The Pennsylvania State University

The Graduate School

College of Information Sciences and Technology

AIDING THE USER INPUT TO VIRTUAL TRAINING ENVIRONMENTS: VIRTUAL ROLE PLAYERS WITH SPEECH AND GESTURE RECOGNITION

A Thesis in

Information Sciences and Technology

by

Robert Floyd Stark

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science

May 2010

This thesis of Robert Floyd Stark was reviewed and approved* by the following:

John Yen
Associate Dean for Strategic Initiatives, College of Information Sciences and Technology
University Professor of Information Sciences and Technology
Thesis Co-Advisor

Frank E. Ritter
Professor of Information Sciences and Technology
Professor of Computer Science and Engineering
Professor of Psychology
Thesis Co-Advisor

C. Lee Giles
David Reese Professor of Information Sciences and Technology
Professor of Computer Science and Engineering
Professor of Supply Chain and Information Systems

Frederico Fonseca
Interim Associate Dean for Education, College of Information Sciences and Technology
Associate Professor of Information Sciences and Technology
Director of Graduate Programs

* Signatures are on file in the Graduate School

ABSTRACT

The purpose of this thesis is to address the fact that users' input to training systems in virtual environments is not suited to their natural skills and abilities. These skills and abilities include speaking and gesturing with their bodies. This mismatch may have negative effects on their usage of the virtual environment. One assumption guiding this thesis is that it would increase immersion to allow the users to interact with the system in the same way they interact with real people. The second assumption is that multimodal input can increase users' performance in the training scenario, especially regarding habitual and physical skills. While people use the mouse and keyboard inputs to computers all of the time, the third assumption is that natural speech and gestures would make military virtual training systems even easier to get acquainted with and use. The fourth assumption is that more natural systems may increase the amount of training that trainees can transfer to the real world.

To show the potential of the approach of multimodal input, two prototype systems were created. The design and evaluation of the first prototype are described. It was intended to show the potential of gesture recognition and multimodal fusion under both ideal theoretical circumstances and controlled, but more realistic, ones. The primary problem with the first prototype was found to be the limitations with the hand recognition and tracking system. The design of the second prototype is then described. This prototype is a fully-operational virtual checkpoint training system with multimodal input and was created based on the hand tracking and other insights from the first prototype. Then the results of a demonstration at a conference are explained, including environmental factors on its usage. The thesis ends with a discussion of the insights from the last prototype and some future work, including implementation ideas, empirical studies, and general guidelines for multimodal system design.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to first thank my co-advisers, Professors Frank Ritter and John Yen, for their help guiding me through my graduate school experience and the development of my thesis. I would like to further thank them, and also Professor Lee Giles, for their helpful comments and recommendations for the improvement of this thesis at its defense.

Also deserving of many thanks are my mentor and BBN coworker, Dick Pew, and my very patient and loving wife, Megan Kierstead, who read my thesis before its defense and provided me with suggestions for improvement. I would also like to thank my previous manager and coworker, David Diller, for allowing me to work on this very interesting project from start to finish.

Finally, I would like to thank my employer, BBN Technologies, for sponsoring my continued master's work concurrent with my employment. Without this funding, I would not have been able to complete either the graduate classes required for my master's degree, nor used the VRP prototype system as a subject of this thesis.

Section 1: Introduction

Simulations used by the military have been characterized as live, virtual, or constructive (LVC). Live training is using real people operating real devices, but under simulated circumstances. Constructive simulations are those in which trainees provide input to the simulations, but do not actively play roles to determine the outcome. Finally, virtual simulations are using trainees to actively control virtual characters in a virtual environment analogous to real environments (Frank, Helms, and Voor, 2000; Page and Smith, 1998).

Problems with Virtual Training Systems

While virtual environments hold much promise for training skills, their application to such a domain still poses many challenges. According to Caird (1996), features that should be part of any training simulation system that hopes to be effective should include cost effectiveness, good interface usability, transfer of the training to the real world, feedback on the training, a match to theoretical constructs like fidelity and presence, and a system design based on task analysis. This research addresses the issues of usability and transfer of training, in addition to other benefits of natural user input modalities.

This research makes the primary assumption that the standard mode of computer input—a mouse and keyboard—is an unnatural way to train real-life skills. Furthermore, it is assumed that the resulting lack of realistic simulation may make the users feel less immersed in the virtual environment, may reduce their ability to perform well and improve in the simulations, and may increase the time and effort to learn to use the system to its full capacity. As a result, it is assumed that these effects on the users may cause inefficient transfer of the trained skills and

knowledge to the real world upon deployment, which is the whole reason for the training exercise.

Solution

In light of the problems facing virtual training environments, the solution attempted in this study is the idea of a "virtual role player," which plays roles such as civilians or insurgents and reacts in meaningful ways to actions performed by the user, while at the same time being controlled by the computer. Non-player characters (NPCs) are usually scripted to react in predictable ways and to focus on counter-attacking the trainees. Conversely, in the Virtual Role Players (VRP) training system, the NPCs' reactions are based on both the trainee's speech and hand gestures, simulating real-world interaction. Because these virtual role players respond to automatically-recognized speech and gestures from the user, the input is more natural than a keyboard and mouse. This, in turn, makes the training experience more realistic and immersive and allows the trainees to transfer what they learn from the training to the real world. Also, by using the trainees' natural knowledge and skills, they should be able to achieve higher performance and this makes the system easier to use.

The VRP project is aimed at making a prototype training simulation in Virtual Battlespace 2 (VBS2)—the current US Army standard game-based training platform—in the domain of checkpoint security. In a checkpoint security scenario, the user is tasked with asking people to stop their cars, approach the checkpoint, continue through the checkpoint, get out of the car, and move to various places.

Preview of this Thesis

This thesis starts with an overview of the current human factors problems with the usage of virtual environments for training, followed by an overview of gesture-based input mechanisms and the recognition of these gestures by a computer system. The subsequent section outlines the resulting theoretical foundations of a first prototype, its design and implementation, and the results of its theoretical and pilot study evaluations. Following that is a section on what the evaluation results meant for a more robust second prototype. Details of the second prototype's design and conference evaluation are then mentioned. The thesis ends with a discussion of the implications and lessons learned from these prototypes and future work for the successful application of multimodal input for training in virtual environments. The contributions of this thesis therefore are the aggregation of theoretical foundations for multimodal input related to virtual environments, creation and evaluation of the two prototypes, and the insights drawn from these evaluations, including their promise of accurate recognition and their requirements of environmental precautions required for deployment.

Section 2: Human Factors Problems in Virtual Training

Keyboard and mouse inputs are problematic in virtual training systems that are aimed at training physical, spoken, or other similar skills. Trainees, the users, generally have to use standard keyboard and mouse inputs. These controls are inherently unnatural, possibly resulting in less immersion into the virtual environment, reduced performance from the trainees, a perception of system complexity, and a reduction in transferred learning to the real world—the entire purpose of the training exercise.

Some work has been done addressing the issues related to object and character realism in virtual environments (e.g., Gordon, Casey, Burns, and Cohn, 2001). Much of the work that has been done addressing interaction modalities is about the making and evaluation of the systems that utilize natural speech or language. Some examples include a dialog training framework and spoken language prototype created by Luperfoy, Domeshek, Holman, and Struck (2003), intelligent agents to act as foes and teammates that primarily communicate with military-prescribed/limited text and speech (Nielsen, Koss, Taylor, Jones, 2000), a ship navigation system that allows trainees to ask questions (Everett, Tate, Maney, and Wauchope, 2000), and even speech- and gesture-enabled systems, though most having less-than-desirable recognition accuracies or virtual world fidelity (Cavazza, Charles, Mead, Martin, Marichal, and Nandi, 2004; Poddar, Sethi, Ozyildiz, and Sharma, 1998).

The systems created in the above studies made important contributions to the scientific endeavor of creating better interfaces, but, for future systems that hope to build on their work, there are human factors metrics that should be addressed.

Immersion

Immersion is the feeling that the user has that they are, in some sense, "involved" with the virtual environment, NPCs and scenario being used for training. Another similar idea is that of *presence*—the feeling of "being there." Virtual environments inherently offer the benefit of visual immersion, but the user experience may be improved if the user's input were also more natural and thus be made to "fit in" with the environment.

While simple measures can be taken to increase immersion, such as reducing network latency (Kappé and de Vries, 2000), both the input and output of the virtual training systems can be improved as well. Research has been done to increase the immersion value of the output,

including using head-mounted displays and augmented reality (Stone and McDonagh, 2002; Stone and Rees, 2002). Similarly, natural modes of interaction can increase immersion and the feeling of presence in a virtual environment (Witmer and Singer, 1998).

Human Performance

Though using the standard types of computer input for virtual training affects the trainees' feeling of immersion in the scenario, whether it affects their performance is another question entirely. One study showed that multimodal output can increase human performance (Stanney, Mourant, and Kennedy, 1998), though this may not translate as well to input. Another study showed that multimodal input can, in some circumstances, especially spatial in nature, decrease task-related errors (Oviatt, 1997). Yet another study explained how sometimes a more natural interface may make more sense, even if it causes users to be less efficient (Jacob, Girouard, Hirshfield, Hom, Shaer, Solovey, et al., 2008). This seems especially true in the case of training systems, where it may be beneficial for input to mirror real actions.

It is also important to note that sometimes the loss in efficiency is all in the users' perceptions. In a study concerning a pressure mat created to allow for natural movement forward, back, left, and right, users reported that their movement seemed slow and inefficient, though measurements showed that the times for a given distance were "reasonable" (Couvillion, Lopez, and Ling, 2001).

There have been very diverse approaches to optimizing the trainees' performance in virtual training systems. For example, Salas, Rosen, Weaver, Held, and Weissmuller (2009) prescribe 11 general guidelines in simulation-based training. When constructing scenarios to train people, one should:

- Know the competencies that are being trained,

- Develop learning outcomes,
- Identify metrics and performance markers for each outcome,
- Establish diagnostic performance metrics,
- Create an integration plan for monitoring and measurement,
- Give instructors and observers the tools they need,
- Ensure instructors and observers are not over-burdened, and
- Train instructors and observers to maximize their efficacy.

A separate study about human factors outlined three dimensions that can be considered in a virtual training system: the level of user expertise, the type of knowledge and skills being taught, and the type of game upon which the training system is based (Anthony, Chandler, and Heiser, 2009). While these guidelines are closer to the issue of human factors, it is studies like the one done by (Stanney et al., 1998) that are most relevant to designers of multimodal virtual training systems. In their study, they note that the performance of the users of virtual environment training systems is affected by the characteristics of the tasks that are being trained, the characteristics of the users themselves, the sensory limitations of the users that constrain the system, the integration of the multimodal input and output, and the design metaphors with which the system is made.

Ease of Use

Another reason speech and gesture input systems are more natural is because they utilize the trainees' inherent abilities. This naturalness makes the system *easy to use* because the user does not need training on how to use the system. While many people are skilled with standard modes of input (i.e., keyboard-and-mouse input), using body gestures and speech as input for virtual training systems would further increase their naturalness and, therefore, their ease of use.

Research has been done on "reality-based" input modalities—including tangible computing, speech input and gesture input—and how these inputs take advantage of humans' understanding of *naive physics*, their *body awareness and skills*, their *environment awareness and skills*, and their *social awareness and skills* (Jacob et al., 2008). In other words, humans have an innate understanding of how physics work, and they have awareness of their bodies, their external environment and social situations, and they have basic skills to take advantage of this awareness.

Speech and gesture interfaces can be especially natural modes of input in the case of communicating with virtual characters because it is just like talking to other people, and this, in turn, has the potential to make it easier for the user. Speech input takes advantage of their existing social awareness and skills. That is, the users are assumed to know how to speak. Gesture input, on the other hand, takes advantage of users' body awareness and skills and their environment awareness and skills. The user is assumed to know how to use their body and how to interact with their environment with their body to make gestures. It could also be said that the gesture input takes advantage of their social awareness and skills if they are using body gestures for the purpose of social communication.

Transfer of Learning

Training knowledge and skills in virtual environments is largely concerned with their successful transfer to the real world. Therefore, to make a useful training system, one should maximize the *transfer of learning*. That which is transferred can be anything from the know-how to do an activity like flying a plane, to being more efficient at negotiating with people. In any of these cases, the *transfer of learning* could be measured by the trainee's efficacy in the task or skill before training compared with the efficacy after training. Another way that measures

training in virtual environments is to compare the trainees after live training versus after virtual training.

In such a study of sensorimotor training, Rose, Attree, Brooks, Parslow, Penn, and Ambihaipahan (2000) studied whether the training in a virtual environment transferred to doing the task in the real world as well as live training. Previous work had been done that indicated that most types of tasks do transfer very well to the real world compared to live training, but the authors suspected that the details of the transfers may differ and that some types of learning may show significant differences. They found that virtual training transferred to the real world just as well as live training, and in one case—where cognitive interference existed—it possibly was more effective than live training. This study and others like it (e.g., Holden, 2005) show that sensorimotor skills can be trained in virtual environments effectively.

However, the skills necessary for military personnel go beyond sensorimotor skills in that they also include various cognitive and social skills. Most relevant to this thesis are skills in communicating that use speech or body gestures. Some research has been done to show that social skills can be taught to people with autism-spectrum disorders using virtual environments (Parsons and Mitchell, 2002). That does not necessarily mean that similar types of training will be useful for the military domain. However, Knerr (2007) did indicate that virtual environment training is effective for cognitive skills, decision making and even marksmanship. Those skills involving locomotion and touch were seen as not even being possible in the simulators. As a result, virtual training is used a supplement to live training, and mostly only for leadership positions (Jones and Mastaglio, 2006).

Summary

There remain human factors issues in virtual environment training systems, and these issues are relevant to the naturalness of the systems. Therefore, it stands to reason that if a system can be made more natural, such as through its input from the user, then these human factors measures can be improved.

Section 3: Gesture Recognition Overview

The previous section discussed the issues of immersion, trainee performance, the training system's ease of use, and the transfer of leaning. A solution to these problems is to create a system that uses speech and gesture recognition for the users' input to mirror their interactions with people in the real world. These input modalities are assumed to be more natural for the user and they have many potential uses in tomorrow's computer interfaces.

Uses of Gesture Recognition

The way that gesture recognition aids user interaction in an information system is determined at least partially by what type of gesture the user is making. Humans make four broad types of gestures, outlined by Wu and Huang (1999), including *conversational*, *controlling*, *manipulative* and *communicative*. These types of gestures lend themselves to three basic information system use cases: interface control and manipulation, computer observation of human interaction, and interacting with virtual environments. The first is active and possibly contrived, while the second is completely natural but passive, and virtual environment interaction is mentioned as a compromise between these two extremes.

Interface Control and Manipulation

Often when people think of gesture recognition for interfaces, they think of using gestures to manipulate a computer interface by gesturing wildly to effect simple actions, or in a way that is analogous to how a mouse controls a computer. The former is akin to the type of interaction seen on the movie "Minority Report," starring Tom Cruise, where he uses his hands to cycle through video frames by moving his hands and fingers in the air in an unnatural way. An example of the latter is an interface that uses fingertips as mouse-analogues on a touch-based desktop interface (Oka, Sato, and Koike, 2002), which, in turn, is very much like the modern-day *iPhone* or *iPod Touch* by Apple, Inc. While these gestures are a potentially interesting replacement for a mouse, they do not capture the naturalness of which gesture interfaces are capable. A more natural mode of using gestures was first illustrated by a pointing-based proof-of-concept system for drawing shapes, called “Put-That-There” (Bolt, 1980).

Monitoring of Human Interaction

Another mode of interfacing with a computer is having the computer monitor people through a camera as they use their gestures as they normally would. This use of gesture recognition is certainly more natural, however it is also passive. These gestures, and therefore their respective meaningful computer inputs, can be broken down into five categories: (Kendon, 1986):

1. *Gesticulation*, when a person gestures for emphasis;
2. *Language-like*, when a person tries to represent some word or set of words without a formal language;
3. *Sign language*, for when a person gestures in accordance with the international sign language definition;

4. *Pantomimes*, when a person tries to communicate somewhat vague ideas with their gestures without words; and
5. *Emblems*, when a person represents some kind of cultural-specific idea—e.g., the size of a fish they caught—with their hands.

An example system of gestures interpreted as sign language was created by Starner, Weaver, and Pentland (1998), while a more holistic example can be found in an analysis of videotaped presentations (both motion and gesture) by Ju, Black, Minneman, and Kimbler et al. (1998).

Interacting with Virtual Environments

A reasonable medium between these extremes of computer control and computer monitoring lies in users interacting with virtual environments. Because of the analogue of the virtual environment to the real world, natural gestures are better than contrived manipulation gestures (Wexelblat, 1995), while, at the same time, at least some of the gestures should have immediate impacts on the simulated environment as they happen.

An early but prescient example is work done at the MIT Media Lab in 1995 by Pattie Maes' group, where gestures were used to interact with virtual agents, of which particular fame went to a virtual dog pet that responded to gestural commands like a real dog (Maes, Darrell, Blumberg, and Pentland, 1997). Another, more recent, example of interacting with virtual environments with gesture was done with some more advanced methods that will be covered below (Demirdjian, Ko, and Darrell, 2005).

Outline and General Approach to Gesture Recognition

Regardless of the way the gestures are used, the recognition happens in the same way. This basic meta-algorithm, outlined by Martin and Crowley (1997), is something along the lines of: identify object, track object, transform features, and recognize gesture.

This section takes on the same form and, thus, will outline the basic problem in computer vision research of recognizing objects in a static scene. Then it will explain the basis of tracking objects over time. Both of these steps are necessary precursors to gesture recognition; therefore, the paper goes into symbolic and non-symbolic approaches to using the data provided by object tracking for gesture recognition. The next subsection outlines ways to create novel and useful interfaces with either or both of these approaches. Finally, the section concludes with some general thoughts on symbolic and non-symbolic approaches.

Static Recognition: Object Detection and Recognition

The first step in gesture recognition is to recognize parts of a person's body in any arbitrary image. This can be called the "static recognition" or object detection phase. An important study on this topic was published by Paul Viola and Michael Jones, eponymously referred to as the "Viola-Jones" algorithm (Viola and Jones, 2002) by those in the field of computer vision. This groundbreaking paper built on a history of object detection and hand pose recognition—e.g., Cui and Weng (1996) and Quek and Zhao (1996), respectively—and made it possible in real time. Building on the work done by Viola and Jones (2002), more recent work was done by Kolsch and Turk (2004) to optimize their algorithm for hand detection in a free application called Hand Vu, although this work is largely overshadowed by work in more robust hand and body tracking, explained below.

Dynamic Recognition

While static recognition is necessary and useful for gesture recognition, much recent work has been done on dynamic recognition of gestures and movements over time. Dynamic gesture recognition is done in three basic steps: 1. tracking the object(s), 2. extracting and transforming features, and 3. the recognition of the gestures based on these features.

Object Tracking

A general overview of object tracking was written by Gavrila (1999), who explains how the focus of much modern computer vision research is "looking at people" while they are doing things. While the HandVu makers designed their hand detection algorithm for the purpose of tracking it over time (Kolsch and Turk, 2004), more interesting work is being done in relation to tracking full human bodies. Interesting and novel attempts, with various levels of success, have been accomplished in this area (Bregler and Malik, 1998; Moeslund and Granum, 2001; Pavlovic, Rehg, Cham, and Murphy, 1999; Wren and Pentland, 1998).

Feature Transformation

Often the tracking systems can produce coordinates for either the body part (Kolsch and Turk, 2004) or of various body parts at the same time. While useful to have simple coordinate features, much improvement has been seen after transforming them into more semantically-meaningful ones. These transformed features could be as simple as the change in the coordinate values (Starner et al., 1998), vector-valued 2D motion trajectory (Yang, Ahuja, and Tabb, 2002), simple 3D features (Campbell, Becker, Azarbayejani, Bobick, and Pentland, 1996), or could be more complex like arm length, body lean, angles of body joints, and so on.

Gesture Recognition

After having sufficiently-meaningful features, the next task is to use them to recognize gestures. This can be done with algorithms that are deterministic, as with “symbolic” or logical artificial intelligence methods of the slightly-more-distant past, or nondeterministic, as the “non-symbolic” or statistical machine learning methods that have prevailed in the recent past.

Symbolic and Non-Symbolic Artificial Intelligence

“Symbolic” AI systems, also sometimes called logical systems, can trace their roots back to computational-theoretic concepts such as finite state machines (FSMs) and "Good, Old-Fashioned AI" (GOFAI). GOFAI can include anything from a simple rule-based system that responds to commands with sentences that *seem* meaningful, like the popular ELIZA program (Weizenbaum, 1966), to heuristic search. Since 1966, however, rule based systems have become more complex, especially since the introduction of the Physical Symbol Systems Hypothesis (Newell and Simon, 1976), and the advent of cognitive architectures like Soar (Laird, Newell, and Rosenbloom, 1987) and ACT-R (Anderson, 1996).

“Non-symbolic” systems, or sometimes called probabilistic or statistical systems, on the other hand, can trace their roots back to statistical classifiers such as artificial neural networks and Bayesian decision theory. These classifiers can be used for:

- Pattern recognition on novel input, such as for the aforementioned previous steps in gesture recognition,
- Machine learning, to make computers improve over time, and for
- Data mining, to find out novel pieces of information in large data sets.

Classification may be done in a supervised way, where the model is trained on a data set with class labels; an unsupervised way, where the model is learned from the organization of the data without knowing anything about its classes; or a semi-supervised way, where the *a priori* organization of the data is used to improve the performance of a supervised classifier, or vice versa.

While many AI systems of the past can be classified as “symbolic” or “non-symbolic,” much recent work has been utilizing what could be called “hybrid AI.” These hybrid systems can be ones that use logical and probabilistic components or ones that are both symbolic and probabilistic themselves. Some examples include work done in hybrid cognitive architectures (Sun, 2006), statistical grammars (Hwa, 2000), probabilistic rule derivation (Ullman, Baker, Macindoe, Evans, Goodman, and Tenenbaumet, to appear), and decision tree induction (Quinlan, 1993).

Deterministic Gesture Recognition

Deterministic gesture recognition, like “symbolic” AI, can mean anything from finite state machines and languages and algebras to rule-based systems and production rule cognitive models. In the case of state-based systems, an *a priori* definition of gestures in the form of states and their transitions is matched against actual data to determine the "fit." Some examples include Davis and Shah (1994), which used FSMs to recognize hand gestures, and Bobick and Wilson (1995), which tried something similar for trajectories of simple mouse and hand movements. A more complex type of symbolic AI used a disjunctive normal form model to induce a set of rules to describe and recognize gestures (Quek and Zhao, 1996).

The positive aspects of this approach are that no training is necessary, as it is for many of the non-symbolic approaches below. Another benefit is that it is easy to implement, at least in the

cases of the FSMs and simple rule-based systems. Also, it allows for control over exactly what is being matched against for each gesture. Finally, as a result of the more precise control, it works quite well for simple gestures in prototype systems with development time constraints.

On the other hand, this approach does not gracefully handle errors in data or missed frames in the video because any incorrect states or data points could cause an incorrect classification or could result in no classification possible at all. It also does not handle diverse ways of making any particular gesture, since it is matching an *exact* definition of the gesture against the input; if a person makes the gesture slightly differently, it may be incorrectly classified as a result.

Nondeterministic Gesture Recognition

Nondeterministic approaches to gesture recognition are promising ways to deal with many of the shortcomings of deterministic approaches. In general, they are more robust in real-world conditions. On the other hand, this benefit comes with a price.

Without a doubt, the most popular non-symbolic way to recognize gestures is to use what is called a hidden Markov model (HMM). HMMs are graphical models that represent a transition in states, much like a finite state machine. However, unlike the finite state machine, HMMs do not have to be defined beforehand or *a priori* and, unlike finite state machines, are able to probabilistically match gestures against input.

An HMM models a gesture by breaking it up into *n* states (*n* being predefined) that each have usually the same statistical distribution (the type of which is also predefined). The states, or the "real" states so to speak, are also hidden by another layer of nodes called the *observation nodes*, making the real states the *hidden layer*. The primary question with an HMM is what the probability is that, given an observed node (or set of observed nodes), the hidden state (or set of

hidden states) is one particular state (or that one particular HMM is correct). Another important detail is that each state, in addition to being related to its observation node, is also dependent on the value of the previous hidden state node, thus making the full model a Markov chain—a graphical model where each node depends on a finite number of previous states (usually one, in the case of a first-order Markov chain).

The HMM is trained with the Baum-Welch algorithm—a specialized form of the expectation maximization (EM) algorithm—to determine the parameters of each state's distribution. Once trained, the HMM can be compared against an arbitrary input set by using the Verterbi algorithm—a specialized form of dynamic programming. A more complete explanation of HMMs can be found in a paper by Mitra and Acharya (2007). Examples of using HMMs for gesture recognition include Wilson and Bobick (1995, 1999), which formulated much of the foundational work in this area, and Starner et al. (1998), which used HMMs to interpret human sign language.

Other nondeterministic methods that have been explored include Dynamic Time Warping (DTW) (Corradini, 2001), a method for calculating a distance between time series; dynamic Bayesian networks (DBNs) (Pavlovic et al., 1999), a method similar to HMMs but, instead of a single observation node used to determine the hidden state node, a full Bayesian network models each state; and support vector machines (SVMs) (Avidan, 2004), a standard supervised learning algorithm applied to temporal data to distinguish between classes of gestures.

The benefits of these nondeterministic methods are that they can handle errors and omissions in the data gracefully, since they probabilistically model and match the data. Therefore, if the input data is imperfect, there is still a chance that the correct classification will be made. Also, for many of these approaches, they are perfectly suited to be models of gesture

data sets because of their temporal nature. However, these methods very often do require training, which can take time and also requires good data to be collected. Also, evaluating them is sometimes strenuous in terms of time complexity. For example, the HMMs' Verterbi algorithm, because it is dynamic programming, is O(*n***m*) for *n* and *m* being dimensions of the dynamic programming table (the number of states times the number of dimensions in the data set).

Conclusion: Deterministic vs. Nondeterministic

It should be clear after this analysis that deterministic and nondeterministic approaches have their own advantages and disadvantages, and, thus, their own uses in different situations. Deterministic approaches are good choices for prototypes that require quick implementations, especially if they are made for simple gestures with cameras in relatively-ideal lighting conditions. However, sometimes nondeterministic approaches seem to have much more promise. For example, they would be useful if the system is intended for more "realistic" situations with imperfect lighting conditions (and therefore imperfect data). They are also good when there is a diverse set of users that will gesture in different ways. It is also important that there is time to experimentally determine ideal features and to train the models, non-symbolic approaches seem to have much more promise. Furthermore, in situations when there is a mixture of simple and complex gestures, it might even make sense to combine deterministic and nondeterministic approaches to more intelligently classify gestures according to the different models' strengths.

Interface Possibilities: Static and Dynamic, Deterministic and Nondeterministic

Static and dynamic recognition strategies can be combined in interesting ways to create more effective interfaces. For example, a system could recognize hand or body posture with

some form of static recognition, and then track the hand or body with dynamic recognition to get more complex gestures than either approach alone.

Additionally, gesture recognition itself can be combined with other interaction modalities to create more useful interfaces. For example, speech and gesture recognition could be combined to allow the user to tell a character in a virtual environment to "go over there" and point simultaneously. Another example is combining gesture recognition with head or eye tracking; this way, the user could navigate the virtual environment by looking around while making natural gestures within it. Gesture recognition, and really any modality, could be combined with the future's "brain-computer interfaces," allowing the user to think about some things they would like to do, while gesturing or speaking for other types of inputs.

Examples of these types of interaction, and other, types of synergies can be found in various studies. One of them is by (Darrell, Gordon, Harville, and Woodfill, 2000), which mixes color and face static detections with dynamic user tracking; another was done by (Demirdjian et al., 2005), which uses static recognition for body poses, dynamic gesture recognition and speech recognition for interacting with the virtual environment, using HMMs and SVMs to classify these inputs, and then multimodal fusion methods to combine them.

Section 4: First Gesture Recognition Prototype

The first gesture recognition prototype system was created as a proof-of-concept of a software system that automatically detects hand gestures and that these gestures could be used in a military training scenario. The intention was to create a prototype system using free and available support systems and libraries, and to create something that fulfills the existing gaps in the human factors of virtual training systems.

System Details

As a proof-of-concept for the technical aspects of the gesture recognition and multimodal fusion, this prototype was very much focused on testing to see how well recognition would work, rather than creating a full prototype training system. That task was saved for the second prototype, discussed below. This prototype was created to experiment with the recognition accuracies in ideal conditions (called "offline" recognition) and in more realistic ones (called "online").

The types of gestures that were recognized were intended to be the same or similar ones in a checkpoint security training scenario. As a result, it supports four speech-gesture types, or six total combinations:

- "Stop" / hold palm forward to approaching car
- "Go over there" / point either (a) left or (b) right
- "You can go ahead" / waving from the car to the (a) left or (b) right
- "Come on forward" / palm facing self, move hand back to self and up

Architecture and Design

The gesture recognition system (see Figure 1) is designed to modularize the functionality that is common to both the online application and the offline training application, and to abstract away from the specifics of the HandVu hand tracking system. The rationale behind the former is good object-oriented design, while the purpose for the latter is that the VRP project may eventually change to a better hand tracking system, for reasons outlined later in this paper. HandVu was useful for this project because it is a freely available and open source hand tracking solution. It is based on the Open Computer Vision (OpenCV) library and is based on a method

called "flocks of features" (Kolsch and Turk, 2004). However, as will be explained later, it is rather ineffective in most environments and for most hand gestures when not used as a head-mounted system.

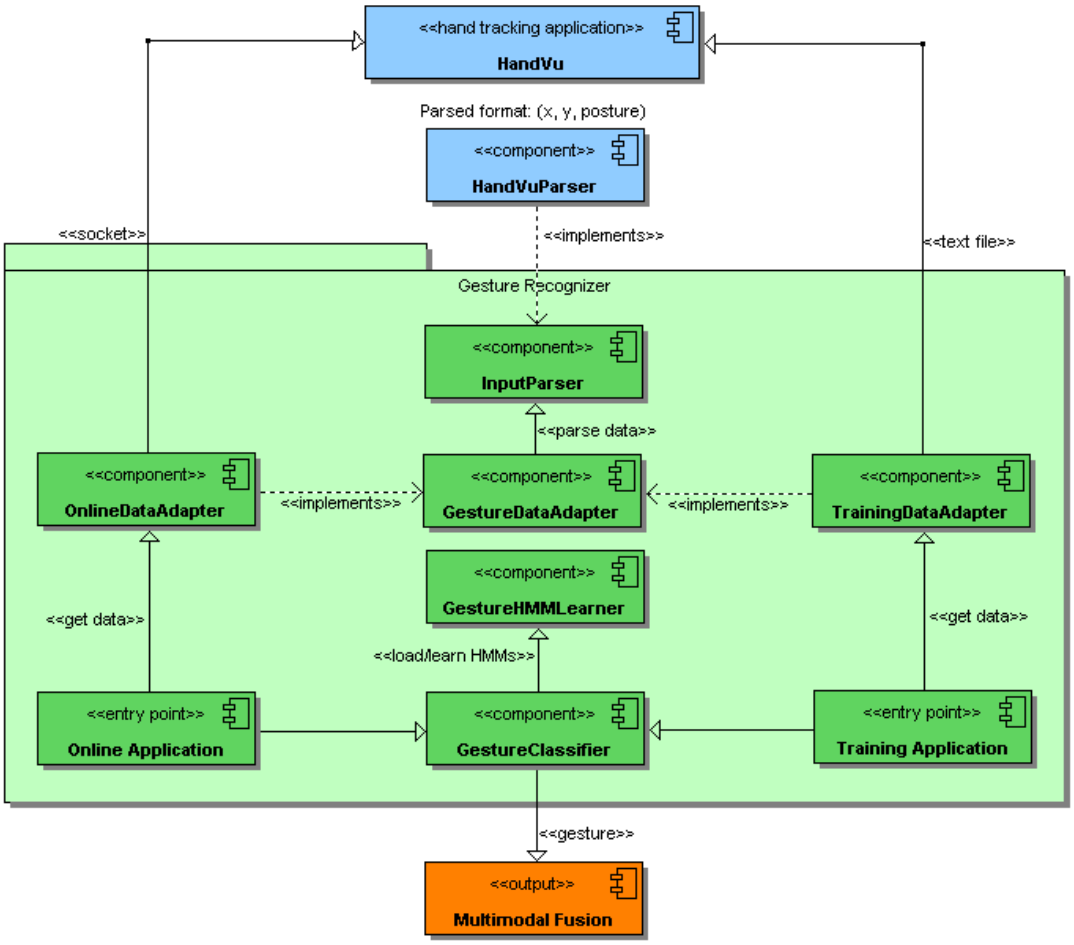

Figure 1: The Architecture of the First Prototype's Gesture Recognizer

Video from the computer's camera is first processed by HandVu to track the hand as it moves across the camera's view. HandVu discovers and reports three key pieces of information about the tracked hand, including the x and y coordinates and its "posture." The coordinates of the hand range from -1, indicating the far left or bottom of the camera's view, to +1, the right and top, respectively. The posture is a particular state of the hand, such as holding up the index and middle finger in a V ("victory") or holding the palm flat with fingers close together ("closed").

These features are passed on to the GestureClassifier via a GestureDataAdapter—either OnlineDataAdapter for information coming directly from the hand tracking system via a socket, or TrainingDataAdapter for hand tracking data located in text files. As mentioned above, these components do not depend on the particular hand tracking system being used. This is because they refer to the InputParser interface, which for this prototype is solely implemented by HandVuParser, which parses the HandVu hand tracking data format.

The GestureClassifier utilizes a cascaded gesture classification algorithm, which first looks to classify a video sequence as being a "stop" gesture by looking for the "closed" posture (see Figure 2). Failing this, the position coordinate data is evaluated against the known hidden Markov models representing the other gestures using the argmax of the log-probability determined by the Verterbi algorithm. The maximum log-probability is compared against a threshold specified in the configuration file and, if lower, the classifier returns null, representing no gesture taking place. The threshold is set to a low value, five, for training since no values seem to accurately remove false examples while retaining positive ones, and is set to 20, a relatively high value, for the online system.

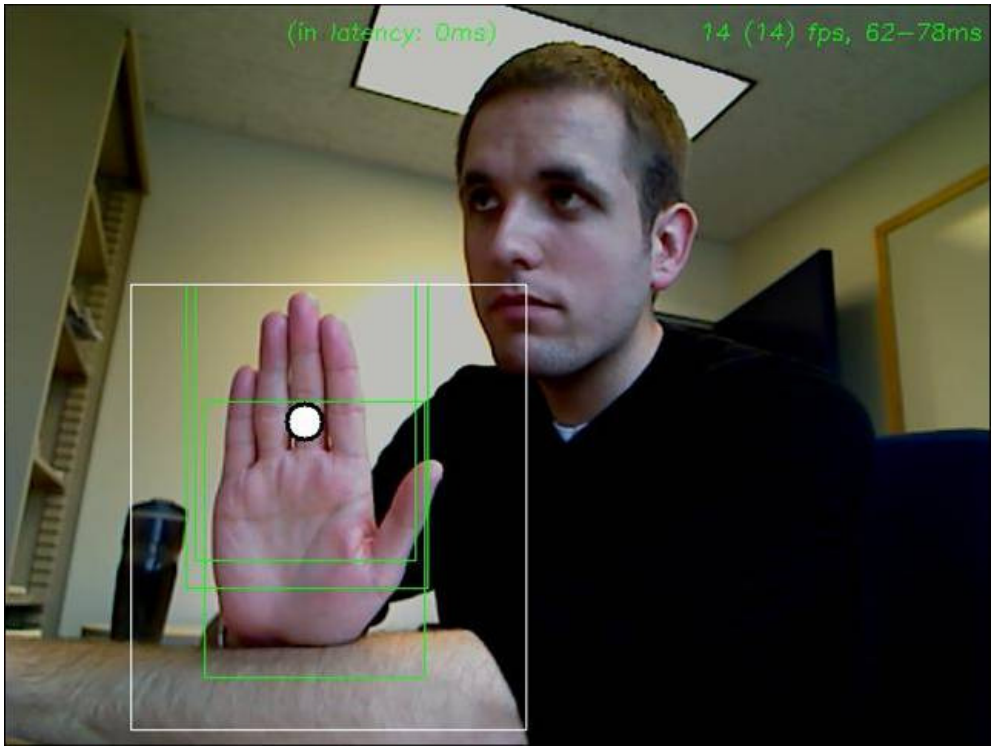

Figure 2: The "Closed" Hand Posture Being Recognized by HandVu

The HMM toolkit that was used is called Jahmm (François, 2006), a Java HMM library that is readily available, free to use and open source. If the system is accessed with one of the offline training applications, learning the HMMs must take place before classification is possible. This is done with the GestureHMMLearner, which is able to learn an HMM for each gesture, utilizing the log-likelihood increases in each iteration of the Baum-Welch algorithm. Once the increase becomes insignificant (defined in the configuration file; 0.1% works well), or a certain number of iterations is reached (also configurable; 100 is sufficient, though is never in practice reached), the training for each HMM ceases. If the system is accessed via the online application, then the data is accessed from a socket connection to the HandVu server. The OnlineDataAdapter accesses this socket and reads a number of data points from it equal to another configurable constant, the online system window size.

Multimodal Fusion

The multimodal fusion system (see Figure 3) is designed with maximum flexibility in mind. It handles unavailable modalities, such as when the NPCs could not logically hear the player's speech input (e.g., they are in a car with the windows up), or when a command is stated but no gesture is provided (e.g., a question is asked about what is in their car trunk). The system also verifies that the information provided by the speech and gesture are consistent. These are both done with the MultimodalVerifier component. After these tasks are done, assuming both modalities are available and consistent, the gesture is used to provide further information about the spoken command in the FreeVarBinder component. Specifically, direction free variables indicated by words like "here" and "there" in the speech, in this version of the system, are "filled in" or bound to values indicated by the pointing gesture (valued as either "left" or "right").

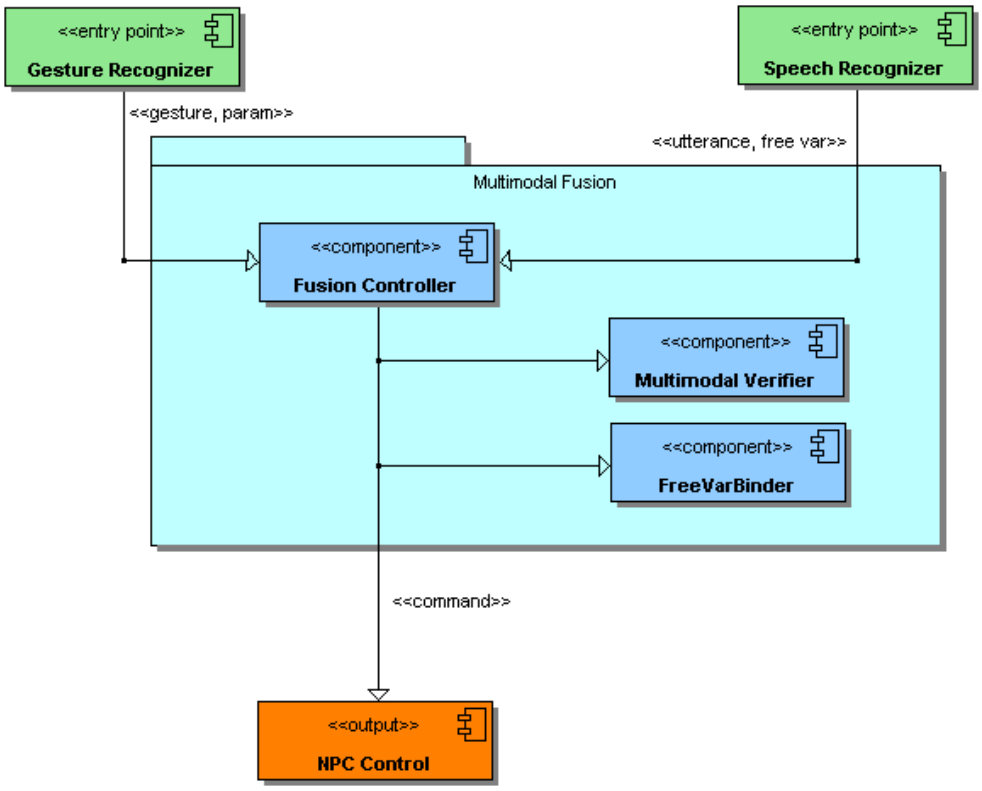

Figure 3: Architecture of the Multimodal Fusion Component in the First Prototype

System Improvements Before Evaluation

Two further changes were made to the system to increase the accuracy of classification, independent of its architecture. The first improvement was achieved by reducing the number of data points in each training sequence. This was done to improve the accuracy of the models with respect to the online system—to better match its window size—and to avoid Java's "double" primitive type from overflow and underflow that was caused by the algorithms used to train the HMMs multiplying numbers by each other for every sample point in a sequence. For low numbers, these could become as low as 10^-325, and for high numbers, as high as 10^250 or so. In some cases, the double data type could not handle the number size, so Java defaulted the value to zero. Reducing the number of samples per sequence to a configurable value (20 works well and improves the speed of training) fixed this and also avoided “non-positive definite”

covariance matrices (a concept similar to non-positive numbers), as did increasing the amount of training data. Another improvement was made by scaling the x and y coordinates by a configurable integer (five works well). This is because each state of the HMMs is represented by a multivariate Gaussian distribution, which generally ranges from about -3 to +3, rather than the data's range of -1 to +1. Scaling the coordinates by five effectively attempts to fit the coordinates to Gaussian distributions ranging from -5 to +5.

Evaluation Method

Four different experiments were performed to test the system. The first determined the ideal training and test data split for cross validation. The second evaluated the gesture recognition component alone. The third evaluated the multimodal fusion component alone. Finally, the fourth tested the online system with a realistic user. The data used for the first three experiments were recorded using HandVu on myself and a coworker doing each of the gestures. The data for the fourth experiment included another coworker who was not involved with this study, to avoid gestural bias. The first two experiments are offline training evaluations using k-fold cross validation. In the first experiment, *k* is defined as 100/*smaller_percent*, where *smaller_percent* is the smaller value of the training and testing percentages. For example, an 80/20 training/test split would result in five folds. The training data used was 10 sample sequences per gesture, and these sequences were recorded to best represent the gestures. Therefore, it should be understood that the offline evaluations are, like the prototype system itself, proofs of concept, and the online evaluations are expected to be significantly worse. Finally, for the gestures aside from "stop", the HMMs were configured to all have five states.

Evaluation Results

Experiment #1: Cross Validation Evaluation of the Gesture Recognition System with Varying Training Percentages

In this experiment, the data were split according to increasing amounts of training data versus test data and used to evaluate the gesture recognition system. When the training percent was set at each iteration, data sets representing each cross validation fold are created by a sliding window method for the smaller percent (training or test) across the data set. The results of this experiment indicate that more training data is generally better. It is also important to note that the first split that produced usable results was at 30% training data. These results were taken to mean that, with 10 data sequences per gesture, a 90/10 split would best evaluate the system's potential, effectively resulting in leave-one-out cross validation (LOOCV).

Experiment #2: Cross Validation for Each Gesture Type

For this follow up to Experiment #1, the training data were split by their gesture category. Then, as in experiment #1, a 90/10 LOOCV split was used to evaluate the gesture recognition accuracy for each gesture group. The accuracy results are shown in a bar chart, one bar for each of *stop*, *come forward*, *point left*, *point right*, *wave left*, and *wave right* (see Figure 4).

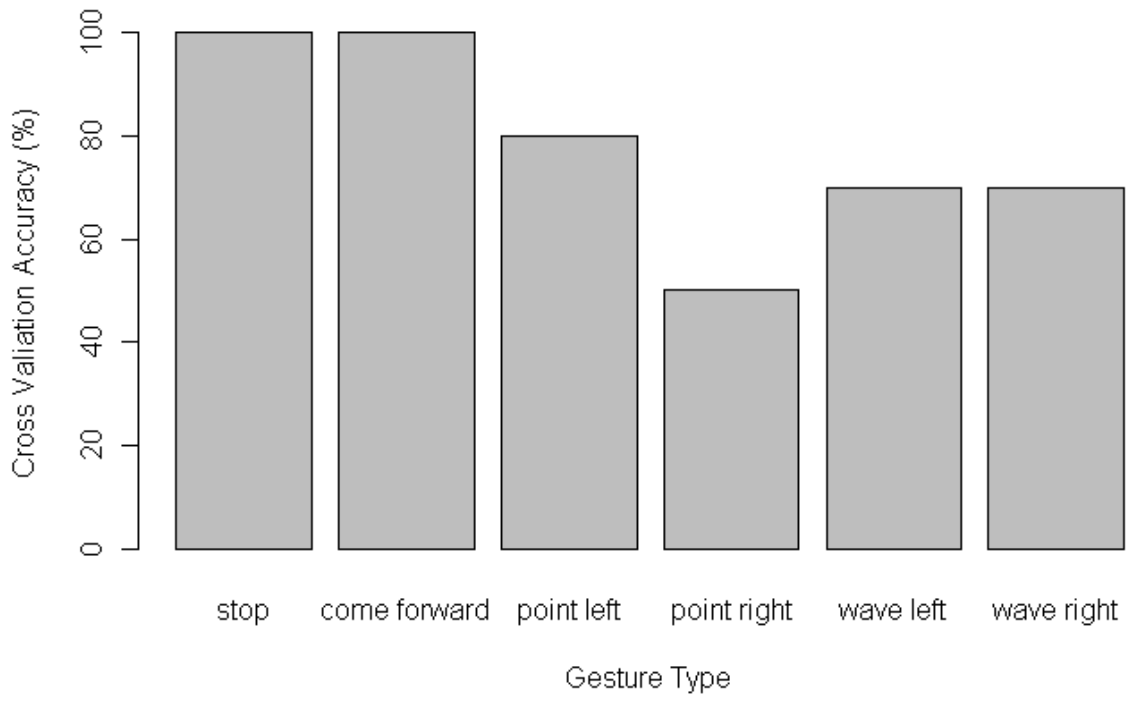

Figure 4: First Prototype Experiment #2 Results

The results pictured in Figure 4 indicate that for *stop* and *come forward*, gestures that do not require moving the hand to the side (non-lateral gestures), the performance is exceptional. For the other gestures, mediocre performance was observed, with an interesting example of poor performance for *point right* for some reason. This is probably explainable by oddities in the training data, such as perhaps the way the subjects—again, myself and a coworker—did the *point right* gesture differently than the *point left*.

Experiment #3: Multimodal Fusion Cross Validation

For this test, the multimodal fusion component's classification of the final commands was used to evaluate the system. Specifically, when a gesture was classified as null—that is, when it was determined to be not classifiable—an associated parsed speech unit was used to aid in

classification. When a gesture is misclassified, however, and contradicts the speech data, the command is classified as null, and is therefore counted as incorrect. The speech data for this experiment was generated from the correct gesture label by a gesture-to-speech-verb mapping defined *a priori*. This therefore is assuming that the speech recognition system is 100% accurate, which is not realistic, but is nevertheless reasonable in this context, given the commands are simple. The overall best performance from Experiment #1 was about 78% accuracy, considering only gesture training data. For this experiment, the accuracy increased to about 93%, a significant increase. This shows, unsurprisingly, that it is theoretically feasible to expect that fusing speech and gesture modalities can improve a system like VRP.

Experiment #4: Online Gesture Recognition System Realistic Performance Evaluation

In the final experiment, the goal was to test the gesture recognition system in a somewhat realistic environment. Only the gesture recognition system was tested because the speech recognition system that the VRP project intends to use and the respective parser component were not yet available for use. Also, it is only somewhat realistic because, though the user, a coworker of mine, had never been exposed to HandVu or this gesture classification system, it was carried out in a usability lab with some directions from me that would not normally be available in real life. The usability lab was chosen because it is an almost entirely blank room with white walls and no exposure to external or uneven lighting. This was intended to avoid issues related to lighting complications, as HandVu seems to use RGB (red-green-blue) color values to track the user's hand. The directions to the participant were to make the six gesture five times each, manually resetting HandVu's recognition system in between each gesture. If this was not done, it is plausible that HandVu would not track a single hand movement during the entire experiment and the results would be meaningless. For each gesture type iteration, a zero was recorded if the

printed classification was incorrect, and one if it was correct. This was not entirely unreasonable because the window size was set to 35 frames, and since the laptop being used could only run HandVu at five to seven frames per second, this meant a gesture was classified about every five to seven seconds. Finally, because it was decided that eliminating false positives was more important than missing true positives, the Verterbi classification threshold was set to 20, four times as much as that of the offline system. This removed false positives, since a classification was attempted at every time window, even though there usually were not any gestures taking place.

The overall accuracy was then calculated to be the total score divided by 30 total gesture iterations. For this experiment, the result was six gesture iterations correct out of 30, or 20%. This contrasts sharply with the offline cross validation results.

Summary

The gesture recognition subsystem was observed to perform very well. In the offline evaluations, it had good performance with little training data. In particular, the "stop" and "come forward" gestures were classified with 100% accuracy, indicating that they could be differentiated from the others with high reliability. For "stop," this is probably because its classification was based on the detected hand posture instead of the (x, y) coordinate time series, and the test data was collected so that this posture was recognized by HandVu. Though this turned out to be unrealistic for HandVu, identifying postures with other algorithms with very high accuracy is not altogether unrealistic. Also, the high accuracy for "come forward" can be explained by its lack of lateral motion, unlike the other four gestures. Again, in the training and test data, this gesture was performed more accurately than it may be in reality.

In general, all of the training and test data was collected in somewhat idealized

conditions. Aside from the ways listed above, the sequences were recorded without allowing HandVu to lose track of the person's hand. If HandVu did–and it did quite often—then that gesture sequence was deleted and recorded again. Regardless of whether the training and test data were unrealistic, the gestures were inherently simple, as well. The simplicity of the gestures made it easy to train small but effective HMMs to classify them. Finally, adding speech recognized in ideal conditions (ideal in the sense that the generated speech data was 100% accurate) aided in command recognition significantly. This makes sense because the guaranteed-correct speech input replaced the gesture input only when it was incorrect; therefore, the performance could only increase.

However, especially in the online system, many things did not work optimally. Given the poor performance of 20%, some speculation can be made as to what needs to be fixed to make a system like this actually usable. It was clear in the course of the last experiment that the primary limitation of the gesture recognition system was the HandVu hand tracker. HandVu had two primary weaknesses that degraded the system's performance. First, the features reported by HandVu, the (x, y) coordinates and the posture if and only if it was "closed", were very limiting. Wave and pointing gestures were commonly mistaken, both in the offline and online systems, probably because the 2d movements are similar and, without accurate posture detection, a point is the same thing as a wave. As can be seen in the confusion matrix in Table 1, this phenomenon occurs more frequently than any other misclassification. Therefore, fixing this would significantly improve its performance.

Table 1: Confusion Matrix of the Cross Validation in Experiment #1

Gesture \ label	STOP	POINT_LEFT	POINT_RIGHT	COME_FORWARD	WAVE_LEFT	WAVE_RIGHT	NULL
STOP	10	0	0	0	0	0	0
POINT_LEFT	0	8	0	0	2	0	0
POINT_RIGHT	0	0	5	1	1	2	1
COME_FORWARD	0	0	0	10	0	0	0
WAVE_LEFT	1	1	0	0	7	0	1
WAVE_RIGHT	0	0	2	0	1	7	0

The second problem with HandVu is its tendency to lose track of the user's hand very easily. Because the tracking fundamentally failed (in the online system, in particular), the results of Experiment #4 were less meaningful than they could have otherwise been. If hand tracking had been robust, the results of Experiment #4 would have indicated performance of the gesture and command classification systems instead of the failure of hand tracking. Also, because of the failed tracking, hand posture reports were not reliable. Therefore, the "stop" gestures were not even able to be classified like they were in the offline evaluations.

Some of these problems may be explained by the fact that HandVu was made to work from a camera mounted either on the ceiling or the users' head looking down at their hands. This means there would be less noise in the background, fewer changes in colors as the hand moved around, and fewer problems with lighting and shadows.

Finally, those problems that cannot be explained by the weaknesses in the hand tracking may be explained by the windowing approach to the online system. However, these problems are probably minor in comparison.

Section 5: Possible System Improvements

It was found that the first prototype system could be improved in three basic ways. First, further work could have been done to improve the recognition system as a whole. Second,

improvements could have been made to the Jahmm HMM toolkit. Finally, improvements could have been made to HandVu, or another hand tracking system could be created or used in its place.

The Gesture Command Recognition Algorithms

To improve the first prototype, more training data could be collected, parameters could be further optimized, the features could be transformed, the cascaded classifier could be adjusted, and the multimodal fusion could be more complex and powerful.

More training data would improve the HMMs used for the online system because of the idealized circumstances of the existing training data. Also, it may be helpful to have another gesture type and HMM that represents "no movement," since its likelihood would be higher than the others when no gestures are occurring. This would probably be more effective than using the existing "null" gesture (i.e., no gesture classified) because the thresholding proved ineffective, and because with HMM training there is no way to utilize negative examples.

Parameters could be further optimized because, for this project, parameter values were found that provided sufficient system performance so I could move on to later parts of the project. For a deployed system, the number of HMM states per gesture, the online system window size, the Gaussian scale factor, and the training sequence sizes all could be optimized to improve performance of the system. A methodology of optimizing dependent parameters, however, would have to be reduced to heuristics and guess work, as the whole complex optimization problem is likely intractable.

The features could be transformed to be change in x and change in y values, rather than using x and y values. However, this has the potential to confuse pointing and waving gestures further. Therefore, along with this change, better posture recognition or 3d coordinates (see

below) should be used. If one of these changes were made, then the classifier itself could be improved by checking for "stop" gestures last instead of first (as it is now). This is because the user may have a "closed" posture while doing other gestures (e.g., waving) and these should not be classified as "stop" (as happened for one of the *wave_left* examples in the above confusion matrix in Table 1).

The multimodal fusion component could be significantly improved to use more advanced fusion methods and to take advantage of contextual information. The primary potential fusion method that is not used but could be is using both modalities to help each other classify their input, either through re-ranking or bootstrapping. Speech recognition could be used to re-rank the gesture outputs in a way more intelligent than just using the speech input when the gesture input is unavailable or unrecognizable. If access were granted to the speech recognizer's rankings, rather than its single best guess, then this would be possible. Given this information and access to the language and phoneme models of the speech recognition, it would also be possible to have the two modalities actually bootstrap off each other, by taking turns labeling the data and training from it. Regardless of whether this information were provided, it would still be plausible to use contextual information to better classify the commands, given the speech and gesture outputs. This contextual information could be from the VBS2 game engine, such as when the NPC is in a car and therefore cannot hear what the user is saying, or it could be from past commands from the user or responses from the NPC, such as when a follow-up question is asked without re-specifying all of the information needed for the system to understand it.

The Jahmm Toolkit

The Jahmm toolkit is a very well-implemented system with very few errors as it stands. However, it does not support mixtures of multivariate Gaussians, and it also uses full

probabilities in some of its calculations. Though a mixture of multivariate Gaussians density function is not available, the results with the normal multivariate Gaussian were still quite good. However, the results with a mixture of Gaussians would probably be better for this data set. Implementing this would be relatively trivial, especially since a monovariate mixture of Gaussians is provided with the toolkit. On the other hand, it may be of value to implement some code to link the Jahmm library with a statistical library that has implementations for more distributions than are available. Changing the internal Jahmm code to use log-probabilities would also be helpful, as it would help avoid over/underflow problems like those that occurred in this project. This would also possibly reduce the need to artificially shorten the data sequences for training, possibly better representing real data from the hand tracking system.

Hand Tracking

The core problem with the system, the hand tracking, could be vastly improved in a number of ways. Adding another camera for stereo vision would allow for 3d coordinates, and it may be possible to improve or replace HandVu with another hand tracking system to:

- Better recognize static postures,
- Better track the movement of the hands with either better algorithms, colored gloves, or non-RGB data, or
- Track more parts of the body to improve the quality of the potential features and the tracking overall.

Some of these algorithms and ideas are already implemented in an updated version of the HandVu code base. This library has not been used thus far, however, because it is difficult to get its front-end application working on a Windows machine without Visual Studio. Also, it was unclear whether it would have been as convenient and decoupled an interface as HandVu's telnet

server. Therefore, if it were to be used in the future, it would likely be as a library for a customized version of HandVu or an entirely new hand tracking program.

The static hand posture recognition in HandVu could be vastly improved upon, especially to distinguish pointing postures from all others. This would reduce or eliminate the system's confusion between wave movements and pointing movements, in addition to creating the possibility of new and more complex gestures being recognized, such as cultural ones. The first way to do this is to define new posture masks—bit rasters that directly represent a hand posture—so the system recognizes new ones, or old ones from different angles. The utility of this is limited, however, in respect to the inherent limitation of direct image-matching recognition: the inability to handle affine transformations. Another way to do this would be to use the face detection algorithm of Viola and Jones (2002). This algorithm has the potential to recognize features that distinguish hand postures with high accuracy in real time. Also potentially useful may be the parametric gesture recognition outlined by Wilson and Bobick (1999), since the direction the finger is pointing in a pointing gesture could be framed as a parameter of a HMM.

Improving the actual hand tracking itself would make HandVu more efficacious. The first step to this is configuring what HandVu calls a "conductor," which controls how it tracks the user's hand. The limitation of this is that it is only within the algorithms that are already implemented in HandVu, none of which work well for this project's purposes. A simple solution to this problem would be to implement a trivial hand tracker that searches video for a single color, and to have the user wear gloves of that color. This would be useful to demonstrate the power of the gesture recognition system and the multimodal fusion. However, this would not be ideal for a deployed system because it requires users to wear special gloves to use the system, and it requires that the scene not have anything else of that color. More research would be

required to determine if there exist other usable hand tracking algorithms. Some progress could be made by experimenting with using non-RGB video data, so as to avoid occlusions and shadows. One example is hue-saturation-intensity (HSI) or hue-saturation-brightness (HSB), where the brightness value can be ignored and the edges can still be retained. A similar representation is called YIQ (luma in-phase quadrature), representing the computer's vision as the brightness and a "coordinate" in a defined 2d color spectrum.

Failing discovery of a fast and robust hand tracking algorithm, another option is to explore arm or full body tracking. Two examples of research that may be helpful in this area are Maes et al. (1997) and Demirdjian et al. (2005). It is unclear how robust and fast these systems are. However, if they are used, having full body information would enable the extraction of features on arm position, elbow joint angle, shoulder joint angles, and so on, for both arms. More research is therefore required if this system or something like it is to be practical.

A more practical solution might be to try a commercial body tracking system. These systems are relatively new, but have shown significant success. The perhaps most popular such system is a 3d camera created by a company named 3dv, recently purchased by Microsoft and rolled into what they are now calling Project Natal, a hands-free controller system for the XBox 360. Others like this system that are commercially available include the Swiss Ranger by Mesa Imaging in Zurich, Switzerland, and the OptriCam by Optrima 3d Imaging of Brussels, Belgium. Another competing approach is to use an array of simple cameras to take shots at different angles and track a person's body that way. This is the approach taken by the company Organic Motion, and is currently being used to model multiple people at once in realistic scenes and semi-live training simulations.

Summary

While it is true that HMM parameters can be further tweaked, and some oddities of the Jahmm framework could be dealt with, the primary problem with the first prototype system seems to be in the hand tracking technology. If HandVu were to be improved or replaced by a more accurate hand or body tracking system, there is real potential for a multimodal virtual training system.

Section 6: The Final VRP System

After the previous proof-of-concept system showed the feasibility of gesture recognition, a more accurate body tracking system was acquired. Using this system, a full customer-ready prototype system called Virtual Role Players was created to be demonstrated at a conference, for which I was the chief developer of the gesture recognition subsystem. This prototype is a set of Java applications that communicate with each other and with Virtual Battlespace 2 (VBS2), the current Army standard virtual training engine.

It was decided to use the 3d time-of-flight (TOF) OptriCam by Optrima 3d Imaging, coupled with a body tracking software system called Iisu, made by a company called Softkinetic. Furthermore, for the sake of displaying the prototype at a conference, some shortcuts were taken with the gesture recognition itself. Instead of continuing the HMM-based dynamic recognition, simple, illustrative gestures were programmed statically with the information output from the Iisu body tracker: stop, come forward, and pointing. Finally, some simple multimodal fusion was done to illustrate the power of combining speech and gesture recognition—namely, pointing while talking.

System Details

The VRP system is based around a training scenario in VBS2 that simulates a military checkpoint in Iraq (see Figure 5). The user is a military officer operating the checkpoint, and, as such, his job is to: tell cars to move forward; tell them to stop, if desired; talk to the driver, or ask the driver and/or passengers to get out of the vehicle; talk to any of the passengers, with cultural preference to the male driver and head of the household, asking them any relevant questions; search them, if desired; tell them to move to various locations in the scenario; and, if everything checks out, to inform them that they may proceed. To support these actions, the system automatically recognizes speech, hand gestures and body position, with various other actions available via button presses on a Wiimote. The non-player characters (NPCs), or virtual role players, in the game represent the civilian drivers and passengers of the vehicles, and they respond to the user's speech and gestures as a normal person would at a checkpoint.

Figure 5: The Checkpoint Scenario in VBS2

Overall Architecture

The entire VRP system (see Figure 6) has two basic parts relevant to this thesis, with some supporting components that will not be discussed here. The first main part is the gesture recognition subsystem. More details will be discussed in the following subsection, but, in this diagram, it is important to note that its input is video of the user's body from the 3d camera and the output is a series of gesture events dispatched to the VRP core system. The second main part is the VRP core, wherein most of the logic for the checkpoint scenario resides. In the context of this thesis, it is important to note that the VRP core takes input from the gesture recognition and the speech recognition systems and, based on these inputs, interacts with the VBS2 application through its VBS2 plugin component.

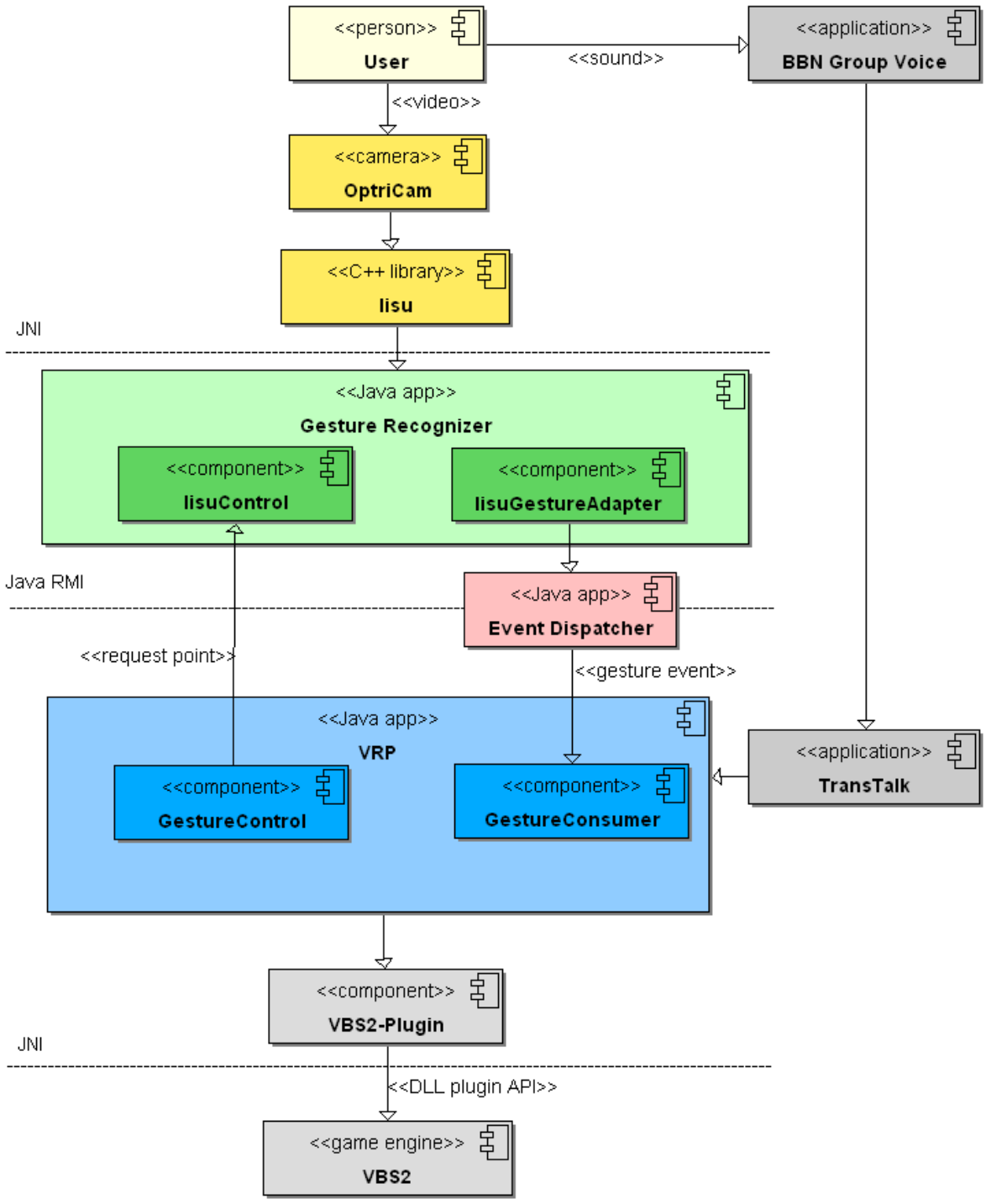

Figure 6: The Architecture of the VRP System

Because the full VRP system was created from a collection of distinct technologies and was also designed to be potentially distributed, the separate applications communicate in a

variety of ways. First, the Gesture Recognizer pulls data from the Iisu body tracking system using the Java Native Interface (JNI), which allows Java applications to communicate with native applications without any operating system-specific interface. Similarly, the VRP core communicates with the VBS2 game engine with JNI via VBS2's DLL-based plugin interface. Later, the Gesture Recognizer communicates with the VRP core application with the Java Remote Method Invocation (RMI) framework, allowing both of them to be on different machines, if necessary. These two communication protocols are described in more detail below.

The parallel path of processed input starts at the BBN Group Voice (BGV) voice-over-IP (VoIP) system, which transmits what the users are saying when they speak a command into a headset. From here, the audio is sent to the TransTalk speech recognition system. TransTalk is a system created by BBN as part of the TransTac DARPA program. Though the VRP project only uses the speech recognition part, TransTalk's full purpose is to recognize speech, translate it into the target language—Arabic or English, the opposite of the original language—and then synthesize speech in the new language.

Gesture Recognizer

As shown in Figure 6, the gesture recognizer takes input from the Iisu body tracking system and outputs GestureEvents to the VRP system's GestureConsumer. Figure 7 illustrates more details, which are discussed below by division into the input, the recognition, and the output.

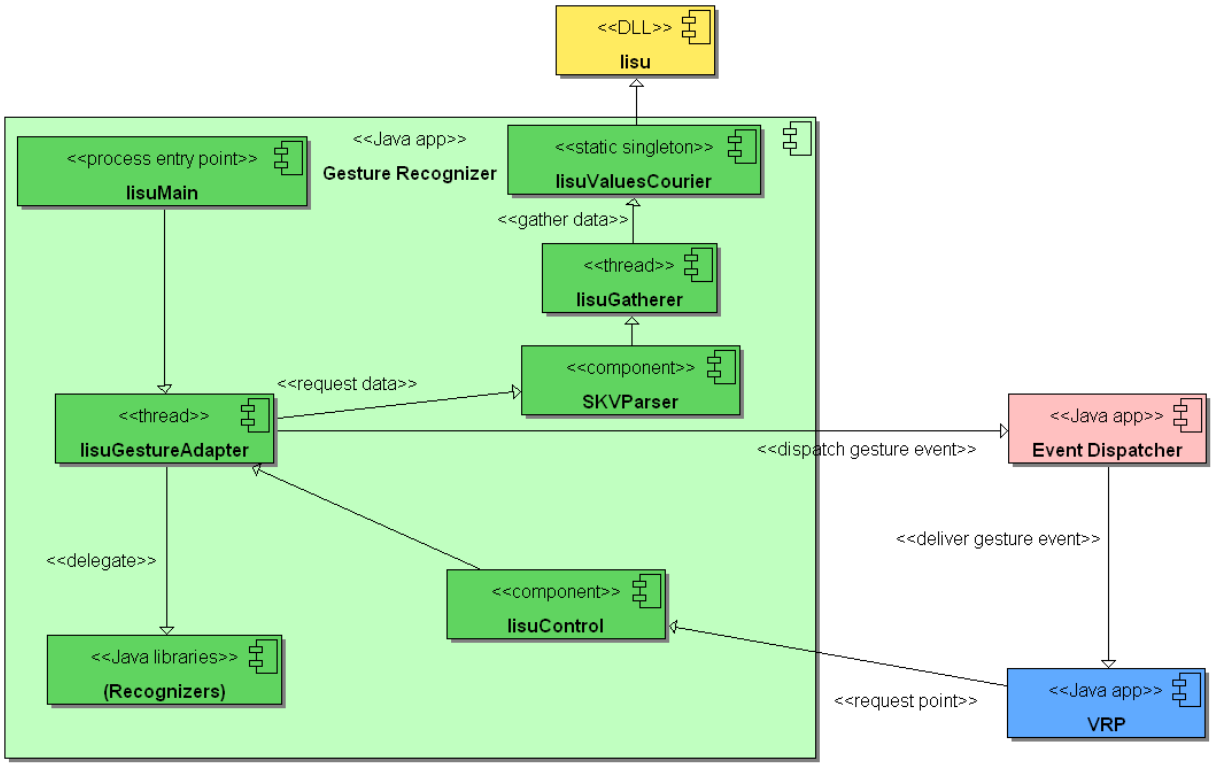

Figure 7: The Gesture Recognizer Subsystem in VRP

Input

Figure 7 shows that the logical flow of the gesture recognition program starts from an independent launch point, IisuMain, and pulls data from the Iisu dynamic link library (DLL). The data about various parts of the user's body from Iisu comes in proprietary formats, but the IisuValuesCourier converts it into standard formats for use in the rest of the system, including vector classes from the Java3d library. Some example data points are the *x*, *y*, and *z* coordinates of the location of the left and right hands, the location of the head and both shoulders, and the vector representing the leaning of the user. This information is read on an "as-fast-as-possible" basis by the IisuGatherer thread, which sends requested data to the SKVParser component, called that because the Iisu video format is SKV: SoftKinetic video. It has two methods: one to get a

single record of data, and one to get a set of data across a time frame to be averaged and possibly filtered for bogus data.

Recognition

The input data is requested when needed by the IisuGestureAdapter thread, which constantly loops over all of the recognizer plugins and sends them the data they need to recognize gestures. The output of each of the recognizers is collected and then filtered for such things as conflicts between similar gestures and their respective priorities in being recognized.

In the current version of VRP, there are three primary recognizers: the avatar control recognizer, the communication gesture recognizer, and the point recognizer. The avatar control recognizer is built to recognize positions and movements that are relevant to the actual control of the video game avatar. Positions and movements currently include stepping a variable distance away from ones "center point" for locomotion and leaning ones torso to the side to have the avatar peek around a corner. The communication gesture recognizer is programmed to recognize the hand gestures that the user may make to the NPCs in the game to communicate with them. These gestures currently include holding the palm out to communicate "stop" and moving ones hand toward oneself in a cyclical nature to indicate "come here." Finally, the point recognizer determines whether or not the user is pointing at the projected screen and, if so, where they are pointing relative to the virtual world.

Output

As Figure 7 indicates, there are two ways the gesture recognizer outputs information to the VRP core. The first method uses the observer design pattern where event listeners register themselves with the Event Dispatcher application. Then, when the Gesture Recognizer dispatches a gesture event, the Event Dispatcher pushes the events to the relevant listener, which

in this case is the VRP core application, through its GestureConsumer component. The second method is in the opposite direction; the information is requested and sent back to the requester, rather than being pushed to it asynchronously as with the first method. The information passed in this way is from when the user does pointing gestures. It is requested by the VRP core when this information is of interest to it—namely, when the user issues a command that contains key words such as *there* or *here*. When this gesture information is requested by the VRP core, that system takes the screen coordinates and maps them to a position in the virtual world to realize the object or location of the user's pointing.

Informal Conference Evaluation

The VRP prototype system has been evaluated informally at a conference. It has not been evaluated formally, however, though such possible future evaluations are outlined as future work in the next section. BBN's light-weight immersion training (LWIT) group attended the conference to demonstrate the work. We had four demonstrations: an area for BBN's Speech Group to demonstrate their recent work, a Navy training game for new recruits, a sensor visualization system for Second Life, and the VRP system, renamed for marketing purposes as 4Real—named after the four ways our system was realistic. While demonstrating the system, there was usually a person to talk about it, while another person used the system, making gestures to cars at the checkpoint and interviewing their passengers. The informal evaluation results of the VRP system can be split into three categories: general results, those related to the speech recognition and those related to the gesture recognition.

General Results

The first thing noticed was that people watching us demonstrate the system and those who tried it themselves enjoyed it very much. The second thing was that those who did try it

themselves sometimes asked what the commands were. Once we explained the checkpoint scenario, the free-form speech recognition, the supported hand gestures, and the basis of locomotion to them, they were easily able to use the system.

Speech Recognition Results

Overall, the speech recognizer had very high accuracy and very rarely misunderstood checkpoint-specific speech, although this did sometimes happen. More often, the problem was that, though the speech recognizer understood what was said, the actual training scenario NPCs were not programmed to understand it. The VRP system was able to handle most synonyms and other differences in speech, but not everything could be accounted for. Other impacts on the speech recognition include ambient noise around our exhibit and the users sometimes forgetting that they had to push the button on the Wiimote to speak to the NPCs.

Gesture Recognition Results

The gesture recognition also had a high level of accuracy. However, errors did occur, and when they did, they could be traced to some common causes. Lighting was mostly not a problem, largely because it was carefully controlled, but in other venues it can be a significant concern. More problematic was the effect of the crowd on the Iisu human body modeling software. Configuring its feature to limit the location on the floor where the user can stand proved critical. Even if the crowd did not cause noise to show up in the body tracking, there was also a problem with communicating with the NPCs versus communicating with real people. Given this was at a trade show, there was a significant amount of talking and gesturing to people to tell them about the VRP system, so this was sometimes a problem for people using the system.

Some other impacts on the gesture recognition were environmental. One was the color of people's clothes. We did notice that some people who wore dark clothes saw some problems with

gesture recognition, probably because the dark colors absorbed more infrared light that the 3d camera uses for scene recognition. To combat this same effect from objects in the background on the demonstration floor, we put up a white back-screen, though we do not know whether or not it was necessary, or whether the software bounding box was sufficient. Another environmental factor was infrared light from other booths affecting the 3d camera used for gesture recognition.

Other issues resulted from the definitions of the various gestures. For the hand gestures mostly, we found that people perform the gestures differently. A lot of variance was encountered with the *come forward* gesture; there are a lot of ways to move ones hand to indicate you want someone to move in your direction. As a result, the illustrative static gesture recognizer had difficulty accounting for all of them.

Another gesture definition problem was the body gesture used for avatar locomotion. Though leaning to move is relatively easy, it also was found to be confusing when the users wanted to have their avatar lean. So it may be that leaning is not the ideal way to model avatar movement gestures. We have since switched the locomotion gesture to having the users stepping away from their "home" or center positions on the ground, and their avatar will move in that direction. If they step out even farther, their avatars will run instead of walk. Their avatars will stop moving when they step back into their center positions. Leaning left and right are now mapped to the avatars itself leaning left and right, as if they were looking around the corner.

Summary

Significant promise of gesture recognition was shown with the first prototype, so a second, full gesture and speech mulitimodal fusion prototype virtual training system was created to display the potential of this approach to interaction. It was demonstrated at a conference. While demonstrating it, we found it to hold a lot of promise with respect to recognition accuracy,

as long as the environment is somewhat controlled. Some measures can be taken to ensure this, including correct lighting, a noise-cancelling head set, and limiting crowding around the user. It is also important to account for the differences in how people speak and make gestures and to make sure any of the gestures supported by the system do not conflict with the natural impulses of the users.

Section 7: Discussion

Conclusions

There is significant promise for the creation of multimodal virtual training systems. Both the first and the second prototype showed great potential for gesture recognition accuracy, though they both had limitations.

The first system illustrated the theoretical power of hidden Markov model-driven gesture recognition and multimodal fusion. However, the hand tracking system, though free and open source, and though it worked fine for theoretical evaluations, proved to be insufficient for a robust gesture-based system.

The second prototype system illustrated the full potential of deployed gesture-based systems. It had far-superior hand and body tracking and, therefore, removed that as a constraint to realize gesture recognition's full potential. This system's limitations were of the more practical sort, in that, while it performed admirably at the conference, much care was taken to ensure environmental factors had limited effect on it.

In conclusion, it seems that a mixture of dynamic gesture recognition for complicated gestures (e.g., HMMs for hand waves) and static recognition for static gestures (e.g., finite state machines or their equivalent for the stop gesture) optimize both the theoretical and practical

gesture recognition. Similarly, environmental control improves both the probability that the gesture recognition will be near its theoretical ideal and that the users of the system will be able to use the system effectively. With these precautions taken, systems like VRP could very easily incorporate multimodal input to improve their effects on human factors.

Summary of Contributions

This work shows how multimodal input can address human factors problems in virtual training systems. Human factors issues relevant to virtual training systems were reviewed, and the argument was made that multimodal input can increase a system's immersion, trainee performance, ease of use, and transfer of learning to the real world. A detailed description of the process of gesture recognition was presented, along with its potential for use in different types of information systems, including virtual training systems.

A proof-of-concept system to show the potential of gesture recognition and multimodal fusion was created. Then this system was evaluated based on its gesture recognition accuracy under ideal conditions (using cross validation) and then under more realistic conditions. The first prototype showed the importance of computer vision body tracking for systems that use gesture input.

Based on these insights and better body tracking software, a second prototype was created. This prototype supports speech input, gesture input and virtual role-playing NPCs that can interpret these inputs in a military checkpoint scenario. The author of this thesis contributed the design and implementation of the gesture recognizer subsystem. VRP, aka 4Real, was then shown at a conference. Some informal results were outlined that include very good recognition accuracy, but with some caveats that the environment must be conducive for such a system.

Open Problems and Future work

Though significant work has been done to show the potential alleviation of human factors problems in virtual training systems, some work remains to be completed. There is more work that can be performed on the system itself to improve recognition accuracies and to more formally show the usefulness of the multimodal approach to training.

The first step to improve the system is to build in recognition of more numerous and more complicated gestures, and more accurately. While "stop," "come forward," and pointing are illustrative gestures for the purpose of demonstrating to customers, there are other gestures a checkpoint guard must sometimes use, including waving the car past without an inspection if they are an authorized vehicle, in addition to cultural gestures that can improve or damage the user's virtual relationship with the civilians. To support some of the more complicated gestures, and to make gesture recognition more accurate, a more dynamic approach to recognition might be attempted. Like the first prototype system utilized, hidden Markov models might be a useful way to accurately recognize gestures. It also might be a good way to more easily support new gestures without hard-coding any of the recognition, but instead relying on training videos for the system.

The next way to improve VRP is to implement more multimodal fusion. Though speaking-and-pointing multimodal fusion is already present, there are a lot of other ways to utilize the multiple modalities. The first of these might be multimodal verification. That is, using the speech, the gestures a user makes could be more accurately identified, or, conversely, using a identified gesture, the speech could be more accurately recognized. Taking this concept a bit further, the two modalities could be recognized in parallel using an iterative bootstrapping algorithm. Another way to fuse the modalities is using one modality when the other is

obstructed, such as when an NPC cannot "hear" or "see" the user's speech or gesture, respectively. A third way would be to require both modalities for the purposes of training. For example, if the user said "come here" but did not motion, then the system might act as if they did not say anything at all. The purpose of this might be to enforce good communication habits in the trainee.

Aside from improving the system itself, there are also multiple ways to do a more formal evaluation of the VRP system. This is important from an academic perspective because otherwise there is no way to know for sure what approaches encompassed by the VRP prototype work, and which ones do not. The ways to do this range from a theoretical evaluation of the entire system using empirical data about atomic actions, to a full empirical evaluation of the system with respect to the human factors constructs identified earlier in the thesis. Another formal evaluation that could be done for the sake of the military training domain would be determining how the multimodal input contributes to specific training objectives.

More work could also be done to study the theoretical use of gestures. Instead of focusing research on the VRP system alone, it is also possible to isolate modalities and tasks to determine the best fit between them. Another idea would be to try multimodal inputs for domains other than virtual training.

Comparing Task Performance

Before these evaluations are performed, a full task analysis should be performed on the checkpoint scenario training domain when using multimodal input, and possibly also when not using multimodal input. These analyses could be specified to accord with theoretical goals of the system, or they could be obtained with verbal protocol analysis of actual or representative users using the system. No matter how they are done, they might take the form of some type of GOMS

(goals, operators, methods, and selection rules) analysis (Card, Moran, and Newell, 1983), which has been shown to work well for traditional computer interaction, or it might be similar to recent work done for reality-based interaction, called the Cognitive Description and Evaluation of Interaction (CoDeIn) task and evaluation framework (Christou, 2007). It might also be interesting to compare these frameworks using the same version of VRP, and perhaps also comparing traditional and multimodal input versions of VRP under both the same and different task frameworks.

Once a task ontology of some sort is established, the next step is to specify times for each atomic action contained therein. Once these are specified, it is a simple matter of adding up these atomic times within various task flows that users might partake to come up with full activity times. An alternative to evaluating theoretical execution times is to use a cognitive model of some sort to simulate these actions (Teo and John, 2008). The benefit of this approach is that variations in performance and improvements with task execution over time can also be modeled. If a cognitive model of some sort is used, metrics other than time could also be evaluated, including fatigue and the usage and gaining of knowledge and skills due to training, especially if the knowledge-based framework of CoDeIn is used instead of GOMS.

User Studies

Although these time comparisons would be informational, a more thorough user study of the entire system would provide more specific insights, including whether it actually addresses and improves upon each of the human factors issues outlined above: immersion, performance, ease of use, and transfer of learning. From an academic perspective, it also might be interesting to evaluate other human factors issues to see if multimodal input helps with them, as well. For

example, whether or not it reduces users' cognitive load or increases their situation awareness, such as in logistics or command-and-control (C2) systems, is a potential future study.

Multimodal Design Methodology

Finally, further work could be done to identify methodological hints or best practices for developers of other multimodal systems. One such area that would be very useful would be which modality is best for various different types of tasks. For example, this thesis has indicated the potential for multimodal input to be used for virtual environment systems, but other types of systems should be more thoroughly explored and the findings organized. If other system designers knew about such a set of guidelines, better multimodal systems could be created in the future, in domains as of yet untouched by multimodal input.

References

Anderson, J. R. (1996). ACT: A simple theory of complex cognition. *American Psychologist*, *51*, 355-365.

Anthony, M., Chandler, T., & Heiser, E. (2009). A Framework for Promoting High Cognitive Performance in Instructional Games. Presented at the Interservice/Industry Training, Simulation and Education Conference (I/ITSEC), 2009, 77-85.

Avidan, S. (2004). Support vector tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(8), 1064-1072.

Bobick, A. F., & Wilson, A. D. (1995). A state-based technique for the summarization and recognition of gesture. In *Proceedings of the International Conference on Computer Vision* (pp. 382-388).

Bolt, R. A. (1980). "Put-that-there": Voice and gesture at the graphics interface. In *Proceedings of the 7th annual conference on Computer graphics and interactive techniques* (pp. 262-270). ACM.

Bregler, C., & Malik, J. (1998). Tracking people with twists and exponential maps. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 8-15). Institute of Electrical Engineers Inc (IEEE).

Caird, J. K. (1996). Persistent issues in the application of virtual environment systems to training. In *Proceedings of the 3rd Symposium on Human Interaction with Complex Systems (HICS'96)* (p. 124). IEEE Computer Society.

Campbell, L. W., Becker, D. A., Azarbayejani, A., Bobick, A. F., & Pentland, A. (1996). Invariant features for 3-D gesture recognition. In *Automatic Face and Gesture Recognition, 1996., Proceedings of the Second International Conference on* (pp. 157-162).

Card, S. K., Moran, T. P., & Newell, A. (1983). *The psychology of human-computer interaction*. L. Erlbaum Associates Inc. Hillsdale, NJ, USA.

Cavazza, M., Charles, F., Mead, S. J., Martin, O., Marichal, X., & Nandi, A. (2004). Multimodal acting in mixed reality interactive storytelling. *IEEE Multimedia*, 30-39.

Christou, G. (2007). Towards a new method for the evaluation of reality based interaction. In *CHI'07 extended abstracts on Human factors in computing systems* (p. 2165-2170). ACM.

Corradini, A. (2001). Dynamic time warping for off-line recognition of a small gesture vocabulary. In *IEEE ICCV*

Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems (pp. 82–89).

Couvillion, W., Lopez, R., & Ling, J. (2001). The pressure mat: a new device for traversing virtual environments using natural motion. In *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC)*, 2001, 199-211. NTSA.

Cui, Y., & Weng, J. J. (1996). Hand segmentation using learning-based prediction and verification for hand sign recognition. In *IEEE Computer Society Conference on Computer Vision And Pattern Recognition*, 88-93. Institute Of Electrical Engineers Inc (IEEE).

Darrell, T., Gordon, G., Harville, M., & Woodfill, J. (2000). Integrated person tracking using stereo, color, and pattern detection. *International Journal of Computer Vision*, *37*(2), 175-185.

Davis, J., & Shah, M. (1994). Visual gesture recognition. *IEE Proceedings-Vision Image and Signal Processing*, *141*(2), 101-106.

Demirdjian, D., Ko, T., & Darrell, T. (2005). Untethered gesture acquisition and recognition for virtual world manipulation. *Virtual Reality*, *8*(4), 222-230.

Everett, S. S., Tate, D. L., Maney, T., & Wauchope, K. (2000). A Speech-Controlled Interactive Virtual Environment for Ship Familiarization. In *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC)*, 2000, 172-181. NTSA.

François, J. M. (2006). Jahmm-an implementation of hmm in java, *0.6.1*. Retrieved from http://www.run.montefiore.ulg.ac.be/~francois/software/jahmm/

Frank, G. A., Helms II, R. F., & Voor, D. (2000). Determining the right mix of live, virtual, and constructive training. In *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC)*, 2000, 1268-1277. NTSA.

Gavrila, D. M. (1999). The visual analysis of human movement: A survey. *Computer Vision and Image Understanding*, *73*(1), 82-98.

Gordon, J., Casey, S., Burns, D. J., & Cohn, D. J. (2001). Addressing Realism in Determining Requirements for Simulation Based Learning Environments. In *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC)*, 2001, 673-682. NTSA.

Holden, M. K. (2005). Virtual environments for motor rehabilitation: review. *Cyberpsychology & behavior*, *8*(3),

187-211.

Hwa, R. (2000). Sample selection for statistical grammar induction. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics*, 45-52.

Jacob, R. J. K., Girouard, A., Hirshfield, L. M., Horn, M. S., Shaer, O., Solovey, E. T., & Zigelbaum, J. (2008). Reality-based interaction: a framework for post-WIMP interfaces. *Proceedings of the Twenty-sixth Annual SIGCHI Conference on Human Factors in Computing Systems*, 201-210.

Jones, P. N., & Mastaglio, T. (2006). *Evaluating the Contributions of Virtual Simulations to Combat Effectiveness*. MYMIC LLC, Portsmouth, VA.

Ju, S. X., Black, M. J., Minneman, S., & Kimber, D. (1998). Summarization of videotaped presentations: automatic analysis of motion and gesture. *IEEE Transactions on Circuits and Systems for Video Technology*, *8*(5), 686-696.

Kappé, B., & de Vries, S. C. (2000). Networked Simulators: Effects on the Perceptual Validity of Traffic in Driving Simulators. In *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC)*, 2000, 621-626. NTSA.

Kendon, A. (1986). Current issues in the study of gesture. *The Biological Foundations of Gestures: Motor and Semiotic Aspects*, 23-47.

Knerr, B. W., (2007). Immersive Simulation Training for the Dismounted Soldier. Study Report 2007-1, US Army Research Institute.

Kolsch, M., & Turk, M. (2004). Fast 2d hand tracking with flocks of features and multi-cue integration. In *CVPRW'04: Proceedings of the Conference on Computer Vision and Pattern Recognition Workshop*, 10, 158.

Laird, J. E., Newell, A., & Rosenbloom, P. S. (1987). Soar: An architecture for general intelligence.

Luperfoy, S., Domeshek, E., Holman, E., & Struck, D. (2003). An architecture for incorporating spoken dialog interaction with complex simulations. In *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC),* 2003, 1515-1523. NTSA.

Maes, P., Darrell, T., Blumberg, B., & Pentland, A. (1997). The ALIVE system: Wireless, full-body interaction with autonomous agents. *Multimedia Systems*, *5*(2), 105-112.

Martin, J., & Crowley, J. L. (1997). An appearance-based approach to gesture-recognition. *Lecture Notes in Computer Science*, *1311*, 340-342.

Mitra, S., & Acharya, T. (2007). Gesture recognition: A survey. *IEEE Transactions on Systems Man And Cybernetics Part C Applications and Reviews*, *37*(3), 311.

Moeslund, T. B., & Granum, E. (2001). A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, *81*(3), 231-268.

Newell, A., & Simon, H. A. (1976). Computer science as empirical inquiry: Symbols and search. *Communications of the ACM*, *19*(3), 113-126.

Nielsen, P., Koss, F., Taylor, G., & Jones, R. M. (2000). Communication with intelligent agents. In *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC),* 2000, 824-834. NTSA.

Oka, K., Sato, Y., & Koike, H. (2002). Real-time fingertip tracking and gesture recognition. *IEEE Computer Graphics and Applications*, 22(6), 64-71.

Oviatt, S. (1997). Mulitmodal interactive maps: Designing for human performance. *Human-Computer Interaction*, *12*(1), 93-129.

Page, E. H., & Smith, R. (1998). Introduction to military training simulation: a guide for discrete event simulationists. In *Proceedings of the 30th conference on Winter simulation*, 53-60. IEEE Computer Society Press Los Alamitos, CA, USA.

Parsons, S., & Mitchell, P. (2002). The potential of virtual reality in social skills training for people with autistic spectrum disorders. *Journal of Intellectual Disability Research*, *46*(5), 430-443.

Pavlovic, V., Rehg, J. M., Cham, T. J., & Murphy, K. P. (1999). A dynamic Bayesian network approach to figure tracking using learned dynamic models. In *Proc. Int. Conf. on Computer Vision*, 1, 94–101.

Poddar, I., Sethi, Y., Ozyildiz, E., & Sharma, R. (1998). Toward natural gesture/speech HCI: A case study of weather narration. In *Proceedings 1998 Workshop on Perceptual User Interfaces (PUI'98)*, 1-6.

Quek, F. K. H., & Zhao, M. (1996). Inductive learning in hand pose recognition. In *Proc. of IEEE Int'l Conf. on Automatic Face and Gesture Recognition*, 78–83.

Quinlan, J. R. (1993). *C4. 5: programs for machine learning*. Morgan Kaufmann.

Rose, F. D., Attree, E. A., Brooks, B. M., Parslow, D. M., Penn, P. R., & Ambihaipahan, N. (2000). Training in virtual environments: transfer to real world tasks and equivalence to real task training. *Ergonomics*, *43*(4),

494-511.

Salas, E., Rosen, M. A., Weaver, S. J., Held, J. D., & Weissmuller, J. J. (2009). Guidelines for Performance Measurement in Simulation-Based Training. *Ergonomics in Design: The Quarterly of Human Factors Applications*, *17*(4), 12-18.

Stanney, K. M., Mourant, R. R., & Kennedy, R. S. (1998). Human factors issues in virtual environments- A review of the literature. *Presence: Teleoperators and Virtual Environments*, *7*(4), 327-351.

Starner, T., Weaver, J., & Pentland, A. (1998). Real-time american sign language recognition using desk and wearable computer based video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *20*(12), 1371-1375.

Stone, R. J., & McDonagh, S. (2002). Human-Centered Development of A Helicopter Voice Marshalling Simulator. In *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC)*, 2002, 107-117. NTSA.

Stone, R. J., & Rees, J. B. M. (2002). Application of Virtual Reality to the Development of Naval Weapons Simulators. In *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC),* 2002, 129-139. NTSA.

Sun, R. (2006). The CLARION cognitive architecture: Extending cognitive modeling to social simulation. *In: Cognition and multi-agent interaction: From cognitive modeling to social simulation*, 79-99.

Teo, L., & John, B. E. (2008). Towards a tool for predicting goal-directed exploratory behavior. In *Human Factors and Ergonomics Society Annual Meeting Proceedings*, 52, 950-954.

Ullman, T. D., Baker, C. L., Macindoe, O., Evans, O., Goodman, N. D., & Tenenbaum, J. B. (to appear). Help or Hinder: Bayesian Models of Social Goal Inference. *Advances in Neural Information Processing Systems,* 22.

Viola, P., & Jones, M. (2002). Robust real-time object detection. *International Journal of Computer Vision*, *57*(2), 137-154.

Weizenbaum, J. (1966). ELIZA—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1), 36-45.

Wexelblat, A. (1995). An approach to natural gesture in virtual environments. *ACM Transactions on Computer-Human Interaction (TOCHI)*, *2*(3), 200.

Wilson, A., & Bobick, A. F. (1995). Learning visual behavior for gesture analysis. In *Proceedings IEEE International Symposium on Computer Vision*, 229-234.

Wilson, A. D., & Bobick, A. F. (1999). Parametric hidden Markov models for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(9), 884-900.

Witmer, B. G., & Singer, M. J. (1998). Measuring presence in virtual environments: A presence questionnaire. *Presence*, *7*(3), 225-240.

Wren, C. R., & Pentland, A. P. (1998). Dynamic models of human motion. In *IEEE International Conference on Automatic Face and Gesture Recognition*, 22-27.

Wu, Y., & Huang, T. S. (1999). Human hand modeling, analysis and animation in the context of HCI. *International Conference on Image Processing (ICIP)*, 1999(3), 6-10.

Yang, M. H., Ahuja, N., & Tabb, M. (2002). Extraction of 2d motion trajectories and its application to hand gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8), 1061-1074.