as if the presenter were holding a cloth. It also looks like waving goodbye in westernized cultures. As the object disappears, it appears to turn into particles which then gradually fade away.

4.4.3 Supplemental Inputs

In addition to the in-air gestures mentioned above, we have experimented with several conventional input devices such as a keyboard, toe-clickers, and foot-pedals, to add extra functionality. For example, the keyboard can be used to select content to show, which is useful theatrically if the presenter wants to hide the choices they are making. It can also be used as a simple way to "click" instead of using both fingers. In practice, glancing down at the keyboard is unappealing to the audience and it is difficult to find most small keys on the keyboard without looking. This restricts the use of a standard computer keyboard to the spacebar, the Return key, and perhaps a few other keys on the periphery. A promising alternative that we have explored is the use of a small musical keyboard with full-size keys that we have found to be easy to use without looking. As a final downside of a keyboard approach in general, it severely limits the use of bimanual gestures.

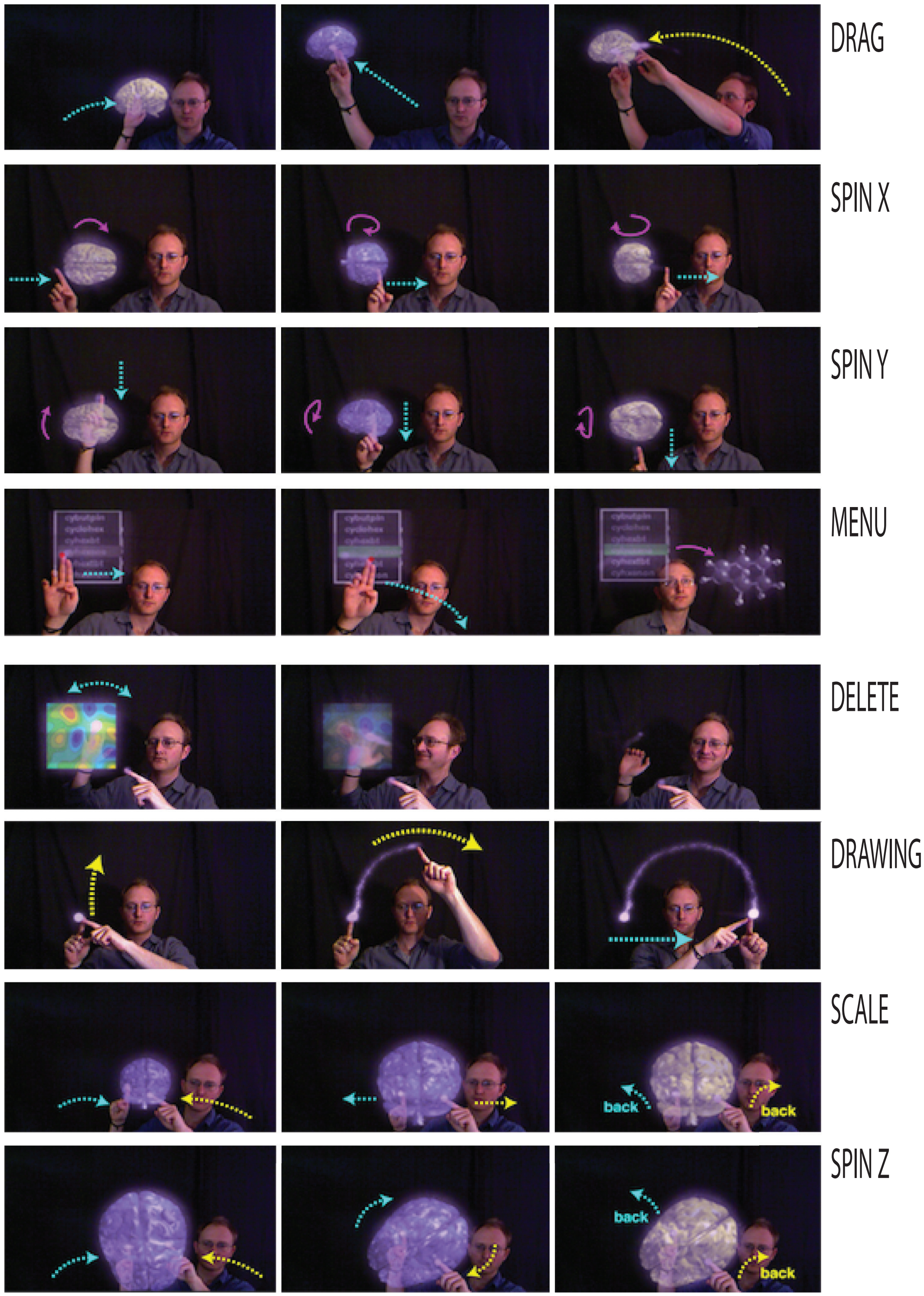


Figure 4.4: Gesture grammar of ARCADE applied to manipulating a 3D brain model. Yellow and blue arrows indicate motion of each hand. Pink arrows indication motion of virtual object (brain).

As people are accustomed to using both their hands and feet at the same time (e.g. driving, playing musical instruments) we have also investigated the use of foot pedals and wearable toe-clickers that give one or two additional signal sources (like keys on a keyboard) without restricting bimanual interaction. To test this out in practice, we developed a toe-clicker that is rigidly affixed above the big toe in a shoe. The toe-clickers use ZigBee for wireless communication allowing the use of two toe-clickers, one on each big toe. In practice, the only real drawback to this approach is the annoyance of having to put on the specialized device, and wear it.

4.4.4 Content Creation

ARCADE currently supports the ability to import standard 3D models and images and manipulate them using the gesture grammar mentioned above. It also supports the ability to sequence pre-existing content and step-through it linearly, or dynamically present material in any order. However, to make new interactive animated content, significant programming expertise is required. This is a limitation of our current implementation. Specifically, knowledge of our in-house software renderer, as well as an ability to do object-oriented programming in Java are prerequisite. A natural direction for future work is to make it easier to make new interactive content by integrating a version of a snap-together tiles language that have been shown to be usable by even young children [51, 1, 40]. In practice, the need to program is not necessarily a limiting factor for non-programmers to do sophisticated presentations with ARCADE because it is possible to use someone else‘s pre-made content. This is a common tactic in popular presentation formats, and is referred to as sharing your “slide-deck”.

4.5 Applications

Given the ability to present 3D virtual objects in space around the presenter, and control these objects with 6DOF for each hand, the amount of possible content is vast. During the course of development, we created a large number of self-contained examples, called *patterns* which we demoed live to large (auditorium-sized), and small (group-sized) audiences. This provided informal feedback which we incorporated into our iterative design process. For example, this process informed aesthetic choices, such as how to effectively use rendering and visual effects, and core design decisions, such as the need to incorporate a moving cursor for the audience to see which areas of the presenter‘s hands are active for gestures. We discuss a number of the representative patterns which grew out of this process below.

4.5.1 7DOF manipulation of 3D Models

By default any object in ARCADE can be manipulated using the built-in gesture grammar to change its position in space (3DOF), its rotation (3DOF), and its overall scale (1DOF), yielding 7DOF overall. As one example, this allows for a simple way to show the important landmarks of a human brain for a neuroscience lesson. By zooming in on the brain, the presenter can be very specific. As a second example, a topographic map can be shown from above, and then easily rotated to show the true 3D undulations of the terrain. As translation is not important for either of these demonstrations, this can easily be disabled. See Figures fig:ar-brain and fig:ar-map for screenshots from the brain and the topographic map demonstrations, respectively.

Figure 4.5: Screenshot of presenter rotating a 3d model of the human brain.

4.5.2 Smart Objects

Additionally, ARCADE supports more advanced interactions that can be specified by programming “smart” objects to respond to gesture with specialized behaviors. This is similar to procedural animation found in games which react differently depending on user actions. For example, we have made a pendulum which can be pulled and swung as if it were undergoing ordinary physics, but it has the additional behavior that the pendulum rod can be lengthened by holding the top of the pendulum and pulling down on its bob, or by pressing a key on the keyboard. This enables a presenter to directly demonstrate the fact that the pendulum’s period is proportional to the square root of the pendulum rod length. The Pendulum pattern augments the default gesture grammar by “knowing” how to lengthen the rod in response to the bimanual gesture just mentioned. It is worth noting that a useful side effect of using a depth camera in this work is that, the virtual pendulum can be rendered according to real-world coordinates. This allows

Figure 4.6: Screenshot of presenter rescaling a 3d topographic map.

the presenter to hold up a real ruler and measure the pendulum rod length just as if it were a “real” pendulum.

Another example of a smart object with additional interactions is the Sorting pattern, which is a pattern we built to explain the difference between various Exchange sorting algorithms in computer science. It features a row of 10 rainbow colored balls which mix themselves up when the pattern loads and responds to two custom gestures based on the positions of the presenter‘s tracked fingers. First when the presenter places a tracked finger within 2 cm of the bottom of a ball (as if pointing at the ball from below), the ball bounces up and down to draw attention. Second, when the presenter places his second finger underneath a ball (also within 2 cm) the balls swap places. In this way the presenter can easily demonstrate the difference between several sorting algorithms, such as Selection Sort and Bubble Sort [19]. The Sorting pattern’s interaction is shown in Table 4.2.

4.5.3 Tangible Objects

Using custom-programmed patterns, ARCADE presentations can integrate tracked tangible objects with virtual graphics. As an example, we have created a pattern to explain magnetic field lines, where the presenter holds up two magnets and floating field lines follow them where they are moved. This allows the presenter to make the invisible visible and is a potential advantage over "real" demonstrations. The magnets, and other small rigid objects appear as extensions of the presenter's hands to the back-end tracking system, which means no additional work is required to track them.

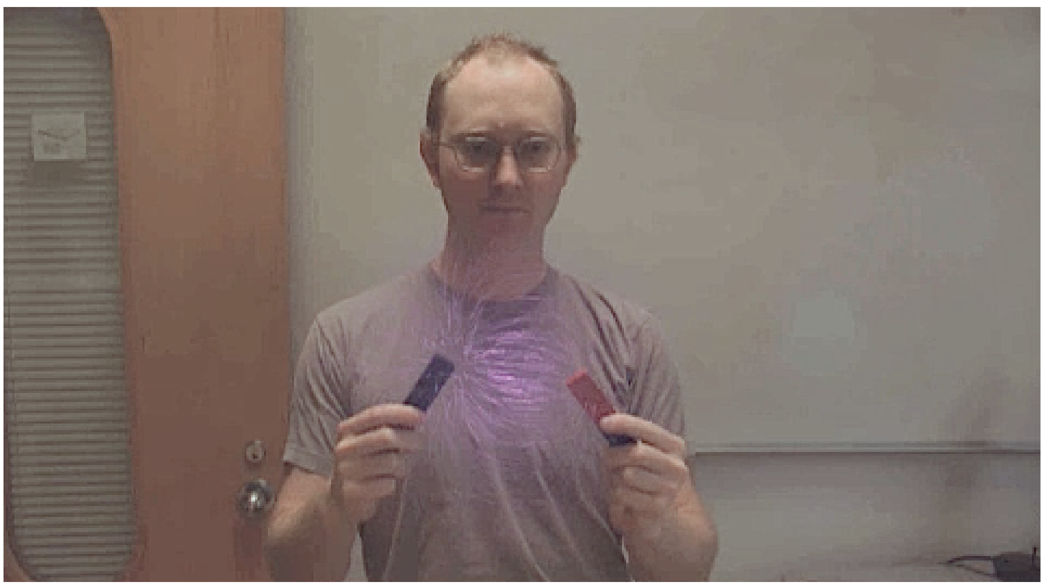


Figure 4.7: Screenshot of presenter holding actual bar magnets, with virtual magnetic field lines superimposed in the gap between them.

4.6 Evaluation

For ARCADE to be a useful live presentation tool it must be easy for the presenter to use and similarly legible for the audience. In this section we first discuss results from early-stage evaluations of the usability and learning curve of the presenter interface for new users. We then present findings from a study where we collected audience opinions on the same lesson presented in four alternative formats. This work was carried out by myself and Xiao Xiao, from Hiroshii Ishii's Tangible Media Group Laboratory at the MIT Media Lab.

4.6.1 Presenter Studies

Central to the design of ARCADE is the placement of virtual objects superimposed on the physical space around the presenter's body. To manipulate virtual objects, the presenter must be able to quickly and accurately navigate this virtual space, which requires that the interface to provide feedback on the presenter's finger positions in a form that is easy to understand. We conducted two informal studies on the presenter's ability to use ARCADE's tracking feedback. In particular, we investigated the following research questions:

- How difficult was it for users to use the feedback they were given to manipulate objects in the virtual space?
- How long does it take for a new user to get used to the interface?

Pilot Study — Procedure We collected qualitative feedback from 9 new users (7 male, 2 female; Ages 22-36, median 27, mean 24) in a pilot study. After users were given a chance to familiarize themselves with ARCADE tracking, they were presented with a modified version of the Sorting pattern with a row of eight white balls and were asked to swap the two highlighted balls by pointing at them from below. We chose this task because it was unambiguous how to measure performance, and was expected to be the most difficult task in ARCADE because of the degree of accuracy required. After

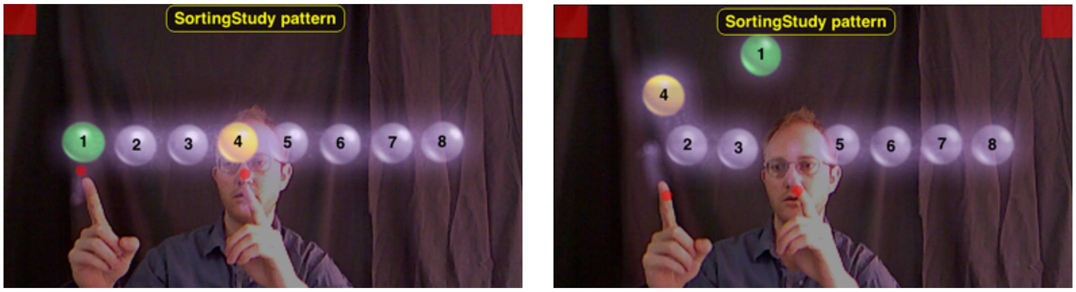


Table 4.2: Screencaptures showing the presenter's view of the ball sorting task during the study. The left view shows the screen just before swapping balls (1) and (4), and the right view shows the swap animation in progress.

2 rounds of 20 ball swaps each, we debriefed each participant to discuss how they got used to the system, what cues they used to navigate themselves within the ARCADE environment, and their suggestions for improving the interface.

Pilot Study — Results In general, participants felt comfortable navigating within the ARCADE environment after a few minutes using the system. None of the participants felt that speaking significantly added to the difficulty of using ARCADE. Several of the users reported that habituation required calibrating the distance they move in the physical world with the corresponding distance in the virtual world, which took a few practice tries to develop. About half the users also reported a small but noticeable lag between moving in real life and seeing the action captured on screen (up to 60ms in practice), which was more difficult to get used to.

The presenter's interface displays the mirror image video of the user with red dots indicating the positions of the index fingertips. Participants stated that the Kinect video stream and the red dot were useful and provided enough visual feedback in helping them position their hands in the virtual world. However, many participants suggested that the interface needed more feedback to indicate when an action will happen. For example, many of the mistakes were accidental triggers due to users not moving their hands low enough when going toward a pair of selections.

Timing Study — Procedure We noticed during the pilot study that because participants were not given a chance to experiment with the behavior of the swapping balls before the trial, many made mistakes due to a lack of understanding of the mechanics of the interaction. To collect more accurate data on how long it takes for new users to gain accuracy manipulating virtual objects with the presenter interface, we ran a second study where we recruited 19 new participants (8 male, 11 female; 18-46, average age 23.3, median 23). 9 had no prior experience using the Kinect, 6 were recreational users and 4 were developers.

For this study, we asked participants to perform the same swapping task as the pilot study while announcing the numbers that they were swapping. This time, participants were given 5 experimental swaps in the beginning along with some suggestions of what to try (e.g. selecting both together or one at a time). After the test swaps, participants were asked to swap 30 more pairs of balls. We recorded how many milliseconds it took for each swap. If the participant swapped incorrect balls, a "mistake" was recorded.

Timing Study — Results While users of all experience levels had similar average accuracy rates, with Kinect novices at 3.3 mistakes (11%), gamers at 2.67 (9%) mistakes and developers at 3 (10%) mistakes, a marked difference can be seen in the speed of completion based on experience of user with the Kinect. In general, Kinect novices took longer to perform the swap task, averaging 5.6 seconds per swap while more experienced users averaged 4.6 seconds for gamers and 4.6 seconds for developers, respectively. The average swap time of a member of our team, who is now a seasoned ARCADE presenter is 3.9 seconds.

The results of experienced Kinect users do not appear to improve very much over the course of the trial, while the results of Kinect novices do seem to improve with practice.

After 15 or so swaps, they take less time to perform the task with a much higher accuracy rate. These results suggest that those experienced with the Kinect can very quickly navigate themselves with ARCADE with reasonable speed and accuracy. Even those without previous experience with Kinect learned to calibrate their movements during the course of the trial.

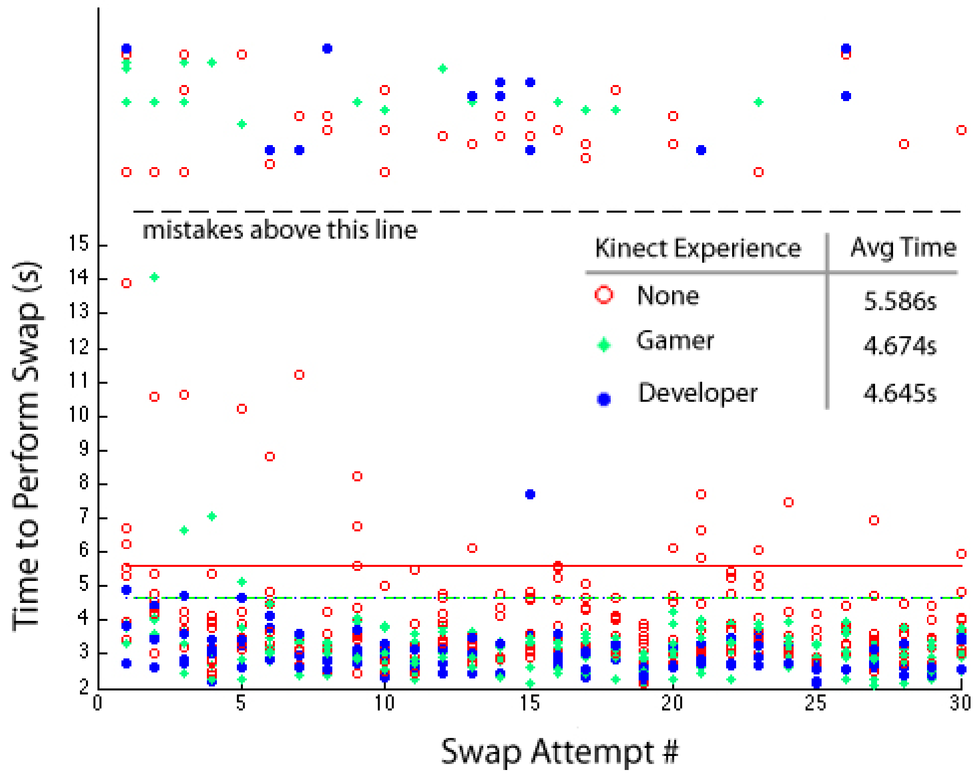


Figure 4.8: Graph of the time it took for 30 swap attempts for each subject. If a mistake was made, the point is plotted above the mistake line. Each subjects' mistakes are offset vertically for better visibility. Mean times for each experience level are plotted as horizontal lines.

4.6.2 Audience Study

From the audience perspective, ARCADE-style presentations feature a presenter shown in the same space as the virtual objects, which allow them to cue audience attention with their gestures and directly manipulate objects. We conducted a qualitative study to understand how this style of presentation is perceived by audiences. To do this we created four versions of an instructional video about pendulum physics. In each video, the presenter explains how the length of the pendulum is mathematically related to its period of oscillation, and demonstrates this relationship using a virtual pendulum. The pendulum swung when the bob was pulled to one side and let go, and the length of the pendulum was modified when the bob was selected and moved up and down. The videos varied along two dimensions: *narrated* vs. *anchored*, and *interactive* vs. *annotated*.

In the two *narrated* videos, the pendulum animation is projected on a black background, and the presenter's voice narrates but he is not seen. Instead he moves a brightly lit cursor on screen. Whereas, in the two *anchored* videos, the pendulum is superimposed in front of the presenter, who is seated in front of a white wall, and no cursor is shown. In the two *interactive* videos, the pendulum moves under the direct control of the presenter. Whereas in the two *annotated* videos, the pendulum moves according to a pre-recorded sequence. In the narrated/annotated video the presenter annotates the animation of the video with his cursor. In the anchored/annotated video, the presenter annotates by gesturing with his hands. The animation of the pendulum in the interactive versions was reused and replayed *exactly* for the two annotated videos. Screenshots from the same moment in each video are shown in table 4.3.

The narrated versions are similar to videos created in the style of Salman Khan, and the anchored interactive video is in the style of ARCADE.

Procedure We recruited 10 participants from MIT, 5 of whom were male. Their age

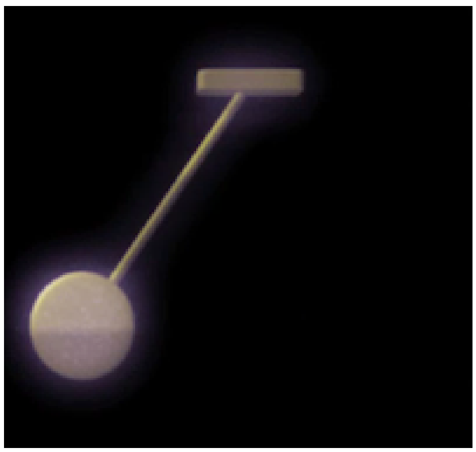
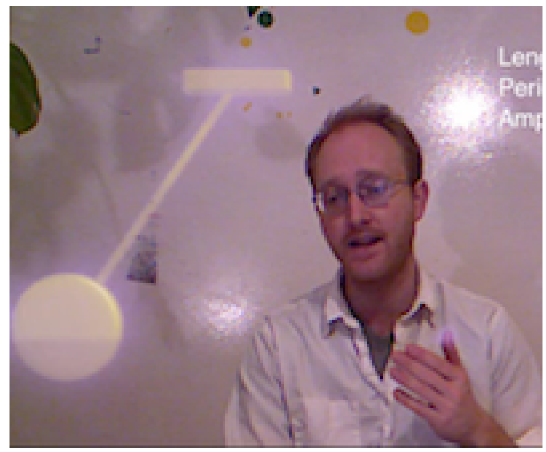
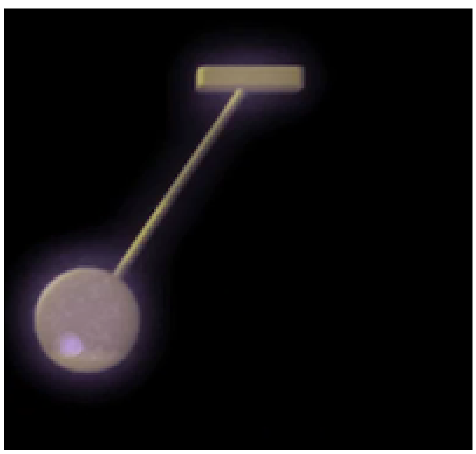
	Narrated:	Anchored:
Annotated		
Interactive		

Table 4.3: Four versions of the same presentation about pendulum physics, varying along two independent dimensions.

ranges were 21-26. The mean age was 27.3 and the median age was 26. Each participant watched the four videos individually. The order of the videos was randomized for each participant. After showing all four videos, we asked participants to rank them from 1 to 4 and to explain the reason for their rankings.

	Rank			
	1st	2nd	3rd	4th
Narrated/Annotated	3	1	2	4
Narrated/Interactive	1	3	3	3
Anchored/Annotated	0	4	3	3
Anchored/Interactive	6	2	2	0
Total	10	10	10	10

Table 4.4: Audience rankings of the 4 types of videos. The table shows the number of participants who indicated each rank per video.

Results Six out of 10 participants ranked the ARCADE-style presentation of presenter with direct manipulation as their favorite (see Table 4.4). Participants noted that the presentation was "more captivating since there's there's someone showing it" and that "seeing the person with facial expressions constantly changing kept me focused even when I did not find the topic interesting". One participant was a non-native english speaker and he indicated that being able to see the presenter was crucial for his understanding of the material because he depended on visual information like facial expression and body gesture.

The other 4 participants chose one of the narrated versions as their favorite. However, only one person in ten had a hard preference for not seeing the presenter. She said she liked seeing the presenter, but she had difficulty focusing on what she heard. The three other participants indicated that they would prefer seeing the presenter for longer presentations on more complex material, if the video had a less cluttered background, or if the presenter looked more at the camera and at what he was demonstrating.

All 10 users preferred the interactive version when the presenter was shown. Participants noted that "direct manipulation highlights the object that you're acting on, draws your attention, and is easy to understand." Opinions were split evenly on whether interactivity was helpful in the narrated videos.

Although our results are not statistically significant due to the small sample size, they do suggest that the ARCADE-style of embodied presenter directly manipulating virtual objects could be a compelling way of delivering information. The audience study also raises the question of ARCADE's dependence on presenter skill and its lack of support for eye contact, limitations that we will detail in the discussion.

4.7 Discussion and Future Work

We discuss ARCADE with respect to three aspects of presentations. For each, we detail design tradeoffs, lessons learned and limitations of our current implementation.

4.7.1 Engagement vs. Distraction

A key feature of ARCADE is that the video of the presenter is displayed in the same space as the visual information he or she is presenting, such that the audience can better see the relationship between the presenter's body motions and the animated information. While our study suggests most people find the presentation more engaging with the presenter featured as prominently as the presented information, a portion of people seem to find the presence of the presenter distracting.

Audience engagement is sensitive to the presenter performance, which includes facial expressions, gaze and posture. For instance, an unenthusiastic presenter who never looks at the audience does more damage when visible than when not in view.

One limitation of our current system is its lack of support for eye contact. To "make eye contact with the audience", the presenter must actually look toward the Kinect. Since eye contact with the audience only needs to be made from time to time, the presenter interface could display reminders for the presenter to look toward the Kinect, especially during key points of a demonstration.

Our current system also does not provide gaze correction. Because the presenter must be constantly monitoring the screen for visual feedback, he rarely looks where virtual objects appear. As gaze is an important cue to help audiences look toward focal points, we are currently experimenting with interfaces that enable the presenter to appear with correct gaze even when looking at the screen. Unfortunately, the classical beamsplitter-setup used in teleprompters would not work unless the mirror is as large as the presenter's field of view – a costly requirement.

4.7.2 Rehearsal Time vs. Authoring Time

In earlier prototypes, we experimented with ARCADE content controlled by supplemental inputs where the presenter simulates interaction with virtual objects with correctly timed theatrical gestures. However, achieving correct timing was difficult to master even after many rehearsals. In ARCADE objects are controlled by fully tracked gestures, which significantly decreases the rehearsal time.

However, to use gestural interactions outside the current gesture grammar would require significant programming time. An important aspect of future work is designing authoring tools to enable creation of content and controls without sophisticated programming knowledge.

4.7.3 Scripted vs. Improvised

Unlike presentations that rely on canned animations for theatrical gestures that must faithfully follow a script during runtime, ARCADE allows presenters some freedom for some improvisation. The presenter can vary the way he explains concepts to different audiences because he does not have to adhere to a precise timeline of actions.

To give presenters more freedom of expression during runtime, we plan to improve the tracking of our system to decrease latency and add more feedback for performing different types of gestures. By doing this, presenters can have better control of the virtual content so that they can be more expressive with the medium.

We also plan to iterate on our design of the presenter's interface, adding and modifying existing side panels to provide more useful information during the presentation.

4.8 Conclusion

We discussed ARCADE, a platform enabling real-time presentation of interactive 3D graphics controlled in mid-air by the presenter. Avoiding post-processing gives presenters the opportunity to change the structure of their presentation during run-time, allowing interactive question and answer sessions, and on-demand exploration of certain topics areas. This work creates a natural progression towards interactive two-way communication between a teacher and her students where not only the teacher but also the students are able to interact with information represented as 3D virtual objects. We anticipate that students would like to like to take a hands-on approach and become presenters themselves.

Part III

Real-Time Hand Pose Estimation

Chapter 5

Real-time Hand Pose Estimation

5.1 Introduction

Inferring the pose of articulable objects from depth video data is a difficult problem in markerless motion capture. Requiring real-time inference with low-latency for real-time applications makes this even harder. The difficulty arises because articulable objects typically have many degrees of freedom (DOF), constrained parameter spaces, self-similar parts, and suffer from self-occlusion. With the advent of commodity depth cameras, the problem is made still harder because depth sensors are usually inaccurate near depth discontinuities, which is a common feature of most views of an articulable object. All these factors make fitting a model directly to the depth data hard, and even undesirable in practice, unless the fitting process is able to account for such missing data.

One common approach to ”fill in” missing data is to combine multiple simultaneous video streams; but this is a costly demand on the end-user and may prohibit widespread use of otherwise good solutions. A second common approach, called ”supervised learning” in computer vision and machine learning, is to train a model on ground truth data, which combines the full pose of the object in the frame with the depth image. The trained model can then use a priori information from known poses to make informed guesses about the likely pose in the current frame.

Large ground truth datasets have been constructed for important articulable objects such as human bodies, and robust real-time pose inference systems have been trained on them using supervised learning. Unfortunately, most articulable objects, even common ones such as human hands, do not have publicly available datasets, or these datasets do not adequately cover the vast range of possible poses. Perhaps more importantly, it may be desirable to infer the real-time continuous pose of objects that do not yet have such datasets. The vast majority of objects seen in the world fall into this category, and a general method for dataset acquisition of articulable objects is an important contribution of this work.

The main difficulty with using supervised learning for our problem is in obtaining ground truth data for hand pose. Typical models of the human hand have 25-50 degrees of freedom [26] and exclude important information such as joint angle constraints. Since real hands exhibit joint angle constraints that are pose dependent, faithfully expressing such limits is still difficult in practice. Unfortunately, without such constraints, most models are capable of poses which are anatomically incorrect. This means that sampling the space of possible parameters using a real hand is more desirable than exploring it with a model. With the advent of commodity depth sensors, it is possible to economically capture continuous traversals of this constrained low-dimensional parameter space in video, and then robustly fit hand models to the data to recover the pose parameters [62].

In this chapter, we present a solution to the difficult problem of inferring the continuous pose of a human hand by first constructing an accurate database of labeled ground

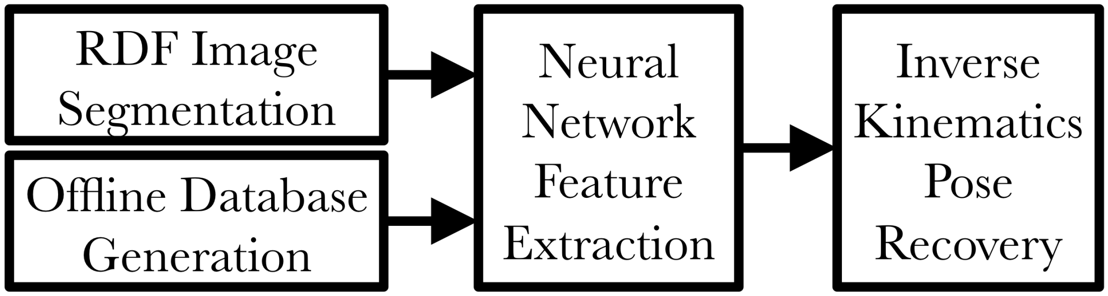


Figure 5.1: Hand Pose: Pipeline overview

truth data in an automatic process, and then training a system capable of real-time inference. Since the human hand represents a particularly difficult kind of articulable object to track, we believe our solution is applicable to a wide range of articulable objects. Our method has a small latency equal to one frame of video, is robust to self-occlusion, requires no special markers, and can handle objects with self-similar parts, such as fingers. To allow a broad range of applications, our method works when the hand is smaller than 10% of the image frame.

Our method can be generalized to track any articulable object that satisfies three requirements: 1) the object to be tracked can be modeled as a 3D boned mesh, 2) a binary classifier can be made to label the pixels in the image belonging to the object, and 3) the projection from pose space (of the bones) to a projected 2D image in depth is approximately one-to-one. The model is used to automatically label depth video captured live from a user. This data is used to train a Randomized Decision Forest (RDF) architecture for image segmentation as well as a ConvNET to infer the position of key model features in real-time. We also suggest a simple and robust inverse kinematics (IK) algorithm for real-time, high degree of freedom pose inference from the ConvNet output. The system can accommodate multiple commodity depth cameras for generating training data, but requires only a single commodity depth camera for real-time tracking.

We believe the key technical contribution of this work is the creation of a novel pipeline for fast pose inference, which is applicable to a wide variety of articulable objects. An overview of this pipeline is shown in Figure 5.1.

As a single example, training our system on an open-source linear-blend-skinning model of a hand with 42 degrees of freedom takes less than 10 minutes of human effort (18,000 frames at 30fps), followed by two days of autonomous computation time. Tracking and pose inference for a person's hand can then be performed in real-time using a single depth camera. The trained system can be readily used to puppeteer related objects such as alien hands, or real robot linkages, and as an input to 3D user interfaces [73].

Since objects in the real-world exhibit temporal coherence in pose space on small time scales, direct-search algorithms such as particle swarm optimization (PSO) have been used successfully for pose recovery on models with more than 50 DOF [62]. Unfortunately, temporal coherence cannot be relied on for robust real-time tracking since dropped frames and fast moving objects typically break this temporal coherency assumption. In this work, we present a solution that uses direct search to create a database of poses, and a novel ConvNet and IK architecture that learns pose inference for articulable objects from a single frame, negating the need for any temporal coherency assumption.

5.2 Related Work

A large body of literature is devoted to real-time recovery of pose for markerless articulable objects, such as human bodies, clothes, and man-made objects. As the primary contribution of our work is a fast pipeline for recovery of the pose of human hands in

3D, we will limit our discussion to the most relevant prior work.

Many groups have created their own dataset of ground-truth labels and images to enable real-time pose recovery of the human body. For example, Wang et al. [79] use the CyberGlove II Motion Capture system to construct a dataset of labeled hand poses from users, which are re-rendered as a colored glove with known-texture. A similar colored glove is worn by the user at run-time, and the pose is inferred in real-time by matching the imaged glove in RGB to their database of templates [78]. In later work, the CyberGlove data was repurposed for pose inference using template matching on depth images, without a colored glove. Later, Shotton et al. [72] used randomized decision forests to recover the pose of multiple bodies from a single frame by learning a per-pixel classification of the depth image into 38 different body parts. Their training examples were synthesized from combinations of known poses and body shapes. In similar work, Keskin et al. [41] created a randomized decision forest classifier specialized for human hands. Lacking a dataset based on human motion capture, they synthesized a dataset from known poses in American Sign Language, and expanded the dataset by interpolating between poses. Owing to their prescribed goal of recognizing sign language signs themselves, this approach proved useful, but would not be feasible in our case as we require unrestricted hand poses to be recovered.

Several other groups have used domain-knowledge and temporal coherence to construct methods that do not require any dataset for tracking the pose of complicated objects. For example, Wiese et al. [81] devise a real-time facial animation system for range sensors using salient points to deduce transformations on an underlying face model by framing it as energy minimization. In related work, Li et al. [48] showed how to extend this technique to enable adaptation to the user's own facial expressions in an online fashion. Melax et al. [53] demonstrate a real-time system for tracking the full pose of a human hand by fitting convex polyhedra directly to range data using an approach inspired by constraint-based physics systems. Ballan et al. [7] show how to fit high polygon hand models to multiple camera views of a pair of hands interacting with a small sphere, using a combination of feature-based tracking and energy minimization. In contrast to our method, their approach relies upon inter-frame correspondences to provide optical-flow and good starting poses for energy minimization. Oikonomidis et al. [62] demonstrate the utility of PSO for tracking single and interacting hands by searching for parameters of a model that reduce the reconstruction error of a z-buffer rendered model compared to an incoming depth image. In contrast to their work, which used PSO directly for interactive tracking on the GPU at 4fps, our work shows that PSO is an invaluable *offline* tool for generating labeled data, and we demonstrate the use of supervised learning techniques for real-time pose inference.

To our knowledge, there is no published prior work on using ConvNets to recover 3D pose of non-rigid objects from depth or RGB data. However, several groups have shown ConvNets can recover the pose of rigid 3D objects such as plastic toys and faces. For example, LeCun et al. [46] used ConvNets to deduce the 6DOF pose of 3D plastic toys by finding a low-dimensional embedding which maps RGB images to a six dimensional space.

5.3 Binary Classification

For the task of hand-background depth image segmentation we trained an RDF classifier to perform per-pixel binary segmentation on a single image. The output of this stage is shown in Figure 5.2. Decision forests are well-suited for discrete classification of body parts [72]. Furthermore, since decision forest classification is trivially parallelizable, it is well-suited to real-time processing in multi-core environments.

After Shotton et al., our RDF is designed to classify each pixel in a depth image as belonging to a hand or background. Each tree in the RDF consists of a set of sequential deterministic decisions, called weak-learners (or nodes), that compare the relative depth of the current pixel to a nearby pixel located at a learned offset. The particular sequence of decisions a pixel satisfies induces a tentative classification into hand or background. Averaging the classification from all trees in the forest induces a final probability distribution for each pixel. As our implementation differs only slightly from that of Shotton et al. in the form of the weak-learner we used, we refer interested readers to their past work, and focus on the innovations particular to our implementation.

While full body motion capture datasets are readily available [3], these datasets either lack articulation data for hands or else do not adequately cover the wide variety of poses that were planned for this system. Therefore, it was necessary to create a custom database of full body depth images with binary hand labeling for RDF training (Figure 5.2). We had one user paint their hands bright red with body paint and used a simple HSV-based distance metric to estimate a coarse hand labeling on the RGB image. The coarse labeling is then filtered using a median filter to remove outliers. Since current commodity RGB+Depth (RGBD) cameras, typically exhibit imperfect alignment between depth and RGB, especially near depth discontinuities that occur frequently on the hand, we used a combination of graph cut and depth-sensitive flood fill to further clean up the depth image labeling [12].

We used a weak-learner decision function similar to Shotton et al. but with some modifications for improved performance on our dataset. At a given pixel (u, v) on the

(a) Target Labels

(b) Learned Labels

Figure 5.2: Decision forest learned labels closely match target.

depth image I each node in the decision tree evaluates:

$$I\left(u + \frac{\Delta u}{I(u, v)}, v + \frac{\Delta v}{I(u, v)}\right) - I(u, v) \geq d_t \tag{5.1}$$

where $I(u, v)$ is the depth pixel value in image I, Δu and Δv are learned pixel offsets, and d_t is a learned depth threshold. Experimentally, we found that the objective function (5.1) performs better on our dataset than the objective function proposed by Shotton et al. [72]. However, we found that equation (5.1) requires a large dynamic range of pixel offsets during training to achieve good classification performance. We suspect that this is because a given decision path needs to use both global and local geometry information to perform efficient hand-background segmentation. Since training time is limited, we define a discrete set of weak-learners that use offset and threshold values that are linear in log space and then we randomly sample weak-learners from this space during training.

5.4 Dataset Creation

The goal of this stage is to create a database of RGBD sensor images representing a broad range of hand gestures with accurate ground-truth estimates (i.e. labels) of joint parameters in each image that may be used to train a ConvNet. The desired ground truth label consists of a 42-dimensional vector describing the full degree of freedom pose for the hand in that frame. The DOF of each hand-joint is shown in Figure 5.3. After the hand has been segmented from the background using the RDF-based binary classification just described, we use a direct search method to deduce the pose parameters based on the approach of Oikonomidis et al. [62]. An important insight of our work is that we can capture the power of their direct search method in an offline fashion, and then use it to train ConvNets (or similar algorithms) that are better suited to fast computation. One advantage of this decoupling is that during offline training we are not penalized for using more complicated models, which are more expensive to render, and which better explain the incoming range data. A second advantage is that we can utilize multiple sensors for training, thereby mitigating problems of self-occlusion during real-time interaction with a single sensor.

The algorithm proposed by Oikonomidis et al. [62] and adopted with modifications for this work is as follows: Starting with an approximate hand pose, a synthetic depth image is rendered and compared to the depth image using an scalar objective function. In practice the hand pose is estimated using the previous frame's pose when fitting a sequence of recorded frames. The pose is manually estimated using a simple UI for the first frame in a sequence. This results in a single scalar value representing the quality of the fit given the estimated pose coefficients. The particle swarm optimization with partial randomization (PrPSO) direct search method [84] is used to adjust the pose coefficient values to find the best fit pose that minimizes this objective function value.

Since PSO convergence is slow once the swarm positions are close to the final solution (which is exacerbated when partial randomization is used to prevent premature swarm collapse on early local minima), we then use a robust variant of the Nelder-Mead optimization algorithm [76] after PSO has completed.

Since this dataset creation stage is performed offline, we do not require it to be fast enough for interactive frame rates. Therefore we used a high-quality, linear-blend-skinning (LBS) model [87] (shown in Figure 5.3) as an alternative to the simple ball-and-cylinder model of Oikonomidis et al. After reducing the LBS model's face count to increase render throughput, the model contains 1722 vertices and 3381 triangle faces, whereas the high density source model contained 67606 faces. While LBS fails to ac-

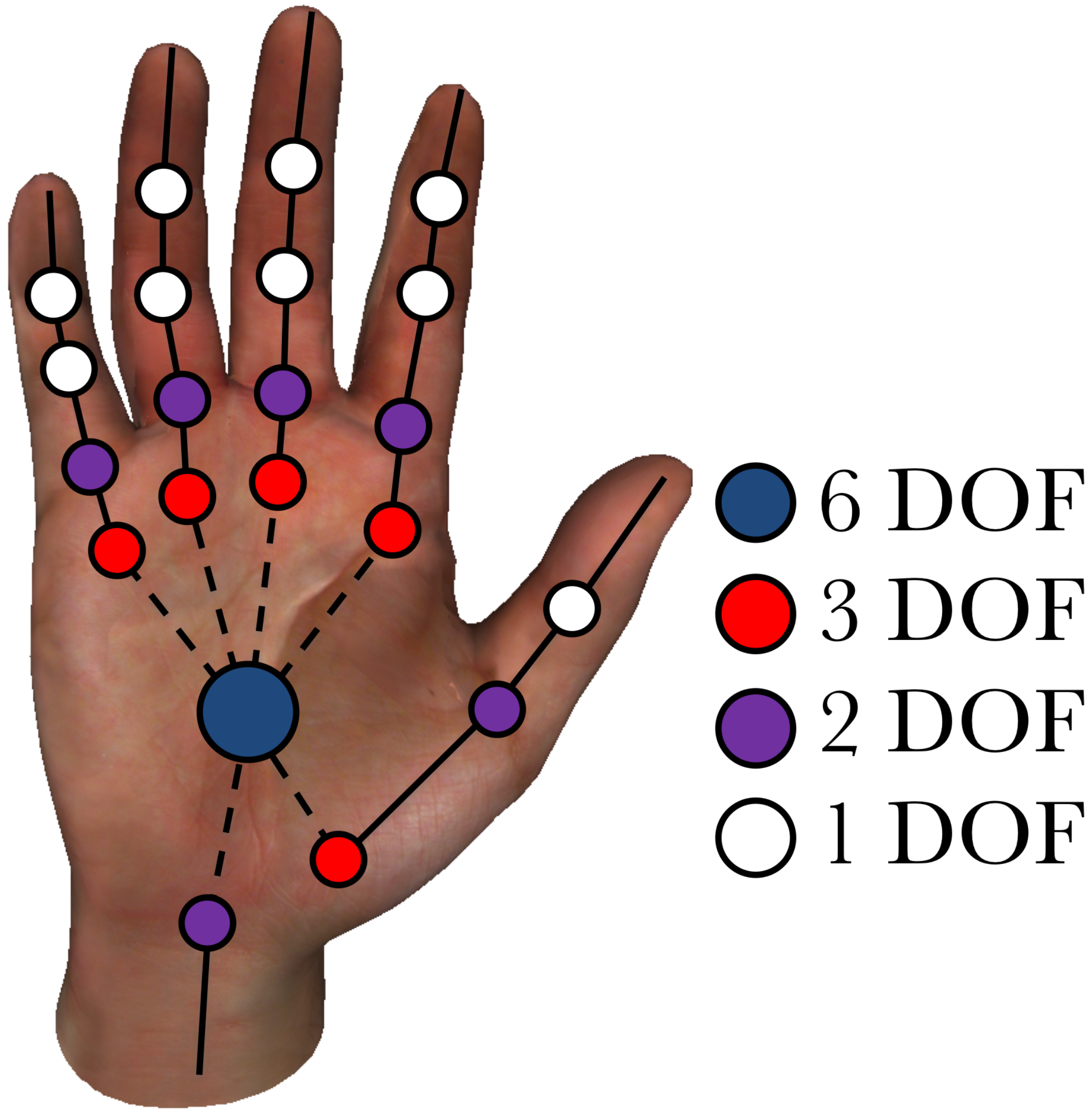


Figure 5.3: Linear blend skinning (LBS) model with 42 degrees of freedom.

curately model effects such as muscle deformation and skin folding, it represents many geometric details that ball-and-stick models cannot.

To mitigate the effects of self-occlusion we used three sensors (at viewpoints separated by approximately 45 degrees surrounding the user from the front) with attached vibration motors to reduce IR-pattern interference [14]. The contributions from each camera were accumulated into an overall fitness function $F(C)$, which includes two *a priori* terms ($\Phi(C)$ and $I(C)$) to maintain anatomically correct joint angles as well as a data-dependant term $\Delta(I_s, C)$ from each camera's contribution. The fitness function is as follows:

$$F(C) = \sum_{s=1}^{3} \Big(\Delta(I_s, C) \Big) + \Phi(C) + I(C) \tag{5.2}$$

Where I_s is the s sensor's depth image and C is a 42-dimensional coefficient vector that represents the 6 DOF position and orientation of the hand as well as 36 internal joint angles (shown in Figure 5.3). $I(C)$ is an interpenetration term (for a given pose) used to invalidate anatomically incorrect hand poses and is calculated by accumulating the interpenetration distances of a series of bounding spheres attached to the bones of the 3D model. $\Phi(C)$ enforces a soft constraint that coefficient values stay within a predetermined range ($C_{\min}$ and $C_{\max}$):

$$\Phi(C) = \sum_{k=1}^{n} w_k \left[\max(C_k - C_{k\max}, 0) + \max(C_{k\min} - C_k, 0) \right]$$

Where, w_k is a per-coefficient weighting term to normalize penalty contributions accross different units (since we are including error terms for angle and distance in the same objective function). Lastly $\Delta(I_s, C)$ of Equation (5.2), measures the similarity between the depth image I_s and the synthetic pose rendered from the same viewpoint:

$$\Delta(I_s, C) = \sum_{u,v} \min\left(|I_s(u,v) - R_s(C,u,v)|, d_{\max}\right)$$

Where, $I_s(u,v)$ is the depth at pixel (u,v) of sensor s, $R_s(C,u,v)$ is the synthetic depth given the pose coefficient C and $d_{\max}$ is a maximum depth constant. The result of this function is a clamped L1-norm pixel-wise comparison. It should be noted that we do not include energy terms that measure the silhouette similarity as proposed by Oikonomidis et al. since we found that when multiple range sensors are used these terms are not necessary.

Since the same pose is seen by multiple cameras from different viewpoints, it is necessary to calculate an affine transformation from each camera's viewpoint into a consistent basis. Due to variations in the imaging systems of each camera, small three-dimensional scale variations are common across views of the same object. For this reason, classical iterative closest point (ICP) with scale [35] is insufficient for our needs, and instead we use a straightforward weighted variant of ICP based on energy minimization. In this variant, for an arbitrary 3D point in the data p_j, and the corresponding closest point q_j on the model, we minimize the overall discrepancy of the fit E, given the current camera to world coordinate transformation T_i (for each camera i), as follows:

$$E = \sum_{i=1}^{N} \sum_{j=1}^{M} w_j \left\| T_i q_j - p_j \right\|^2$$

$$w_j = \frac{\max\left(0, \frac{\mathrm{dot}(\hat{q}_j, \hat{p}_j) - k}{1-k}\right)}{1 + \left\| p_j - q_j \right\|}$$

Where there are $N + 1$ cameras and M points in our system. The per-correspondence weight w_j biases the fit towards points that are already close and have normals $\hat{p}_j, \hat{q}_j$

pointing in the same direction. The resulting fit is mostly insensitive to the constant $k \in (0, 1)$, and we used $k = 0.5$. Using suitable calibration geometry and a known approximate initial transformation, the above algorithm converges within 50 to 100 iterations using BFGS to perturb the parameters of the affine matrices T_i.

5.5 Feature Detection

While Neural Networks have been used for pose detection of a limited set of discrete hand gestures (for instance discriminating between a closed fist and an open palm) [56, 60], to our knowledge this is the first work that has attempted to use such networks to perform dense feature extraction of an articulable object in order to infer continuous pose. To do this we employ a multi-resolution, deep ConvNet architecture inspired by the work of Farabet et al. [27] in order to perform feature extraction of 14 salient hand points from a segmented hand image. ConvNets are biologically inspired variants of multi-layered perceptrons, which exploit spatial correlation in natural images by extracting features generated by localized convolution kernels. Since depth images of hands tend to have many repeated local image features (e.g., fingertips), ConvNets are well suited to perform feature extraction since multi-layered feature banks can share common features, thereby reducing the number of required free parameters.

We recast the full hand-pose recognition problem as an intermediate collection of easier individual hand-feature recognition problems, which can be more easily learned by ConvNets. In early experiments, we found inferring mappings between depth image space and pose space directly (for instance measuring depth image geometry to extract a joint angle), yielded inferior results to learning with intermediate features. We hypothesize that one reason for this could be that learning intermediate features allows

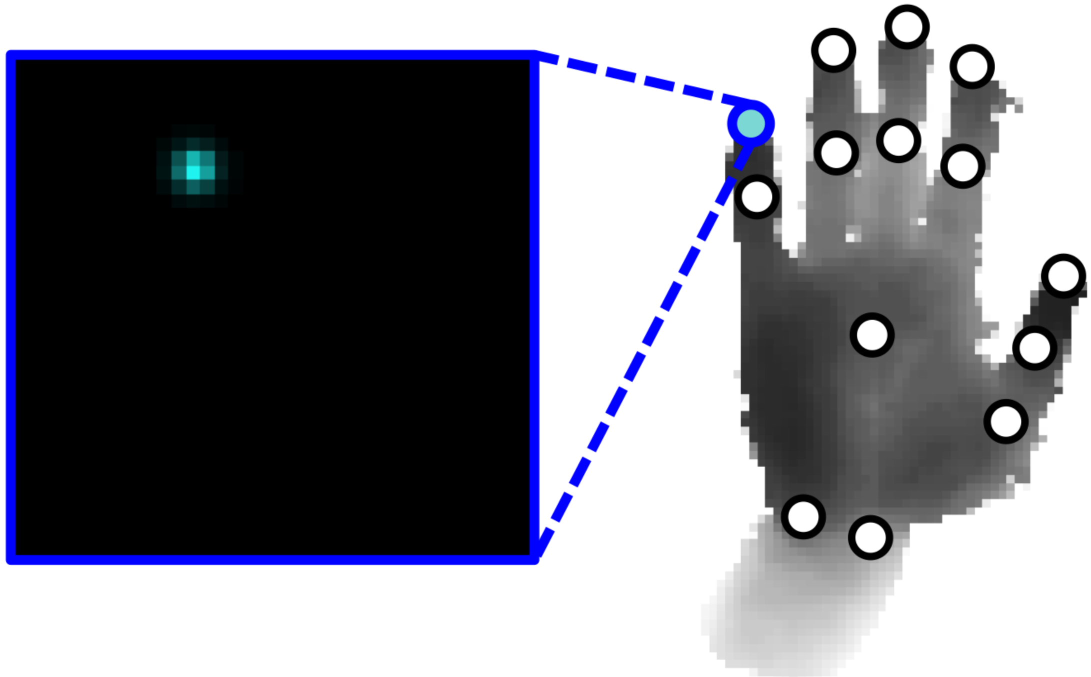


Figure 5.4: Normalized depth image of an open hand from the training set, overlaid with 14 feature locations and the heat-map for one fingertip feature.

ConvNets to concentrate the capacity of the network on learning local features, and on differentiating between them.

We trained the ConvNet architecture to generate an output set of “heat-map” feature images (Figure 5.4). Each feature heat-map can be viewed as a 2D Gaussian (truncated to have finite support), where the pixel intensity represents the probability of that feature occurring in that spatial location. The Gaussian UV mean is centered at one of 14 feature points of the user’s hand. These features represent key joint locations in the 3D model (e.g., knuckles) and were chosen such that the inverse kinematics (IK) algorithm described in Section 5.6 can recover a full 3D pose.

We found that the intermediate heat-map representation not only reduces required learning capacity but also improves generalization performance since failure modes are often recoverable. Cases contributing to high test set error (where the input pose is

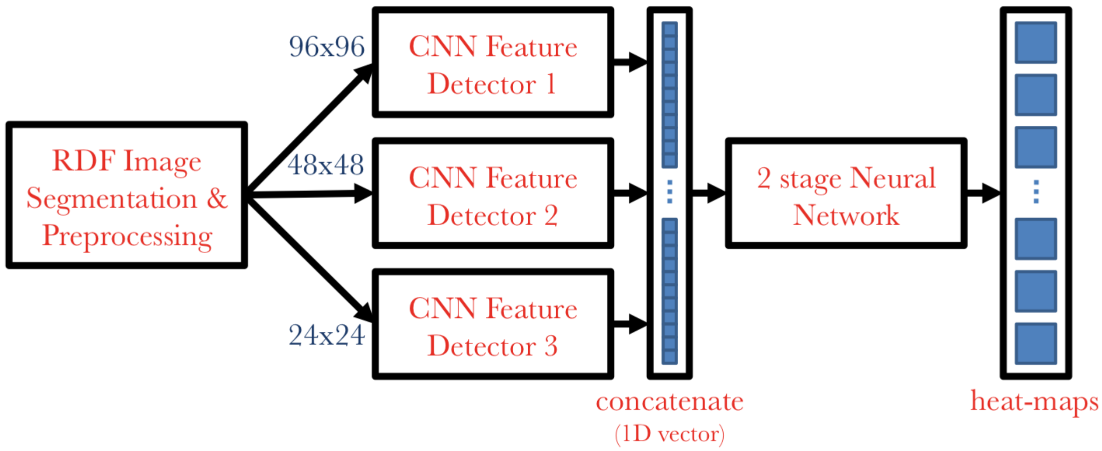


Figure 5.5: The convolutional neural network architecture used for hand pose recovery.

vastly different from anything in the training set) are usually heat-maps that contain multiple hotspots. For instance, the heat-map for a fingertip feature might incorrectly contain multiple lobes corresponding to the other finger locations as the network failed to discriminate among fingers. When this situation occurs it is possible to recover a reasonable feature location by simple heuristics to decide which of these lobes corresponds to the desired feature (for instance if another heat-map shows higher probability in those same lobe regions then we can eliminate these as spurious outliers). Similarly, the intensity of the heat-map lobe gives a direct indication of the system's confidence for that feature, which is an extremely useful measure for practical applications.

Our multi-resolution ConvNet architecture is shown in Figure 5.5. The segmented depth image is initially pre-processed, whereby the image is cropped and scaled by a factor proportional to the mean depth value of the hand pixels, so that the hand is in the center and has size that is depth invariant. The depth values of each pixel are then normalized between 0 and 1 (with background pixels set to 1). The cropped and normalized image is shown in Figure 5.4.

The preprocessed image is then filtered using local contrast normalization [38],

(a) 96×96px (b) 48×48px (c) 24×24px

Figure 5.6: Multi-resolution pyramid input

which acts as a high-pass filter to emphasize geometric discontinuities. The image is then downsampled twice (each time by a factor of 2) and the same filter is applied to each image. This produces a multi-resolution band-pass image pyramid with 3 banks (shown in Figure 5.6), whose total spectral density approximates the spectral density of the input depth image. Since experimentally we have found that hand-pose extraction requires knowledge of both local and global features, a single resolution ConvNet would need to examine a large image window and thus would require a large learning capacity.

The pyramid images are propagated through a 2-stage ConvNet architecture. The highest resolution feature bank is shown in Figure 5.7. Each bank is comprised of 2 convolution modules, 2 piecewise non-linearity modules, and 2 max-pooling modules. The convolution window sizes range from 4x4 to 6x6 pixels. We use max-pooling since it effectively reduces computational complexity at the cost of spatial precision [57]. The max-pooling windows range from 2x2 to 4x4 pixels. The nonlinearity is a Rectify Linear Unit (ReLU), which has been shown to improve training speed and discrimination performance in comparison to the standard sigmoid units [43].

Lastly, the output of the ConvNet banks are fed into a 2-stage neural network shown in Figure 5.8. This network creates the final 14 heat-map images. In practice these

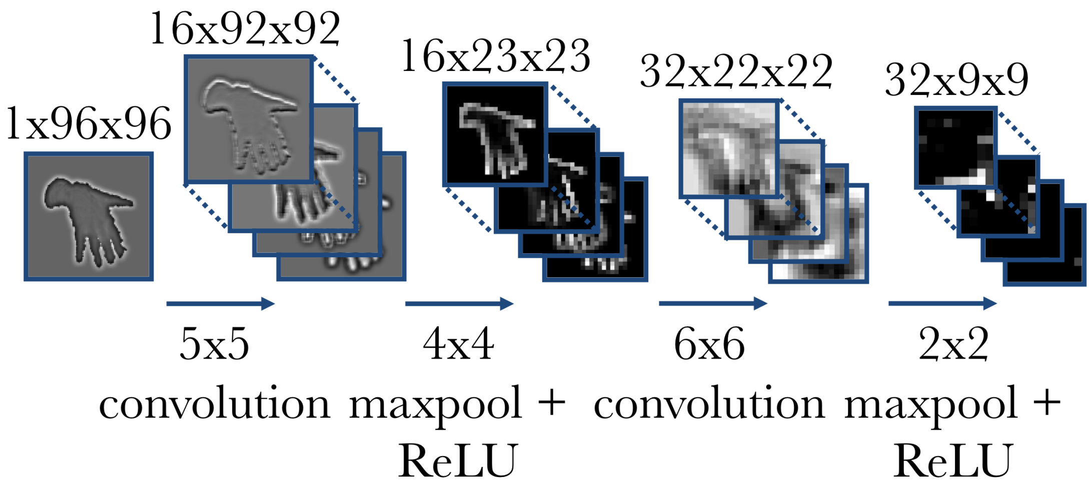


Figure 5.7: The high-resolution bank feature detector (96×96px): each stage: ($N_{\text{features}} \times \text{height} \times \text{width}$)

two large and fully-connected linear networks account for more than 80% of the total computational cost of the ConvNet. However, reducing the size of the network has a very strong impact on runtime performance. For this reason, it is important to find a good tradeoff between quality and speed.

ConvNet training was performed using the open-source machine learning package Torch7 [17], which provides access to efficient GPGPU back-propagation primitives. During supervised training we use stochastic gradient descent with a standard L2-norm error function, batch size of 64 and the following learnable parameter update rule:

$$\begin{aligned} \gamma_i &= \max\left(\gamma_{min}, \gamma_0 - k\left(i-1\right)\right) \\ \Delta_i &= \gamma_i \Delta_{i-1} - \lambda\left(\eta w_i - \frac{\partial L}{\partial w_i}\right) \\ w_{i+1} &= w_i + \Delta_i \end{aligned} \tag{5.3}$$

where w_i is a bias or weight parameter for each of the network modules for epoch i (with each epoch representing one pass over the entire training-set) and $\frac{\partial L}{\partial w_i}$ is the partial

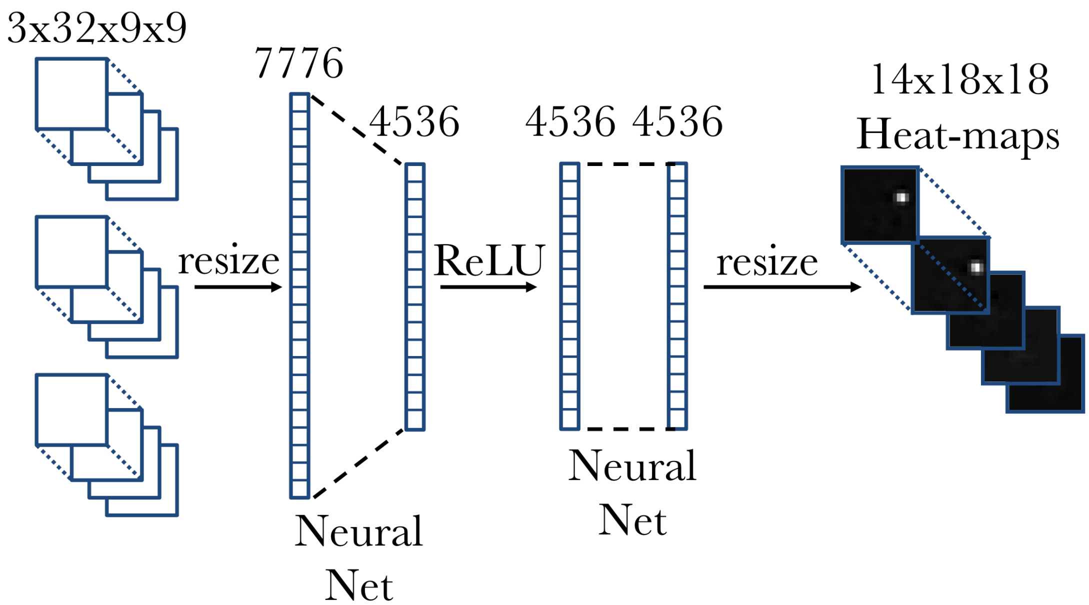


Figure 5.8: 2-Stage neural network to create the 14 heat maps

derivative of the error function L with respect to the learnable parameter w_i averaged over the current batch. We use a constant learning rate of $\lambda = 0.2$. Additionally, we have found that using a high initial training momentum of $\gamma_0 = 0.9$ and then linearly decaying it by a factor of $k = 0.005$ per epoch until the momentum is $\gamma_{min} = 0.4$ helps improve learning rate. Lastly, an L2 regularization factor of $\eta = 0.0005$ is used to help improve generalization.

During ConvNet training the database images were randomly rotated, scaled and translated to improve generalization performance [27]. Not only does this technique effectively increase the size of the training set (which improves test set error), it also helps improve performance for other users whose hand size is not well represented in the original training set. We perform this image manipulation in a background thread during batch-training so the impact on training time is minimal.

5.6 Pose Recovery

We formulate the problem of pose estimation from the heat-map output as an optimization problem, similar to inverse kinematics (IK). We extract 2D and 3D feature positions from the 14 heat-maps and then minimize an appropriate objective function to align 3D model features to each heat-map position.

To infer the 3D position corresponding to a heat-map image, we need to determine the most likely UV position of the feature in the heat-map. Although the ConvNet architecture is trained to output heat-map images of 2D Gaussians with low-variance, in general, they output multimodal grayscale heat-maps which usually do not sum to 1. In practice, it is easy to deduce a correct UV position using simple heuristics. First we clamp heat-map pixels below a fixed threshold to get rid of improbable points.

We then normalize the resulting image so it sums to 1, and measure its mean and standard deviation (STD). In case the STD is above a threshold, we fit the best 2D Gaussian using Levenberg-Marquardt, and use the mean of the resulting Gaussian as the UV position. If the STD is below the threshold, we use the computed image mean directly as the UV position. Once the UV position is found for each of the 14 heat-maps, we perform a lookup into the captured depth frame to obtain the depth component at the UV location. In case this UV location lies on a depth shadow where no depth is given in the original image, we store the computed 2D image for this point in the original image space. Otherwise, we store its 3D point.

We then perform unconstrained nonlinear optimization on the following objective function:

$$f(m) = \sum_{i=1}^{n} [\Delta_i(m)] + \Phi(C) \;\; \Delta_i(m) = \begin{cases} \left\| (u,v,d)_i^t - (u,v,d)_i^m \right\|_2 & \text{If } d_i^t \neq 0 \\ \left\| (u,v)_i^t - (u,v)_i^m \right\|_2 & \text{otherwise} \end{cases} \tag{5.4}$$

Where $(u,v,d)_i^t$ is the target 3D heat-map position of feature i and $(u,v,d)_i^m$ is the model feature position for the current pose estimate. Equation (5.4) is an L2 error norm in 3D or 2D depending on whether or not the given feature has a valid depth component associated with it. We then use a simple linear accumulation of these feature-wise error terms as well as the same linear penalty constraint ($\Phi(C)$) used in Section 5.4. We use PrPSO to minimize Equation (5.4). Since function evaluations for each swarm particle can be parallelized, PrPSO is able to run in real-time at interactive frame rates for this stage. Furthermore, since a number of the 42 coefficients from Section 5.4 contribute only subtle behavior to the deformation of the LBS model, we found that removing coefficients describing finger twist and coupling the last two knuckles of each finger into a single angle coefficient significantly reduces function evaluation time of (5.4) without noticeable loss in pose accuracy. Therefore, we reduce the complexity of the model to 23 DOF during this final stage to reduce the computation time of function evaluations.

This IK approach has one important limitation; the UVD target position may not be a good representation of the true feature position. For instance, when a feature is directly occluded by another feature, the two features will incorrectly share the same depth value (even though one is in front of the other).

5.7 Results

The RDF classifier described in Section 5.4 was trained using 6,500 images and validated with a test set of 1,000 images of a user performing typical one and two handed gestures (pinching, drawing, clapping, grasping, etc). Training was performed on a 24 core machine for approximately 12 hours. For each node in the tree, 10,000 weak-learners were sampled. The error ratio of the number of incorrect pixel labels to total number of hand pixels in the dataset for varying tree counts and tree heights is shown in Figure 5.9.

We found that 4 trees with a height of 25 was a good tradeoff of classification accuracy versus speed. The test set classification error for 4 trees of depth 25 was 4.1%. Of the classification errors, 76.3% were false positives and 23.7% were false negatives. We found that in practice small clusters of false positive pixel labels can be easily removed using median filtering and blob detection. The common classification failure cases occur when the hand is occluded by another body-part (causing false positives), or when the elbow is much closer to the camera than the hand (causing false positives on the elbow). We believe the cause for this inaccuracy is due to the fact that the training set did not contain any frames containing these poses. A more comprehensive dataset, containing examples of these poses, should improve performance in future.

Since we do not have a ground-truth measure for the 42 DOF hand model fitting, quantitative evaluation of this stage is difficult. *Qualitatively*, the fitting accuracy was visually consistent with the underlying point cloud. An example of a fitted frame is shown in Figure 5.10. Only a very small number of poses failed to fit correctly; for these difficult poses, manual intervention was required.

One limitation of this system was that the frame rate of the PrimeSense™camera

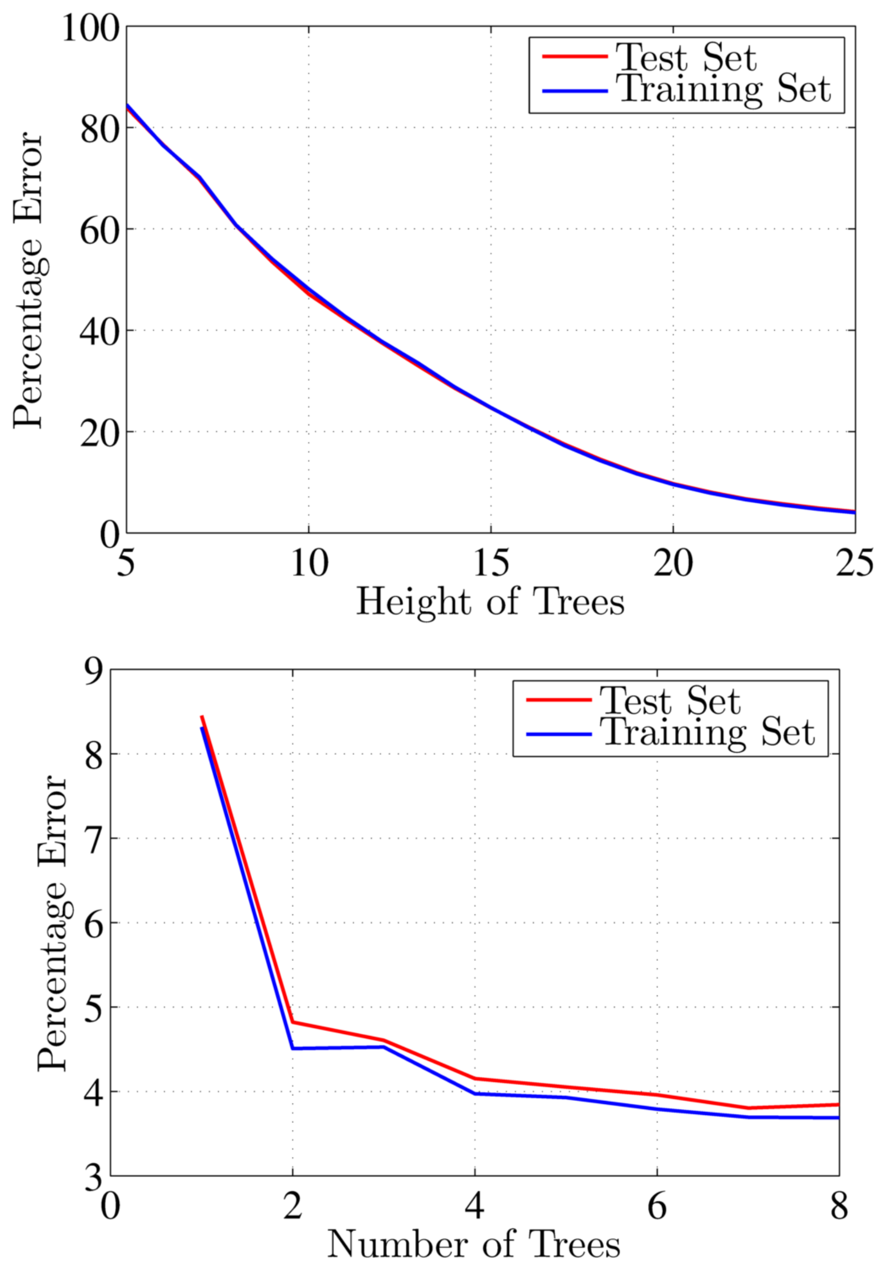


Figure 5.9: Percentage of incorrectly labeled pixels for randomized decision forest, varying tree height (a) and number of trees (b).

(a) Sensor Depth (b) Synthetic Depth (c) Per-Pixel Error

Figure 5.10: Dataset generation vs. ground truth

(30fps) was not enough to ensure sufficient temporal coherence for correct convergence of the PSO optimizer. To overcome this, we had each user move their hands slowly during training data capture. Using a workstation with an Nvidia GTX 580 GPU and 4 core Intel processor, fitting each frame required 3 to 6 seconds. The final database consisted of 76,712 training set images and 2,421 test set images with their corresponding heat-maps, collected from multiple participants. A small sample of the test set images is shown in Figure 5.11.

The ConvNet training took approximately 24 hours and the test-set error failed to improve after approximately 350 training epochs. Learning rate, momentum, L2 regularization, and ConvNet architecture parameters (e.g., max-pooling window size) were chosen by coarse meta-optimization to minimize the final test-set error. Figure 5.12 shows the mean squared error (MSE) after each epoch. The MSE was calculated by taking the mean of sum-of-squared differences between the calculated 14 feature maps and the corresponding target feature maps.

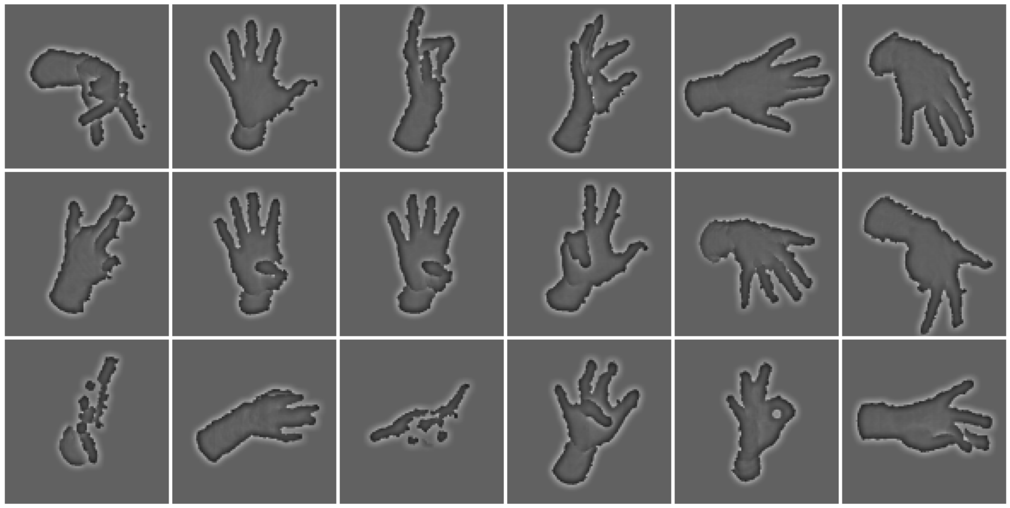


Figure 5.11: Example test set images for ConvNet

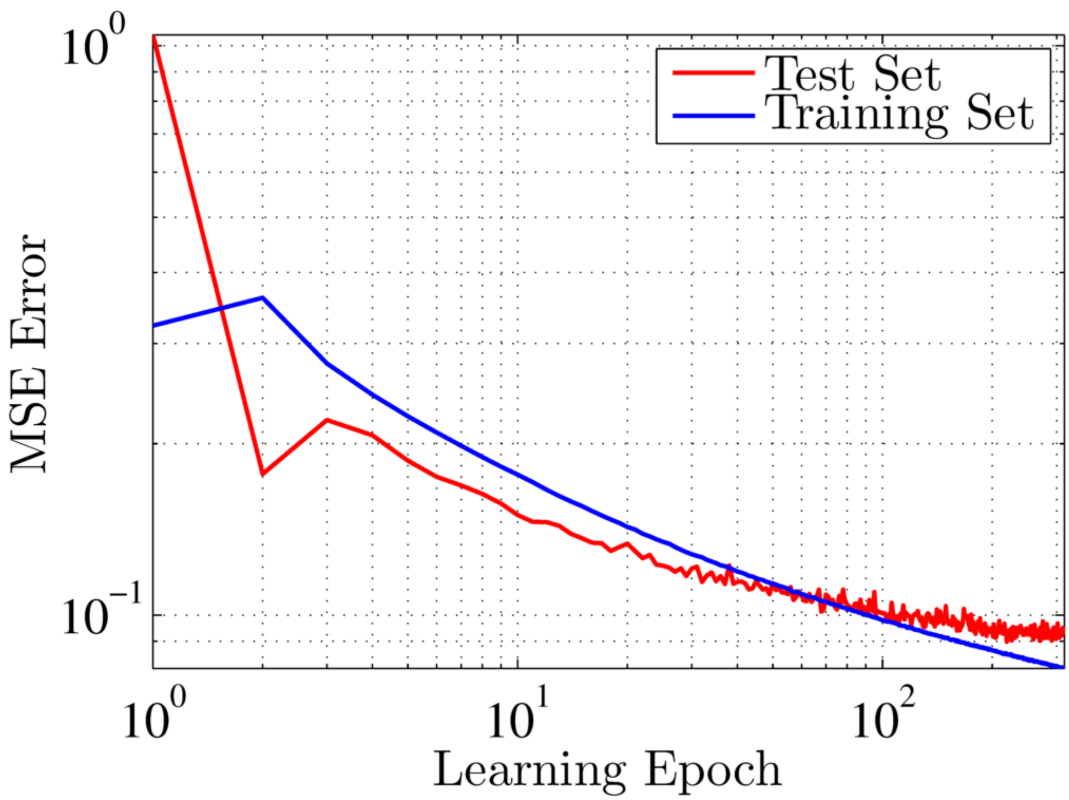


Figure 5.12: ConvNet Learning Curve

Feature Type	Mean (px)	STD (px)
Palm	0.33	0.30
Thumb Base & Knuckle	0.33	0.43
Thumb Tip	0.39	0.55
Finger Knuckle	0.38	0.27
Finger Tip	0.54	0.33

Table 5.1: Heat-Map of UV Error by Feature Type

The mean UV error of the ConvNet heat-map output on the test set data was 0.41px (with standard deviation of 0.35px) on the 18x18 resolution heat-map image[1]. After each heat-map feature was translated to the 640x480 depth image, the mean UV error was 5.8px (with standard deviation of 4.9px). Since the heat-map downsampling ratio is depth dependent, the UV error improves as the hand approaches the sensor. For applications that require greater accuracy, the heat-map resolution can be increased for better spatial accuracy at the cost of increased latency and reduced throughput.

Table 5.1 shows the UV accuracy for each feature type. Unsurprisingly, we found that the ConvNet architecture had the most difficulty learning fingertip positions, where the mean error is 61% higher than the accuracy of the palm features. The likely cause for this inaccuracy is twofold. Firstly, the fingertip positions undergo a large range of motion between various hand-poses and therefore the ConvNet must learn a more difficult mapping between local features and fingertip positions. Secondly, the PrimeSense™Carmine 1.09 depth camera cannot always recover depth of small surfaces such as fingertips. The ConvNet is able to learn this noise behavior, and is actually able to approximate fingertip location in the presence of missing data. However the accuracy for these poses is low.

The computation time of the entire pipeline is 24.9ms, which is within our 30fps

[1]To calculate this error we used the technique described in Section 5.6 to calculate the heat-map UV feature location and then calculated the error distance between the target and ConvNet output locations

performance target. Within this period: decision forest evaluation takes 3.4ms, depth image preprocessing takes 4.7ms, ConvNet evaluation takes 5.6ms and pose estimation takes 11.2ms. The entire pipeline introduces approximately one frame of latency.

5.8 Future Work

Qualitatively, we have found that the ConvNet generalization performance to hand shapes not represented in the training set is acceptable but could be improved. We are confident we can make improvements by adding more training data from users with different hand sizes to the training set.

For this work, only the ConvNet forward propagation stage was implemented on the GPU. We are currently working on implementing the entire pipeline on the GPU, which should improve performance of the other pipeline stages significantly. For example, the GPU ConvNet implementation requires 5.6ms, while the same network executed on the CPU (using optimized multi-threaded C++ code) requires 139ms.

The current implementation of our system can track two hands only if they are not interacting. We see this as a significant limitation. While we have determined that the dataset generation system can fit multiple strongly interacting hand poses with sufficient accuracy, we have not evaluated the neural network recognition performance on these poses. Likewise, we hope to evaluate the recognition performance on hand poses involving interactions with non-hand objects (such as pens and other man-made devices).

Lastly, while the pose recovery implementation presented in this work is fast, we hope to augment this stage by including a model-based fitting step that trades convergence radius for fit quality. Specifically, we suspect that replacing our final IK stage with an energy-based local optimization method, inspired by the work of Li et al. [47]

could allow our method to recover second-order surface effects such as skin folding and skin-muscle coupling from very limited data, and still with low-latency. In addition to inference, such a localized energy-minimizing stage would enable improvements to the underlying model itself. Since these localized methods typically require good registration, our method, which gives correspondence from a single image could advance the state-of-the-art in non-rigid model capture.

5.9 Conclusion

In this chapter, we discussed our contribution of a novel pipeline for tracking the instantaneous pose of articulable objects from a single depth image. As an application of this pipeline we showed state-of-the-art results for tracking human hands in real-time using commodity hardware. This pipeline leverages the accuracy of offline model-based dataset generation routines in support of a robust real-time ConvNet architecture for feature extraction. We showed that it is possible to use intermediate heat-map features to extract accurate and reliable 3D pose information at interactive frame-rates using inverse kinematics.

Part IV

Back Matter

Appendix A

Games: Evaluation Materials

FactorReactor Arithmetic Fluency Pretest/Posttest

Name: ____________________ School: ____________________

10 - 5	1 + 6	3 × 7	7 + 9	2 ÷ 1	8 + 8	6 ÷ 3	4 × 7	6 ÷ 2	4 × 0
5 - 2	35 ÷ 7	10 - 2	4 × 4	4 - 0	14 ÷ 7	8 - 3	2 × 5	2 × 7	8 × 1
8 - 6	9 - 2	7 - 3	6 × 3	5 × 3	4 × 9	7 - 2	6 + 7	2 × 8	49 ÷ 7
1 × 2	0 ÷ 5	9 × 6	8 ÷ 1	4 + 5	3 × 9	18 ÷ 6	2 + 5	2 × 2	8 × 3
7 × 6	10 - 6	15 ÷ 5	2 × 6	9 - 1	3 + 4	3 × 4	9 + 8	32 ÷ 8	4 ÷ 2
5 - 5	39 ÷ 3	5 ÷ 1	6 - 5	8 - 7	12 ÷ 3	25 ÷ 5	9 × 9	9 - 7	8 - 4
5 + 5	10 - 9	2 + 9	8 × 5	8 + 4	64 ÷ 4	54 ÷ 6	4 + 6	72 ÷ 8	3 + 8
9 - 6	4 + 4	48 ÷ 8	3 - 3	7 + 5	6 - 4	7 + 2	9 - 8	36 ÷ 9	3 + 3

Pre

Go to the next page →

7 + 4	16 ÷ 4	10 - 3	6 × 5	10 - 8	9 - 3	3 + 5	9 + 3	9 + 9	6 + 5
3 × 3	2 + 4	7 - 5	2 + 8	4 × 6	16 ÷ 8	24 ÷ 6	54 ÷ 9	6 + 6	7 × 1
42 ÷ 6	8 + 4	8 × 6	8 - 5	10 ÷ 2	9 - 4	6 × 6	9 + 5	4 × 2	7 - 1
21 ÷ 3	4 + 0	10 - 7	7 - 4	0 ÷ 3	10 - 4	9 - 5	6 - 1	10 + 5	1 + 9
8 + 5	5 × 9	7 - 7	6 + 8	10 ÷ 5	7 + 7	4 × 8	8 ÷ 2	1 × 1	8 + 1
10 - 0	0 × 6	32 ÷ 2	7 ÷ 7	8 ÷ 4	7 × 7	7 - 0	30 ÷ 5	9 × 8	7 × 9
4 - 1	5 × 1	8 + 7	7 × 8	3 - 0	3 + 2	12 ÷ 4	22 ÷ 2	3 + 6	8 - 8
36 ÷ 6	9 + 4	1 × 0	10 + 6	3 + 7	9 ÷ 3	2 × 3	81 ÷ 9	9 + 6	5 × 7

Pre

Stop

FactorReactor Achievement Survey

PALS-G

Name:

School:

Game played:

Date:

INSTRUCTIONS: These questions are about the game you just played. Using the scale below, please choose how much you agree or disagree with each sentence. Please **CIRCLE** your answer.

1. It was important to me that I learned a lot from playing the game.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

2. It was important to me that other students in my class thought I was good at the game.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

3. It was important to me that my performance on the game didn't make me look stupid.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

4. One of my goals was to learn as much as I could about the game.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

PALS-G

5. One of my goals was to show others that I was good at the game.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

6. One of my goals was to keep others from thinking I wasn't good at the game.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

7. One of my goals was to master a lot of new skills while playing the game.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

8. One of my goals was to show others that the game was easy for me.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

9. It was important to me that no one thought I was worse at the game than others in class.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

PALS-G

10. It was important to me that I thoroughly understood how to play the game.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

11. When I was playing the game, one of my goals was to look like I was doing well in comparison to the other students.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

12. One of my goals was to avoid looking like I had trouble playing the game.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

13. It was important to me that I improved my skills on the game.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

14. It was important to me that I looked good at the game compared to others.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

FactorReactor Gameplay Survey

Factor Reactor Game Survey

Your Name: Date:

Age: ______ years Grade: School:

1. Did you have fun playing this game? (Circle your response)

1	2	3	4	5
Not at all		Somewhat		A lot

2. How difficult was the game?

1	2	3	4	5
Too hard		Just right		Too easy

3. How well did you do in the game?

1	2	3	4	5
Not well at all		Somewhat		Very well

4. How much did you like this game?

1	2	3	4	5
I **hated** it		I didn't **love** or **hate** it		I **loved** it

5. What **did** you like about this game?

I liked____

____

____

____

6. What **didn't** you like about this game?

I didn't like____

____

____

____

7. How much did you learn from playing this game?

1	2	3	4	5
Nothing				A lot

8. What was this game trying to teach you?

____

____

____

9. Did you learn anything from this game? If yes, what did you learn?

____

____

____

10. Would you like to play this game again in the future?

1	2	3	4	5
Not at all		Maybe		Definitely

11. Would you recommend it to your friends, classmates or teachers? Why or why not?

1	2	3	4	5
Not at all		Maybe		Definitely

12. If you were the game developer, what would you do to make this game better? Why?

____

____

____

13. How much experience do you have using videogame controllers?

1	2	3	4	5
None				A lot

Noobs Arithmetic Fluency Pretest/Posttest

Name: ______________ School: ______________

10	1	3	7	2	8	6	4	6	4
-5	+ 6	- 7	+ 9	- 1	+ 8	- 3	+ 7	- 2	+ 0
5	35	10	4	4	14	8	2	2	8
- 2	- 7	- 2	+ 4	- 0	- 7	- 3	+ 5	+ 7	+ 1
8	9	7	6	15	4	7	6	20	49
- 6	- 2	- 3	+ 3	+ 3	+ 9	- 2	+ 7	+ 80	- 7
10	0	9	80	4	3	18	2	2	8
+ 2	- 5	+ 6	- 10	+ 5	+ 9	- 6	+ 5	+ 2	+ 3
70	100	50	20	90	30	40	90	32	40
+ 60	- 60	+ 15	+ 6	- 10	+ 40	- 30	+ 8	- 8	- 20
50	39	50	60	80	12	250	90	90	80
- 50	- 3	+ 10	- 50	- 70	- 3	- 50	- 90	- 70	- 40
50	100	200	80	80	64	54	40	72	300
+ 50	- 90	+ 90	+ 50	+ 40	- 4	- 6	+ 60	+ 80	+ 80
90	40	480	30	70	60	70	90	36	30
- 60	+ 40	- 80	- 30	+ 50	- 40	+ 20	- 80	- 9	+ 30

Pre

Go to the next page →

70	160	100	60	140	90	30	90	90	60
+ 40	- 40	- 30	+ 50	- 80	- 30	+ 5	+ 30	+ 90	+ 50
30	20	70	20	40	160	240	540	60	70
+ 30	+ 40	- 50	+ 80	+ 60	- 80	- 60	- 90	+ 60	- 10
420	80	80	80	110	190	60	90	40	70
- 60	+ 40	- 60	- 50	- 20	- 40	- 60	- 50	+ 20	- 10
210	40	100	70	0	170	90	60	100	120
- 30	+ 0	- 70	- 40	+ 30	- 40	- 50	- 10	+ 50	+ 90
80	50	70	60	100	70	400	80	190	80
+ 5	+ 90	- 70	+ 80	- 5	+ 70	- 80	- 20	+ 10	+ 10
110	0	320	175	800	750	70	300	90	70
- 0	+ 60	+ 20	- 80	- 40	- 70	- 0	- 50	+ 80	+ 90
400	560	80	70	30	310	125	220	300	180
- 10	+ 10	+ 70	+ 80	- 0	+ 20	- 40	- 20	+ 60	- 80
360	90	150	130	300	90	200	810	900	50
- 60	+ 40	+ 10	+ 60	+ 70	- 3	+300	- 90	+ 60	+ 70

Pre

Stop

FactorReactor Individual Interest in Math Survey

II-C

Name: Date:

School:

INSTRUCTIONS: Using the scale provided, please rate how much you agree or disagree with each statement. Please **CIRCLE** your answer.

1. I enjoy doing math.

1	2	3	4	5
Not at all true		Somewhat true		Very true

2. I like math.

1	2	3	4	5
Not at all true		Somewhat true		Very true

3. Math helps me in my daily life outside of school.

1	2	3	4	5
Not at all true		Somewhat true		Very true

4. I enjoy the topic of math.

1	2	3	4	5
Not at all true		Somewhat true		Very true

5. Math is practical for me to know.

1	2	3	4	5
Not at all true		Somewhat true		Very true

II-C

6. Thinking mathematically is an important part of who I am.

1	2	3	4	5
Not at all true		Somewhat true		Very true

7. Math is exciting to me.

1	2	3	4	5
Not at all true		Somewhat true		Very true

8. It is important to me to be a person who reasons mathematically.

1	2	3	4	5
Not at all true		Somewhat true		Very true

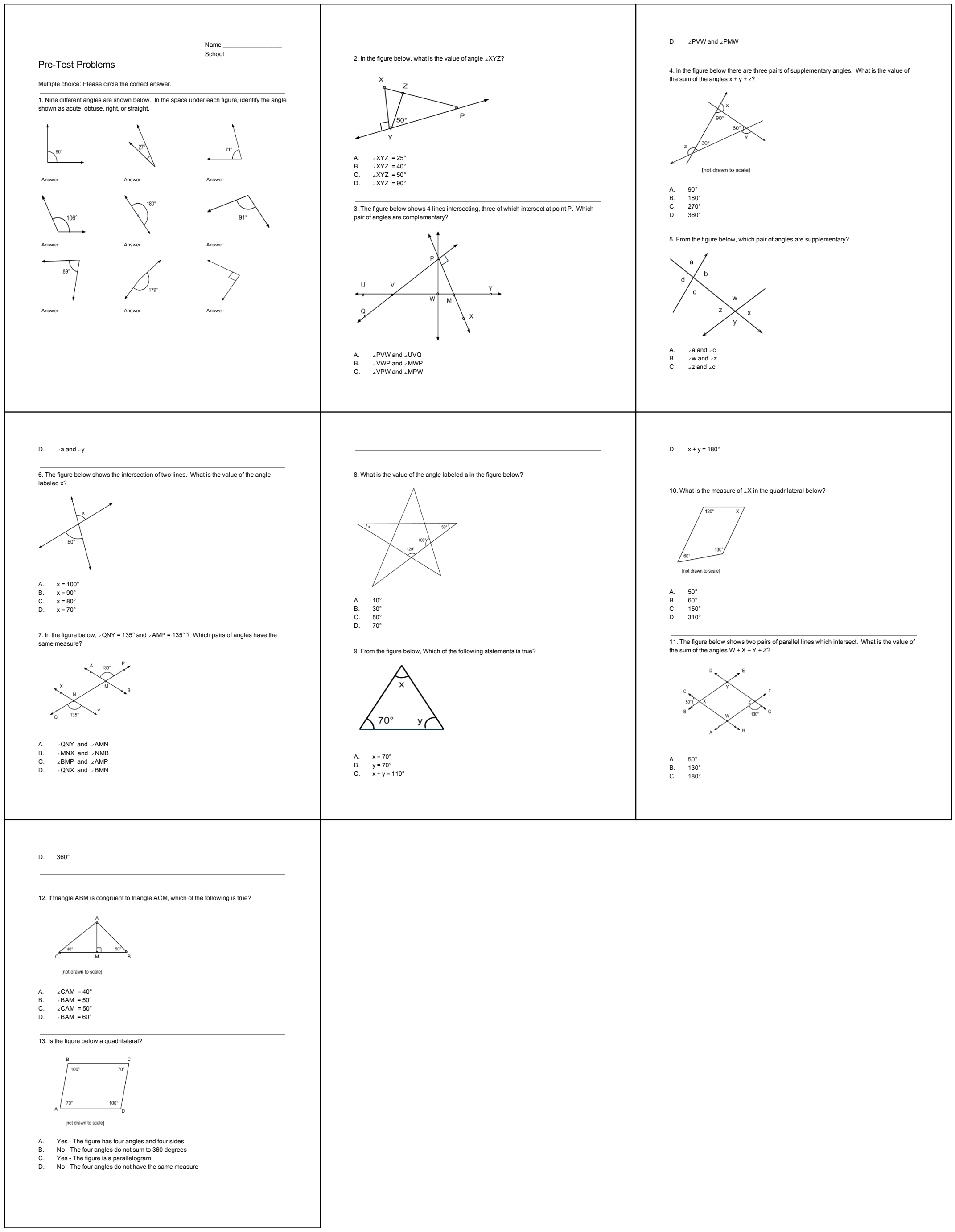
9. How much experience do you have with videogame controllers like the one you'll be using today?

1	2	3	4	5
None		Some		A lot

10. Have you used an Xbox controller before? (Circle one)

Yes No

Noobs Geometry Pretest



Name ______________
School ______________

Pre-Test Problems

Multiple choice: Please circle the correct answer.

1. Nine different angles are shown below. In the space under each figure, identify the angle shown as acute, obtuse, right, or straight.

Answer: Answer: Answer:

Answer: Answer: Answer:

Answer: Answer: Answer:

2. In the figure below, what is the value of angle ∠XYZ?

A. ∠XYZ = 25°
B. ∠XYZ = 40°
C. ∠XYZ = 50°
D. ∠XYZ = 90°

3. The figure below shows 4 lines intersecting, three of which intersect at point P. Which pair of angles are complementary?

A. ∠PVW and ∠UVQ
B. ∠VWP and ∠MWP
C. ∠VPW and ∠MPW
D. ∠PVW and ∠PMW

4. In the figure below there are three pairs of supplementary angles. What is the value of the sum of the angles x + y + z?

[not drawn to scale]

A. 90°
B. 180°
C. 270°
D. 360°

5. From the figure below, which pair of angles are supplementary?

A. ∠a and ∠c
B. ∠w and ∠z
C. ∠z and ∠c
D. ∠a and ∠y

6. The figure below shows the intersection of two lines. What is the value of the angle labeled x?

A. x = 100°
B. x = 90°
C. x = 80°
D. x = 70°

7. In the figure below, ∠QNY = 135° and ∠AMP = 135° ? Which pairs of angles have the same measure?

A. ∠QNY and ∠AMN
B. ∠MNX and ∠NMB
C. ∠BMP and ∠AMP
D. ∠QNX and ∠BMN

8. What is the value of the angle labeled **a** in the figure below?

A. 10°
B. 30°
C. 50°
D. 70°

9. From the figure below, Which of the following statements is true?

A. x = 70°
B. y = 70°
C. x + y = 110°
D. x + y = 180°

10. What is the measure of ∠X in the quadrilateral below?

[not drawn to scale]

A. 50°
B. 60°
C. 150°
D. 310°

11. The figure below shows two pairs of parallel lines which intersect. What is the value of the sum of the angles W + X + Y + Z?

A. 50°
B. 130°
C. 180°
D. 360°

12. If triangle ABM is congruent to triangle ACM, which of the following is true?

[not drawn to scale]

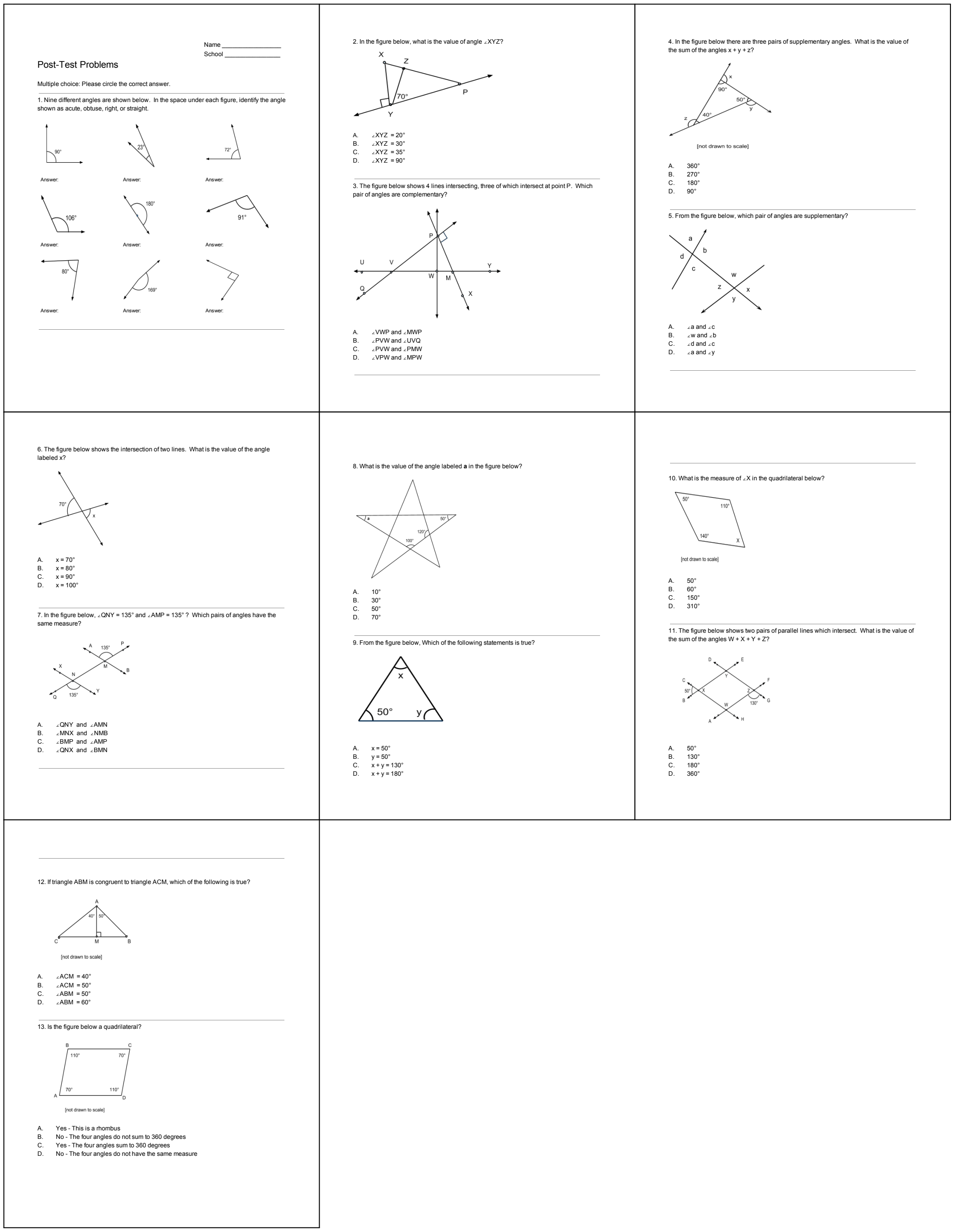
A. ∠CAM = 40°
B. ∠BAM = 50°
C. ∠CAM = 50°
D. ∠BAM = 60°

13. Is the figure below a quadrilateral?

[not drawn to scale]

A. Yes - The figure has four angles and four sides
B. No - The four angles do not sum to 360 degrees
C. Yes - The figure is a parallelogram
D. No - The four angles do not have the same measure

Noobs Geometry Posttest



Name ______________
School ______________

Post-Test Problems

Multiple choice: Please circle the correct answer.

1. Nine different angles are shown below. In the space under each figure, identify the angle shown as acute, obtuse, right, or straight.

Answer: Answer: Answer:

Answer: Answer: Answer:

Answer: Answer: Answer:

2. In the figure below, what is the value of angle ∠XYZ?

A. ∠XYZ = 20°
B. ∠XYZ = 30°
C. ∠XYZ = 35°
D. ∠XYZ = 90°

3. The figure below shows 4 lines intersecting, three of which intersect at point P. Which pair of angles are complementary?

A. ∠VWP and ∠MWP
B. ∠PVW and ∠UVQ
C. ∠PVW and ∠PMW
D. ∠VPW and ∠MPW

4. In the figure below there are three pairs of supplementary angles. What is the value of the sum of the angles x + y + z?

[not drawn to scale]

A. 360°
B. 270°
C. 180°
D. 90°

5. From the figure below, which pair of angles are supplementary?

A. ∠a and ∠c
B. ∠w and ∠b
C. ∠d and ∠c
D. ∠a and ∠y

6. The figure below shows the intersection of two lines. What is the value of the angle labeled x?

A. x = 70°
B. x = 80°
C. x = 90°
D. x = 100°

7. In the figure below, ∠QNY = 135° and ∠AMP = 135° ? Which pairs of angles have the same measure?

A. ∠QNY and ∠AMN
B. ∠MNX and ∠NMB
C. ∠BMP and ∠AMP
D. ∠QNX and ∠BMN

8. What is the value of the angle labeled **a** in the figure below?

A. 10°
B. 30°
C. 50°
D. 70°

9. From the figure below, Which of the following statements is true?

A. x = 50°
B. y = 50°
C. x + y = 130°
D. x + y = 180°

10. What is the measure of ∠X in the quadrilateral below?

[not drawn to scale]

A. 50°
B. 60°
C. 150°
D. 310°

11. The figure below shows two pairs of parallel lines which intersect. What is the value of the sum of the angles W + X + Y + Z?

A. 50°
B. 130°
C. 180°
D. 360°

12. If triangle ABM is congruent to triangle ACM, which of the following is true?

[not drawn to scale]

A. ∠ACM = 40°
B. ∠ACM = 50°
C. ∠ABM = 50°
D. ∠ABM = 60°

13. Is the figure below a quadrilateral?

[not drawn to scale]

A. Yes - This is a rhombus
B. No - The four angles do not sum to 360 degrees
C. Yes - The four angles sum to 360 degrees
D. No - The four angles do not have the same measure

Games For Learning Situational Interest Survey

SI-C

Name: ____________________

School: ____________________

Game played: ____________________

Date: ____________________

INSTRUCTIONS: These questions are about the game you just played. Using the scale below, please choose how much you agree or disagree with each sentence. Please **CIRCLE** your answer.

1. The game was exciting.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

2. The game was so exciting that it was easy to pay attention.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

3. I liked what I learned in the game.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

4. The things I learned from the game are important to me.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

5. The game grabbed my attention.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

SI-C

6. What I learned from the game is fascinating to me.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

7. I thought the game was interesting.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

8. What I learned from the game can be applied to real life.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

9. The game was entertaining.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

10. I am excited about what I learned from the game.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

11. What I learned in the game is useful for me to know.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

12. I learned valuable things from the game.

1	2	3	4	5	6	7
Very much disagree	Mostly disagree	A little bit disagree	Neither agree nor disagree	A little bit agree	Mostly agree	Very much agree

Noobs Individual Interest Survey in Games

IIG-C CIRCLE ONE: **PRE** or **POST** Study

Name:

School:

Date:

INSTRUCTIONS: Read each sentence and then circle the *one* number that shows how true it is of you. There are no right or wrong answers.

1. I enjoy playing games.

1	2	3	4	5
Not at all true		Somewhat true		Very true

2. Knowing how to play games is practical for me.

1	2	3	4	5
Not at all true		Somewhat true		Very true

3. It is important to me to be a person who plays games well.

1	2	3	4	5
Not at all true		Somewhat true		Very true

4. I like games.

1	2	3	4	5
Not at all true		Somewhat true		Very true

5. Being a good game player is an important part of who I am.

1	2	3	4	5
Not at all true		Somewhat true		Very true

IIG-C CIRCLE ONE: **PRE** or **POST** Study

6. I enjoy games.

1	2	3	4	5
Not at all true		Somewhat true		Very true

7. Playing games helps me in my daily life outside of school.

1	2	3	4	5
Not at all true		Somewhat true		Very true

8. Games are exciting to me.

1	2	3	4	5
Not at all true		Somewhat true		Very true

Noobs Individual Interest Survey in Geometry

II-Geo CIRCLE ONE: **PRE** or **POST** Study

Name:____________________ Date:______________

School:____________________

INSTRUCTIONS: Using the scale provided, please rate how much you agree or disagree with each statement. Please **CIRCLE** your answer.

1. I enjoy doing geometry.

1	2	3	4	5
Not at all true		Somewhat true		Very true

2. I like geometry.

1	2	3	4	5
Not at all true		Somewhat true		Very true

3. Geometry helps me in my daily life outside of school.

1	2	3	4	5
Not at all true		Somewhat true		Very true

4. I enjoy the topic of Geometry.

1	2	3	4	5
Not at all true		Somewhat true		Very true

5. Geometry is practical for me to know.

1	2	3	4	5
Not at all true		Somewhat true		Very true

6. Geometry is exciting to me.

1	2	3	4	5
Not at all true		Somewhat true		Very true

Appendix B

Hand Pose Inference: A Post-Mortem

Earlier, I discussed my joint work with Jonathan Tompson to track the full pose of a human hand in real-time from a single depth camera. Collaborative projects often allow you to accomplish something greater than what you can do alone, but they also obscure who did what. Our collaboration was a good example of this. I think we succeeded because we were able to do different things well. In order to give due credit for his own PhD work, I would like to tell the story of our collaboration on this project. I hope that this story will be useful to other young researchers who are still figuring out how to do collaborative research.

The root of this project was a desire to improve the useful, but limited finger tracking capabilities that Ken Perlin and I had obtained using real-time image processing techniques for use in the ARCADE project. Following our live demonstration at SIGGRAPH, Jonathan proposed we work on the challenging unsolved problem of inferring the full 3D pose of the hand (finger positions, joint positions, palm location, etc.) from the Kinect depth camera stream. I loved his challenging proposal, and we spent the next few months researching different approaches we could try. Three different techniques emerged from this research, each with its own advantages and disadvantages.

The first technique used machine learning, and a lot of training data, but it was fast. A large group at Microsoft Research [72] had shown that if you have enough training data, simple randomized decision forest classifiers could be used to do per-pixel body part classification from a single depth image for the whole body. In fact, this is exactly the approach Microsoft used in the first generation Kinect to track the human skeleton, and it boasted 218fps using a single standard GPU.

The second technique used analysis-by-synthesis and it required a parameterized hand-model. A group out of Greece, presumably led by Oikonomidis [62], had shown you can find a reasonable fit of the hand model to the incoming range image by randomly sampling parameters, rendering each hand into the z-buffer, and picking whichever one led to the smallest pixel wise difference. The advantage of this approach was that it converged on a decent fit with fewer than 10,000 samples. The disadvantage was that it was computationally expensive and therefore limited to low frame rates (e.g. 4-15fps).

The third technique used some variant of iterative-closest-point and it required a parameterized 3D model and a good guess. Several groups tried this idea on human bodies, excluding most degrees of freedom of the human hand. In principle, there was no reason why you couldn't apply it to hands. The basic idea was to pull on the model to minimize the total distance between the model and the data. For example, you could project each data point onto the model, and pull on the model along that vector until an equilibrium is reached. This method is guaranteed to converge on a local optimum under mild assumptions, so if the model starts out close to the global optimum, there is a good chance it will find it, or else an approximation to it. In general, however, when the data and model are not close enough, the technique will converge to meaningless solutions that are very bad approximations of the global optimum. Hence, the good guess I mentioned earlier is a good guess of the starting pose of the model.

Based on this research, it seemed like an obvious thing to try and combine the first two approaches. Only later, after fruitful discussions with Ken Perlin and Ashish Myles, did I learn how to utilize the third approach as well. My idea was to use the slow, brute-force analysis-by-synthesis technique in an offline fashion to generate a large dataset which could be used to train a "standard" machine learning system. The dataset would consist of pairs of depth images and parameter vectors which describe the best-fitting model to that depth image. Oikinomidis had used a generalized cylinder model of the hand, and I felt we could create our own version of that simple model from scratch. To segment the hand from the background, we would use Jonathan's already robust hand classifier based on randomized-decision forests. Once we had a sufficiently large training set, we could attempt to learn a hand-part classifier using some standard machine learning technique such as the randomized decision forests used by Microsoft Research, or using Convolutional Neural Networks. Jonathan really liked my approach and we got to work.

Jonathan immediately took the lead in developing the dataset, and he was the first to implement the Oikinomidis paper. Soon after, I implemented a variant of this using simpler random sampling techniques in order to assess how important PSO specifically was. It quickly became clear to us that the PSO routine itself was an important way to sample this high-dimensional parameter space. Next, Jonathan discovered that the generalized cylinder model was not very good at capturing the high-DOF of the hand, and that self-intersection was a major problem in the fitting routine. He found the libhand project [87], which had a very nice linear-blend-skinning hand model. It matched the actual depth images of our hands much more closely than the cylinder model. Jonathan also found the Assimp library to load this complicated model into his renderer. Next he added spheres to the joints of the model to prevent self-intersection. These two innovations and a few others he came up with improved the original Oikinomidis algorithm significantly.

Meanwhile, I started experimenting with my own synthetic hand model to train a randomized decision forest classifier based on the technique taken by Microsoft. I segmented each finger into 3-4 sections, and made two-cylinders for the palm, three for the thumb, and 1 for the wrist. To generate synthetic parameter vectors, I randomly sampled joint angles for each cylinder, essentially taking tiny little samples in a roughly 30-dimensional space. Based on a dataset of 10,000 images, I trained a randomized decision forest classifier (RDF) for per-pixel labeling. The RDF learned the training set very well but on real images of the hand it was basically useless. The results were disappointing to us. Since Jonathan had just finished generating our first dataset of real images, he started to work on trying to train a Convolutional Neural Network (CNN).

The idea we came up with for the CNN was to guess the parameters of the hand model from a single depth image based on the training data. This proved infeasible pretty much from the start. Jonathan got the idea to train the CNN to predict heat maps of UV hand part locations from Yann LeCun, which proved to be much better. Eventually Jonathan's system demonstrated that it was possible to use a CNN to infer UV positions of at least 10 hand positions from a single depth frame. The problem was that the predicted positions were often far enough from the optimal position that joints jiggled from frame to frame or were just not close enough to the true position.

My main contribution at this stage was to show how to use energy-minimization to clean up a decent first guess from Jonathan's CNN. Ken Perlin was actually the first one to articulate this to us. My approach grew out of several productive conversations with Ashish Myles, who suggested a simple formulation based on ICP in parameter space instead of Euclidean space. His suggestion was to take partial derivatives with respect to the model parameters of the total energy given by the sum of squared distances between the data and model. I estimated the partial derivatives numerically using central differences, and then used BFGS for energy minimization. Although this approach achieved about 4fps, it was too slow for what we needed. Instead, I realized this approach could be reused for two separate problems that Jonathan was having with his system. First, we wanted to get more training data from multiple cameras. My energy-minimization approach was exactly what we needed to align the multiple point clouds from each Kinect into a consistent reference frame. Second, we needed to clean-up the first guess provided by the CNN using some form of IK. My contribution was to show how to do IK using this simple BFGS energy minimization approach. In the end, John ended up using PSO to do the IK instead of my approach because the UV located points from the CNN did not always fall on actual 3D points in the original depth image, and so PSO was more robust.

Bibliography

[1] H. Abelson, M. Chang, E. Mustafaraj, and F. Turbak. “Mobile phone apps in CS0 using App Inventor for Android: pre-conference workshop”. In: vol. 25. 6. 2010, pp. 8–10.

[2] Harold Abelson and Andrea diSessa. *Turtle Geometry: The Computer as a Medium for Exploring Mathematics*. MIT Press, 1986.

[3] B. Allen, B. Curless, and Z. Popović. “The space of human body shapes: reconstruction and parameterization from range scans.” In: vol. 22. 3. ACM, July 2003, pp. 587–594.

[4] Erik Andersen, Yun-En Liu, Rich Snider, Roy Szeto, and Zoran Popović. “Placing a value on aesthetics in online casual games”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM. 2011, pp. 1275–1278.

[5] P. Antonelli. *Design and the elastic mind*. Museum of Modern Art, 2008.

[6] M. Arnold. *The Story of Underdog, Tennessee Tuxedo and the Rest*. Created and Produced by Total TeleVision Productions. BearManor Media, 2009.

[7] L. Ballan, A. Taneja, Jürgen Gall, L. Van Gool, and M. Pollefeys. “Motion capture of hands in action using discriminative salient points”. In: *Proceedings of the 12th European conference on Computer Vision - Volume Part VI*. ECCV’12. Florence, Italy: Springer-Verlag, 2012, pp. 640–653.

[8] I. Barakonyi, H. Prendinger, D. Schmalstieg, and M. Ishizuka. “Cascading hand and eye movement for augmented reality videoconferencing.” In: 2007.

[9] T. Baudel and M. Beaudoin-Lafon. “Charade: Remote control of objects using free-hand gestures”. In: *Communications of the ACM 36(7)* (1993), pp. 29–35.

[10] Donald L Bitzer, Elizabeth R Lyman, and John A Easley Jr. *The Uses of PLATO: A Computer-Controlled Teaching System*. Tech. rep. Science Lab, University of Illinois at Urbana-Champaign, 1965.

[11] S. Blakemore and U. Frith. *The learning brain: lessons for education*. Blackwell, 2005.

[12] Y. Boykov, O. Veksler, and R. Zabih. “Fast Approximate Energy Minimization via Graph Cuts”. In: vol. 23. 11. Nov. 2001, pp. 1222–1239.

[13] Jerome Bruner. *Toward a Theory of Instruction*. Belknap Press, 1974.

[14] D. A. Butler, S. Izadi, O. Hilliges, D. Molyneaux, S. Hodges, and D. Kim. “Shake’n’sense: reducing interference for overlapping structured light depth cameras”. In: *Proceedings of the ACM annual conference on Human Factors in Computing Systems*. Austin, Texas, USA: ACM, 2012, pp. 1933–1936.

[15] B. Buxton. *Multi-touch systems that I have known and loved*. Bill Buxton‘s Research Website. 2010. URL: `http://www.billbuxton.com/multitouchOverview.html`.

[16] J. Canemaker. *Winsor McCay: his life and art*. Harry N. Abrams, 2005.

[17] R. Collobert, K. Kavukcuoglu, and C. Farabet. “Torch7: A Matlab-like Environment for Machine Learning”. In: *BigLearn, NIPS Workshop*. 2011.

[18] D. I. Cordova and M. R. Lepper. “Intrinsic motivation and the process of learning: Beneficial effects of contextualization, personalization, and choice.” In: *Journal of educational psychology* 88.4 (1996), p. 715.

[19] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. 3rd ed. MIT Press and McGraw-Hill, 2009.

[20] D. Dimitrov, C. Knauer, K. Kriegel, and G. Rote. “On the bounding boxes obtained by principal component analysis”. In: 2006, pp. 193–196.

[21] S. Domagk, R. Schwartz, and J. L. Plass. “Defining Interactivity in Multimedia Learning”. In: vol. 26. 2010, pp. 1024–1033.

[22] C. S. Dweck. *Mindset*. Random House, 2006.

[23] National Center for Education Statistics U.S. Department of Education Institute of Education Sciences. *National Assessment of Educational Progress (NAEP) Mathematics Assessment*. 2013.

[24] A. J. Elliot and C. S. (Eds.) Dweck. *Handbook of competence and motivation*. Guilford, 2005.

[25] D.C. Engelbart. *Augmenting Human Intellect: A Conceptual Framework*. Tech. rep. Stanford Research Institute, on Contract AF 49(638)-1024, 1962, pp. 1–134.

[26] Ali Erol, George Bebis, Mircea Nicolescu, Richard D. Boyle, and Xander Twombly. “Vision-based hand pose estimation: A review”. In: vol. 108. 12. Special Issue on Vision for Human-Computer Interaction. 2007, pp. 52 –73.

[27] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. “Learning Hierarchical Features for Scene Labeling”. In: vol. 35. 8. 2013, pp. 1915–1929.

[28] S. Feiner, B. Macintyre, T. Hollerer, and A. Webster. “A touring machine: Prototyping 3d mobile augmented reality systems for exploring the urban environ-

ment". In: *Proceedings of the 1st IEEE International Symposium on Wearable Computers*. Washington, DC, USA: IEEE Computer Society, 1997, p. 74.

[29] G. Friedland, L. Knipping, R. Rojas, and E. Tapia. "Teaching with an intelligent electronic chalkboard". In: *Proceedings of the ACM SIGMM workshop on Effective telepresence*. New York, NY, USA: ACM, 2004, pp. 16–23.

[30] James Paul Gee. *What Video Games Have to Teach Us About Learning and Literacy.* Macmillan, 2012.

[31] C. Hand. "A Survey of 3D Interaction Techniques". In: vol. 16. 1997, pp. 269–281.

[32] C. Harrison, H. Benko, and A. Wilson. "Omnitouch: wearable multitouch interaction everywhere". In: *Proceedings of the 24th annual ACM symposium on User interface software and technology (UIST '11)*. 2011, pp. 441–450.

[33] Vi Hart. 2012. URL: `http://youtube.com/user/Vihart`.

[34] Thomas Little Heath et al. *The thirteen books of Euclid's Elements*. Vol. 3. Dover Publications, 1956.

[35] B. K. P. Horn. "Closed-form solution of absolute orientation using unit quaternions". In: vol. 4. 4. 1987, pp. 629–642.

[36] Primesense Inc. *OpenNI API 1.0*. Version 1.1. 2012. URL: `http://openni.org/`.

[37] H. Ishii and M. Kobayashi. "Clearboard: a seamless medium for shared drawing and conversation with eye contact". In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. New York, NY, USA: ACM, 1992, pp. 525–532.

[38] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun. "What is the best multistage architecture for object recognition?" In: 2009, pp. 2146–2153.

[39] C. Johnson. *Harold and the purple crayon*. Harper and Brothers, 1955.

[40] A. Kay, K. Rose, D. Ingalls, T. Kaehler, J. Maloney, and S. Wallace. *Etoys and SimStories*. Tech. rep. Viewpoints Research Institute, 1997.

[41] C. Keskin, F. Kira, Y.E. Kara, and L. Akarun. "Real time hand pose estimation using depth sensors". In: *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*. 2011, pp. 1228–1234.

[42] *Khan Academy*. 2012. URL: `http://khanacademy.org`.

[43] A. Krizhevsky, I. Sutskever, and G. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 25*. Ed. by P. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger. 2012, pp. 1106–1114.

[44] M. W. Krueger, T. Gionfriddo, and K. Hinrichsen. "VIDEOPLACE: An artificial reality". In: *ACM SIGCHI Bulletin*. Vol. 16. 4. 1985, pp. 35–40.

[45] M. Lau, M. Hirose, A. Ohgaware, J. Mitani, and T. Igarashi. "Situated modeling: A shape-stamping interface with tangible primitives". In: *The 6th International Conference on tangible, embedded and embodied interaction (TEI '12)*. 2012.

[46] Y. LeCun, F. J. Huang, and L. Bottou. "Learning methods for generic object recognition with invariance to pose and lighting". In: *Computer Vision and Pattern Recognition, Proceedings of the IEEE Computer Society Conference on.* Vol. 2. 2004, II–97–104 Vol.2.

[47] H. Li, R. W. Sumner, and M. Pauly. "Global Correspondence Optimization for Non-Rigid Registration of Depth Scans". In: vol. 27. 5. Copenhagen, Denmark: Eurographics Association, 2008.

[48] H. Li, J. Yu, Y. Ye, and C. Bregler. "Realtime Facial Animation with On-the-fly Correctives". In: vol. 32. 4. ACM, 2013.

[49] L. Linnenbrink-Garcia, A. M. Durik, A. M. Conley, K. E. Barron, J. M. Tauer, S. A. Karabenick, and J. M. Harackiewicz. “Measuring situational interest in academic domains”. In: vol. 70. 4. Sage Publications, 2010, pp. 647–671.

[50] T. W. Malone. “Intrinsic Motivation and Instructional”. In: *Aptitude, learning, and instruction* 3 (1987), p. 255.

[51] J. Maloney, M. Resnick, N. Rusk, B. Silverman, and E. Eastmond. “The Scratch Programming Language and Environment”. In: 2010.

[52] Kevin S McGrew. *Woodcock-Johnson III*. 2001.

[53] Stan Melax, Leonid Keselman, and Sterling Orsten. “Dynamics based 3D skeletal hand tracking”. In: *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*. I3D ’13. Orlando, Florida: ACM, 2013, pp. 184–184.

[54] Rovio Mobile. “Angry Birds”. In: 2009.

[55] I.V.S. Mullis, M.O. Martin, P. Foy, and A. Arora. *TIMSS 2011 International Results in Mathematics*. Tech. rep. TIMSS & PIRLS International Study Center, Boston College, 2012.

[56] J. Nagi, F. Ducatelle, G.A. Di Caro, D. Ciresan, U. Meier, A. Giusti, F. Nagi, J. Schmidhuber, and L.M. Gambardella. “Max-pooling convolutional neural networks for vision-based hand gesture recognition”. In: *Signal and Image Processing Applications (ICSIPA ’11), IEEE International Conference on*. 2011, pp. 342–347.

[57] J. Nagi, F. Ducatelle, G.A. Di Caro, D. Ciresan, U. Meier, A. Giusti, F. Nagi, J. Schmidhuber, and L.M. Gambardella. “Max-pooling convolutional neural networks for vision-based hand gesture recognition”. In: *Signal and Image Processing Applications (ICSIPA), 2011 IEEE International Conference on*. 2011, pp. 342–347.

[58] EngageNY New York State Education Department. *Annotated 2013 3-8 ELA and Mathematics State Test Questions*. 2010-2011. URL: http://www.engageny.org/resource/new-york-state-common-core-sample-questions.

[59] Donald Norman. *The Design of Everyday Things*. Bantam Doubleday Dell Publishing Group, 1988.

[60] S. J. Nowlan and J. C. Platt. “A Convolutional Neural Network Hand Tracker”. In: *Advances in Neural Information Processing Systems*. Morgan Kaufmann, 1995, pp. 901–908.

[61] I. Oikonomidis, N. Kyriazis, and A. A. Argyros. “Efficient model-based 3D tracking of hand articulations using Kinect”. In: *Proceedings of the 22nd British Machine Vision Conference (BMVC‘2011)*. University of Dundee, UK, 2011.

[62] I. Oikonomidis, N. Kyriazis, and A. Argyros. “Efficient model-based 3D tracking of hand articulations using Kinect”. In: *BMVC*. BMVA, 2011.

[63] Seymour Papert. *Mindstorms: Children, Computers, And Powerful Ideas*. Basic Books, 1993.

[64] J. L. Plass, B. D. Homer, and E. O. Hayward. “Design factors for educationally effective animations and simulations”. In: *Journal of Computing in Higher Education 21, 31-61* (2009).

[65] J. L. Plass, B. D. Homer, E. O. Hayward, J. Frye, T. Huang, M. Biles, M. Stein, and K. Perlin. “The Effects of learning mechanics design on learning outcomes in a computer-based geometry game”. In: *Lecture Notes in Computer Science* (2012).

[66] J. L. Plass, P. A. O’Keefe, B. D. Homer, J. Case, E. O. Hayward, M. Stein, and K. Perlin. “The Impact of Individual, Competitive, and Collaborative Mathematics Game Play on Learning, Performance, and Motivation.” In: *American Psychological Association* (2013).

[67] J.L. Plass, R. Moreno, and R. Bruncken. *Cognitive Load Theory*. Cambridge University Press, 2010.

[68] Hans Rosling. *Hans Rosling's 200 Countries, 200 Years, 4 Minutes — The Joy of Stats*. Aired on BBC Four December 7th. 2010.

[69] K. Salen and E. Zimmerman. *Rules of Play: Game Design Fundamentals*. MIT Press, 2004.

[70] M. Schlattmann and R. Klein. "Efficient bimanual symmetric 3D manipulation for markerless hand-tracking". In: *Virtual Reality International Conference (VRIC '09)*. 2009.

[71] D. L. Schwartz and J. D Bransford. "A time for telling". In: *Cognition and instruction* 16.4 (1998), pp. 475–5223.

[72] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. "Real-time human pose recognition in parts from single depth images". In: *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*. CVPR '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 1297–1304.

[73] M. Stein, J. Tompson, X. Xiao, C. Hendee, K. Perlin, and H. Ishii. "ARCADE: a system for augmenting gesture-based computer graphic presentations". In: *ACM SIGGRAPH Computer Animation Festival*. Los Angeles, California, 2012, pp. 77–77.

[74] C. Steinkuehler, K. Squire, and S. (Eds.) Barab. *Games, learning, and society: Learning and meaning in the digital age*. Cambridge University Press, 2012.

[75] Patrick Suppes. *Computer-Assisted Instruction at Stanford*. Tech. rep. Institute for Mathematical Studies, Stanford University, 1971.

[76] P. Tseng. *Fortified-Descent Simplicial Search Method: A General Approach*. Tech. rep. SIAM J. Optim, 1995.

[77] Donald L. Vickers. *Sorcerer's Apprentics: Head-Mounted Display and Wand*. University of Utah, Doctoral Thesis, 1974.

[78] R. Y. Wang and J. Popović. "Real-time hand-tracking with a color glove". In: *ACM SIGGRAPH 2009 papers*. SIGGRAPH '09. New Orleans, Louisiana: ACM, 2009, 63:1–63:8.

[79] R. Wang, S. Paris, and J. 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. Santa Barbara, California, USA: ACM, 2011, pp. 549–558.

[80] R. Wang and J. Popović. "Real-time hand-tracking with a color glove". In: 63. 2009, pp. 1–8.

[81] T. Weise, H. Li, L. Van Gool, and M. Pauly. "Face/Off: Live Facial Puppetry". In: *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer animation.* New Orleans, Louisiana: Eurographics Association, 2009.

[82] A. Wilson and H. Benko. "Combining multiple depth cameras and projectors for interactions on, above and between surfaces". In: *In Proceedings of the 23nd annual ACM symposium on User interface software and technology (UIST '10).* New York, NY, USA: ACM, 2010, pp. 273–282.

[83] J.A. Wise, V.D. Hopkin, and D.J. Garland. *Handbook of Aviation Human Factors*. CRC Press, 1998.

[84] T. Yasuda, K. Ohkura, and Y. Matsumura. "Extended PSO with partial randomization for large scale multimodal problems". In: *World Automation Congress (WAC), 2010*. 2010, pp. 1–6.

[85] B. Yee, Y. Ning, and H. Lipson. "Augmented reality in-situ 3d sketching of physical objects". In: *Intelligent User Interfaces Workshop on Sketch Recognition, ACM, New York, NY, USA*. 2009.

[86] T. G. Zimmerman, J. Lanier, C. Blanchard, S. Bryson, and Y. Harvill. "A hand gesture interface device". In: *Proceedings of the SIGCHI/GI conference on Human factors in computing systems and graphics interface, John M. Carroll and Peter P. Tanner (Eds.). ACM, New York, NY, USA*. 1987, pp. 189–192.

[87] M. Šarić. *LibHand: A Library for Hand Articulation*. Version 0.9. 2011. URL: http://www.libhand.org/.