Toward an Intelligent Multimodal Interface for Natural Interaction

by

Ying Yin

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2010

Author ....
Department of Electrical Engineering and Computer Science
May 14, 2010

Certified by ....
Randall Davis
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by ....
Terry P. Orlando
Professor of Electrical Engineering and Computer Science
Chairman, Department Committee on Graduate Students

Toward an Intelligent Multimodal Interface for Natural Interaction

by

Ying Yin

Submitted to the Department of Electrical Engineering and Computer Science on May 14, 2010, in partial fulfillment of the requirements for the degree of Master of Science in Electrical Engineering and Computer Science

Abstract

Advances in technology are enabling novel approaches to human-computer interaction (HCI) in a wide variety of devices and settings (e.g., the Microsoft® Surface, the Nintendo® Wii, iPhone®, etc.). While many of these devices have been commercially successful, the use of multimodal interaction technology is still not well understood from a more principled system design or cognitive science perspective. The long-term goal of our research is to build an intelligent multimodal interface for natural interaction that can serve as a testbed for enabling the formulation of a more principled system design framework for multimodal HCI. This thesis focuses on the gesture input modality. Using a new hand tracking technology capable of tracking 3D hand postures in real-time, we developed a recognition system for continuous natural gestures. By nature gestures, we mean the ones encountered in spontaneous interaction, rather than a set of artificial gestures designed for the convenience of recognition. To date we have achieved 96% accuracy on isolated gesture recognition, and 74% correct rate on continuous gesture recognition with data from different users and twelve gesture classes. We are able to connect the gesture recognition system with Google Earth, enabling gestural control of a 3D map. In particular, users can do 3D tilting of the map using nontouch-based gesture which is more intuitive than touch-based ones. We also did an exploratory user study to observe natural behavior under a urban search and rescue scenario with a large tabletop display. The qualitative results from the study provides us with good starting points for understanding how users naturally gesture, and how to integrate different modalities. This thesis has set the stage for further development towards our long-term goal.

Thesis Supervisor: Randall Davis
Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

I would like to dedicate this work to my advisor, Prof. Randall Davis. Words cannot express my gratitude for all the trust, advice and learning opportunities he has provided. The comments and suggestions he gave me on this thesis are invaluable.

Our gesture recognition module depends on the hand tracking system developed by Robert Wang. He has given me tremendous support that allowed me to understand and apply his system in our research project. It has been a great learning experience to collaborate with Robert.

I would like to thank Prof. William Freeman and Prof. Berthold Horn who always promptly answered my computer vision related questions. I also want to express my gratitude to my friendly and supportive groupmates Yale Song, Andrew Correa, Chih-yu Chao and Tom Ouyang. They have given me tips and suggestions throughout the course of my study and research.

I feel fortunate to have a group of supporting friends who make my life at MIT so wonderful. Special thanks to Lei Zhang, Ermin Wei and Bonnie Lam who were ready to lend me a hand (both literally and figuratively :-)) when I needed to collect data for my research.

Thanks to Victor Costan for teaching me Ruby and Rails, and helping me set up a website for user study. His drive and passion in computer science has also made me more motivated and efficient.

Finally, I want to thank my parents, Weimin Yin and Xiaoyan Teng, for their strong and loving support. Even though they are thousands of miles away, their unwavering love is the fuel for my determination. I am deeply indebted to them for everything they taught me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

As Card et al. [5] mentioned in The Psychology of Human-Computer Interaction, our society is "transforming itself to use the power of computers throughout its entire fabric - wherever information is used - and that transformation depends critically on the quality of human-computer interaction". Computers have become ubiquitous, be it in office, at home, in commercial areas or in science laboratories. The amount of time people interact with computers has increased tremendously.

Following Moore's law, there is roughly an exponential rate of improvement in computer hardware in terms of processing speed and memory capacity in the past few decades. Meanwhile, display technology has also had significant advancement. Large displays and small pocket-size portable displays are two major trends of the development. However, the predominant mode of human computer interaction has not changed substantially since the creation of the windows, icons, menus, and pointer (WIMP) more than thirty years ago, interfaces that themselves rely on technology going back forty years and more. As interaction moves away from the confinement of the desktop, keyboards and mice become inconvenient. For example, using a mouse with a large display will be very cumbersome due to the large distance to move. A mouse is also limited to 2D interaction. Even 3D mice are no solution, as they feature so many buttons that these are not very intuitive to use (Figure 1-1[1]).

Some new interaction techniques have emerged to provide novel ways of interac-

[1] `http://www.jr.com/logitech/pe/LGI_3DX700036/`

Figure 1-1: Logitech SpacePilot PRO 3D mouse

tion (e.g., the Microsoft Surface®, the Nintendo® Wii, iPhone®), with a common aspiration of making interacting with computation easier. Our goal is to take this aspiration to the next level by developing an intelligent multimodal interface for natural interaction. By *natural interaction*, we mean the kind of cognitively transparent, effortless multimodal communication that can happen between people; we want to make this possible in human-computer interaction (HCI), so that the computer interface understands what the user is saying and doing, and the user can simply behave.

We believe that natural interaction can provide better learnability, flexibility, memorability, convenience and efficiency, but further user studies are needed to investigate this belief. This thesis work mainly focuses on enabling gesture as one of the input modalities.

1.1 Mulitmodal Interaction for USAR

We use urban search and rescue (USAR) as our motivating appliation. The task of emergency response in a USAR operation involves directing and coordinating geographically distributed teams of people working under time pressure to search buildings and aid victims in the aftermath of a disaster. Clearly this requires strategic assessment of a large volume of complex information. Access to the information by USAR teams in the command center can be facilitated by various human-computer interfaces. However, these interfaces are currently hard to use, require a lot of training, and often impede teamwork [33]. Intelligent, natural, and multimodal interfaces have the potential to lower user cognitive load, allowing them to concentrate on the decision-making task.

Previous work has shown that users have a preference to interact multimodally in spatial application domains [23]. Cohen et al. [7] has shown that there can be a substantial efficiency advantage of multimodal interaction, compared with traditional WIMP-style GUIs, for a map-based application. Oviatt [24] has also noted that gesture is particularly useful for specifying spatial information. Most USAR tasks rely upon geospatial information, often presented as maps. As a result, multimodal interaction with gesture as one of the inputs becomes particularly relevant for the USAR application. Our more general goal is to develop a design framework that is applicable to a range of application domains.

1.2 System Setup

We constructed a tabletop display based on and modified from the one built by Ashdown and Robinson [1]. The custom tabletop structure includes four 1280×1024 pixel projectors (Dell 5100MP) that provide a 2560×2048 pixel resolution (see Figure 1-2). The projectors are connected to two NVIDIA GeForce 8600GT dual-headed graphics card on a Dual-Core 2.4GHz desktop PC with 2GB of RAM.

The display is projected onto a flat white surface digitizer (GTCO Calcomp DrawingBoard V), which uses a stylus as an input device. The digitizer is tilted 10 degrees down in front, and is placed at 41in (104cm) above the floor, following FAA's design standard to accommodate the 5^{th} through 95^{th} percentiles of population. Projected displays were mechanically aligned to produce a single, seamless large display area. One Fire-i$^{\mathrm{TM}}$Digital Camera from Unibrain is placed above the center of the tabletop at the same level of the projectors. It is used for hand tracking based on the method developed by Wang and Popović [38]. The method requires the user to wear a colored glove (see Figure 1-3) for easy and efficient pose estimation. We use the Google Earth web browser plug-in as our basis for 3D maps.

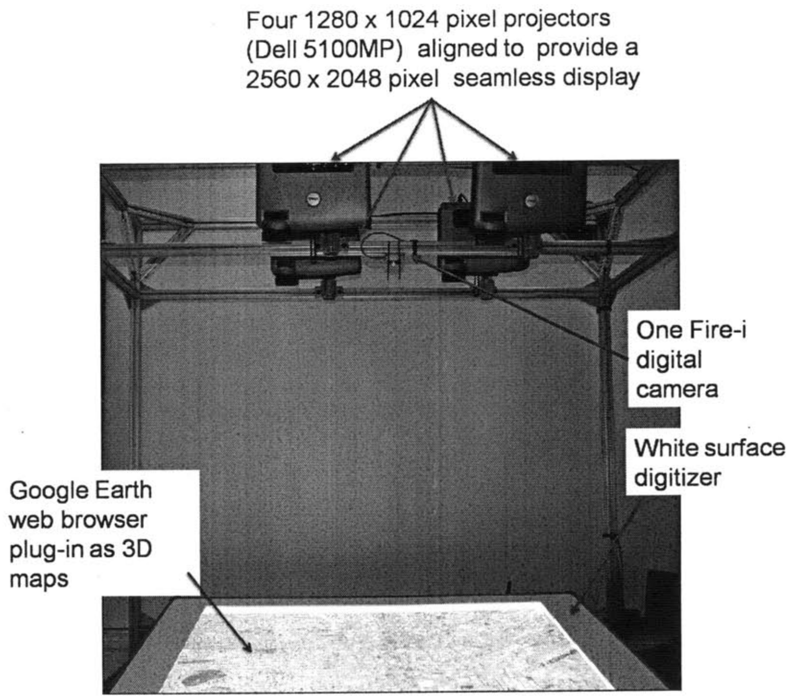


Figure 1-2: System setup

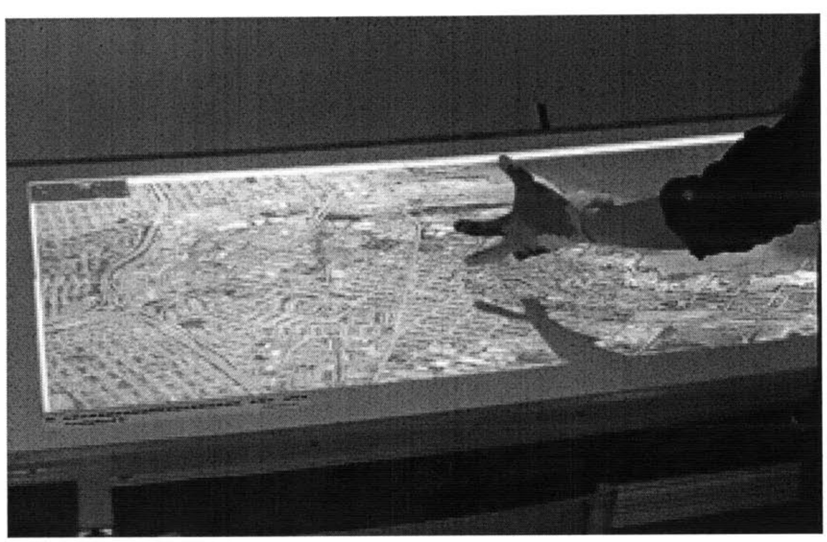


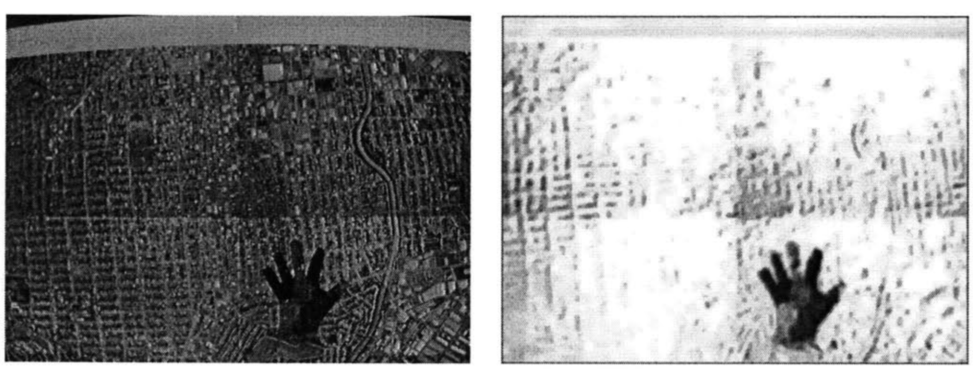
Figure 1-3: Hand tracking with a color glove

1.3 Contributions

Developing a multimodal system requires the integration of different input modalities. The input can occur in three distinct levels - data, feature, or decision level [9]. Among these three levels, decision fusion is the most robust and resistant to individual sensor noise and failure. It has a low data bandwidth and is generally less computationally expensive than feature fusion [17]. To explore the effectiveness of decision fusion, we will use a separate preliminary recognition module for each modality, and combine the results from these modules to provide the recognition outcome. This thesis work focuses on developing the preliminary recognition module for hand gestures.

We applied the new real-time hand-tracking technology developed by Wang and Popović [38] in our tabletop environment. The hand-tracking system allows us to explore finer 3D hand poses for interacting with computer interfaces that goes beyond the normal hand blob tracking [32] or touch-based interface [39].

As our setup has a more complex environment than the hand tracking system was originally designed for, we had to modify it to obtain good tracking results. Chapter 3 describes the details of the modifications. In particular, we developed a new, purely software-based method for removing the dynamic background produced by projectors. Our method computes the background in the camera image from the image being projected. Figure 1-4 shows the result of background removal of a camera image.



(a) (b)

Figure 1-4: (a) Camera image with the glove and complex background; (b) result after background removal

Chapter 4 describes our method for user-independent continuous gesture recognition based on the data from the hand-tracking system. We focus on natural gestures, which are free-form and have no fixed structure. We used two layers of Hidden Markov Models (HMMs), one for gesture segmentation based on the amount of hand motion, and one for gesture classification based on the segmented hand feature sequence. To date we have achieved 96% accuracy on isolated gesture recognition, and 74% correct rate on continuous gesture recognition with data from 3 different users and 12 gesture classes.

We are also able to connect the gesture recognition system with Google Earth to demonstrate the result of real-time hand tracking and gesture recognition. In particular, with our gesture recognition system, we can do 3D tilting of the map using nontouch-based gesture which is more intuitive than touch-based ones.

We also did a exploratory user study to observe natural behavior under a USAR scenario with a large tabletop display. Chapter 5 describes the experimental design, procedure, and results. As we did a Wizard-of-Oz style study, the experiment is highly realistic in eliciting natural multimodal input including speech, hand, and pen-based gesture. The qualitative results from the user study provides us with good starting points for understanding how users will naturally gesture, and for future work in combining the different modalities in the further development of our interface.

Chapter 2

Related Work

This chapter describes previous work related to gesture input and multimodal systems.

2.1 Systems with Gesture Input

Multi-touch displays have gained significant media attention and popularity with the introduction of the iPhone® and Microsoft Surface®. Their wide popularity shows the great potential and demand for natural and convenient input techniques. However, with touch-based input, the interaction is still limited in 2D space, and some 3D interaction cannot be realized, or are hard and unnatural to specify. For instance, it will be difficult to use a touch-based display to rotate a map in 3D. These new interfaces are more of a replacement for mouse input, with the addition of allowing multiple simultaneous control points.

Moving from 2D to 3D, Oblong Industries' g-speak spatial operating environment[1] allows free hand input in 3D space. However, it focuses only on using hand gesture for manipulating pixels, instead of trying to exploit the communicative power of gesture.

The introduction of Nintendo Wii is a break-through in interaction techniques for the gaming industry. Its success once again indicates people's preference for natural interaction, i.e., movement similar to what we do in the real world. At Electronic En-

[1] `http://oblong.com/`

tertainment Expo 2009, Microsoft unveiled its Project Natal, a controller-free gaming and entertainment system. It enables users to play video games through a natural interface using gestures and spoken commands. The device provides full-body 3D motion tracking. However, it does not track the hand posture to the details of finger positions Some features of Project Natal are yet to be implemented, but it does paint an exciting future for natural human-computer interaction.

In our research, we focus on 3D free-form gestures that are not confined to the surface of the display. Instead of just tracking the hands as finger tips or blobs, we explore the expressiveness of the hand poses that can enhance human-computer interaction.

2.2 Gesture Recognition

Much work on computer recognition of hand gestures has focused on synthetic gestures, most notably sign language recognition. Bauer and Hienz [2] developed a video-based continuous sign language recognition system. The system was based on continuous density Hidden Markov Models (HMMs) with one model for each sign. The signer wore simple cotton gloves with colored markers for real-time data acquisition and easy information retrieval. A single color video camera was used for image recording. They used 3.5 hours of training and 0.5 hours of test data from one person's input, and reported an accuracy of 91.7% based on a lexicon of 97 signs.

The research in the sign language recognition provides us with a good start point on natural gesture recognition. However, there are still many differences between the two. Sign language is well structured and has a defined grammar. Natural gestures are free formed and they can occur at any time in any order. Sign language has a large but limited vocabulary, but natural gestures have potentially unlimited number of possibilities. These differences pose great challenges for natural gesture recognition.

Because there is no particular pattern in terms of the sequence of the gestures, and because gestures can be sporadic in natural interaction, we did not use embedded training which is commonly used in speech recognition [40] and sign language recognition.

Cassell [6] was among the first to argue that natural, free-hand gestures can be relevant to HCI, and presented a helpful framework for gesture interaction. Kettebekov and Sharma [17] focused on using natural gestures when developing a multimodal framework for large display control. They report that it is possible to recognize natural gestures continuously with reasonably good rates. Building on this, Krahnstoever et al. [18] described the technical detail of developing several research prototypes based on the multimodal framework. For the continuous gesture recognition part, they focused on deictic gestures, and the system was trained to learn pointing, area and contour gestures.

Our work is in part built on the foundation given there. For gesture recognition, we also support manipulative gestures besides the deictic gestures, and we focus on the manipulative gesture recognition first in this thesis.

2.3 Multimodal Systems

2.3.1 Systems in General

Bolt's pioneering work in the "Put That There" system [3] demonstrated the potential for voice and gestural interaction. In that system, the hand position and orientation was tracked by the Polhemus tracker, i.e., the hand was essentially transformed to a point on the screening. The actual hand posture did not matter, even if it was not in a pointing shape. The speech also followed a rigid and limited command-like grammar. Even though this is an early work, it provides some insight about the advantages of multimodal interaction. As Bolt summarized in the paper, using pointing gesture allows the use of pronouns in the speech, with the corresponding gain in naturalness and economy of expression [3].

More recently, several multimodal interaction prototypes were developed that moved beyond Bolt's "Put That There" system. Cohen et al. [8] developed the Quick-Set prototype which was a collaborative, multimodal system running on a hand-held

PC using pen and voice as input. They used a novel multimodal integration strategy that allows speech and pen gesture to compensate for each other, yielding a more robust system.

Kaiser et al. [15] described an approach to 3D multimodal interaction in immersive augmented and virtual reality environments. Their multimodal system fused symbolic and statistical information from a set of 3D gestures, spoken language, and referential agents. Through a user study, they showed that mutual disambiguation accounted for over 45% of the succussful 3D multimodal interpretations.

These works demonstrate the value of multimodal interaction to improve interpretation accuracy using mutual disambiguation. In environments that would challenge unimodal speech recognizer, particularly for non-native speakers and in mobile environments [19], the benefits of multimodal interaction is even more prominent.

2.3.2 Systems Related to USAR

The work that is most related to ours is that done by Sharma et al. [33]. They did substantial amount of work in analyzing the issues in designing speech-gesture driven multimodal interfaces in the context of crisis management. They did a domain and task analysis to understand the highly dynamic and collaborative work domain of crisis management. As part of this work, Brewer [4] conducted onsite visits to state and county level emergency managers and GIS professionals in South Carolina and Florida to map out the process of emergency management response to hurricanes, to design interface prototypes, and to aid in the creation of realistic emergency management scenarios. Their domain analysis provided us an useful starting point in designing our multimodal interface for the USAR application.

Based on the domain analysis, and as part of the same research effort, Rauschert et al. [29] developed a system called Dialogue-Assisted Visual Environment for Geoinformation (DAVE_G) that used free hand gestures and speech as input. They recognized that gestures are more useful for expressing spatial relations and locations. Gestures in DAVE_G included pointing, indicating an area and outlining contours. Speech and gesture were fused for commands that needed spatial information provided by the gesture. Their work, however, mainly tracked hand location, rather than tracking both location and posture, as in our work.

Chapter 3

Hand Tracking

An important part of multimodal interface is acquiring valid multimodal data. For each modality, we need to obtain accurate input data in order to perform the subsequent recognition. We focus on gesture acquisition first; this chapter describes how we capture hand motion information.

3.1 Related Work

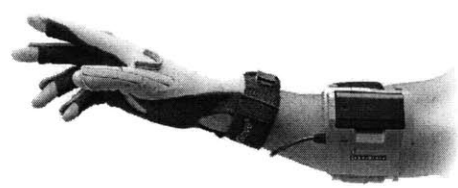


Figure 3-1: Wireless CyberGlove II motion capture data glove by CyberGlove Systems

The most common acquisition methods for hand gestures are magnetic trackers, cybergloves and vision-based approaches. Acquisition using magnetic trackers and cybergloves is efficient and accurate, but suffers from the need to wear restrictive devices (see Figure 3-1[1]). Most existing vision-based hand tracking systems track only hand movement and finger tips, rather than 3D hand postures [10][22][29]. This limitation often requires that artificial gestures be defined for easy tracking and recognition.

[1]`http://www.cyberglovesystems.com/products/cyberglove-ii/overview`

3.2 Basic Tracking Method

Figure 3-2: Hand tracking with a color glove

We use the hand-tracking system developed by Wang and Popović [38]. With one web camera and an ordinary cloth glove imprinted with a custom pattern (see Figure 3-2), the system can track 3D hand postures in real-time. It provides rich hand model data with 26 degree of freedom (DOFs): six DOFs for the global transformation and four DOFs per finger. The glove is very light-weight, with no additional electronics or wires. As a result, the gestures we can use for interaction are not limited by the hand-tracking hardware, opening up possibilities for developing and investigating more natural gestural interaction.

For each frame of the image from the camera, the system classifies each pixel either as background or as one of the ten glove colors using Gaussian mixture models trained from a set of hand-labeled images. After color classification, the system uses mean-shift with a uniform kernel of variable-bandwidth to crop the glove region [38]. The region is then normalized, and the resultant image is used to look up the nearest neighbors in the database.

Wang and Popović [38] used a single Point Grey Research Dragonfly camera for hand tracking. This model is ceased to be manufactured. We used a Fire-i$^{\mathrm{TM}}$digital camera instead which is closer to an everyday webcam that people would use at home. This camera also works better with the Dell projectors we use. The projectors in our setup (Section 1.2) are DLP projects which use a spinning color wheel to modulate the image. This produces a visible artifact on the screen, referred to as the "rainbow effect[2]", with colors separating out in distinct red, green, and blue. At any given instant in time, the image on the screen is either red, green, or blue, and the technology relies upon people's eyes not being able to detect the rapid changes from one to the other. When using a Logitech Qickcam Pro 9000, we observed horizontal color stripes in the camera image. The Fire-i camera does not have this artifact; instead, the whole image turns red, green or blue periodically. We are able to reduce this effect on the Fire-i camera by adjusting its exposure rate to be a multiple of the rotation period of the color wheel. The projector uses 2x color wheel speed, which is about 120 rotation/second. Hence, the period is about 8.3ms. After adjusting the exposure rate of the camera to approximately 8.3ms, the effect is greatly mitigated, but still present.

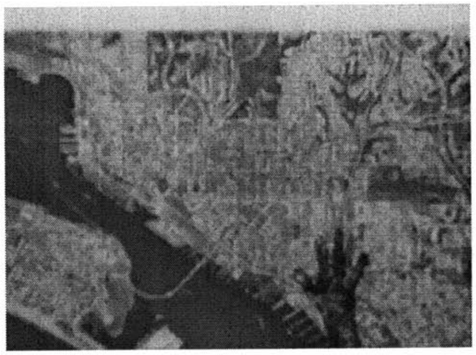
We also developed a wrapper for the Fire-i camera driver to interface with the hand tracker using the FireAPI™1394a/1394b development toolkit. We used the Fire-i DLL interface which directly talks to the camera and offers better performance. This allows the tracker to continuously capture images from the camera with minimum latency. The camera is set to capture 640×480 video with RGB colors at 15Hz.

3.3 Background Removal

Wang's hand-tracking software was developed originally for use in an office environment with standard illumination. The accuracy of the tracking relies largely on the accuracy of color classification of the colored patches on the glove. However, in our tabletop environment, we have a complex and dynamic background (the maps), and non-uniform illumination of the glove from the maps. This poses great challenges for the color classification step described in the previous section (see Figure 3-3(a)).

When the background is simple (as in a an office), color classification is often almost correct, i.e., only the hand is classified as colored pixels. Then we can crop

[2] `http://www.projectorcentral.com/lcd_dlp.htm`



(a)

(b)

Figure 3-3: (a) Camera image with complex background; (b) result of color classification without background removal

the region of the hand in the image easily using the mean-shift method which involves calculating the centroid of all the colored pixel. However, with a complex background that has regions with colors close to the ones on the glove, those regions will be classified as colored pixels instead of background (Figure 3-3(b)). These misclassified colored pixels can adversely affect the centroid calculation.

To remove the effect of the background, we use background subtraction. The way it is done, despite the name, is by division, i.e., dividing the camera image pixel-by-pixel by the background image. Traditional background subtraction involves taking an image of the static background only, or keeping a running average which is a linear combination of the current image and the running average image [13], and then subtracting the background image from the current image.

In our case we have the advantage that the background is the display, and we know what is being displayed by taking a screen capture. We then apply geometric and color transformations to the screen capture to determine how the background image looks to the camera. Figure 3-4 shows how the image pixels from the computer graphics card are transformed as they are projected to the tabletop and then captured by the camera. Steps 1 - 4 are the physical process what produces the image (I_A) in the camera. We want to compute the background image (I_B) to eliminate it from I_A. Step T is the computational process that produce the same effect as steps 1 - 4. In step T, we take the image (I_C) directly from the graphics card through screen capture and transform (scaling, cropping, and color mapping) it to I_B.

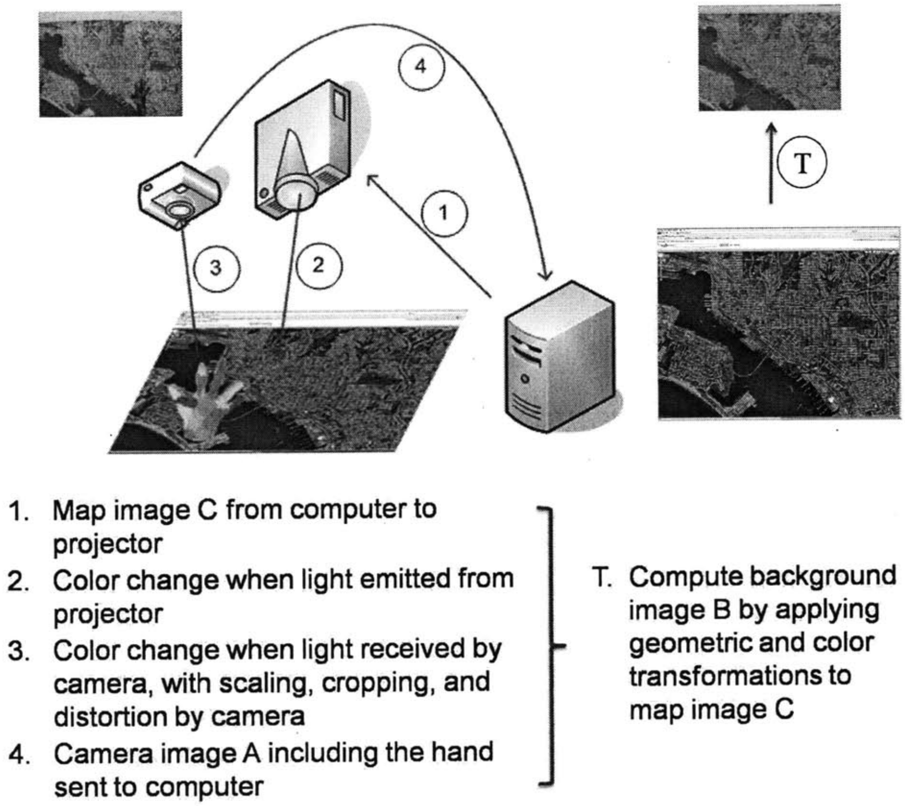


Figure 3-4: Transformation of image pixels

As the projectors are above the table, the light from the project also shines on the hand. The color signal reaching the camera is the multiplication of illumination and reflectance of the color glove for each pixel. To eliminate the effect of the illumination on the glove, we first rectify the camera image (removing distortion in I_A) to get I_{A_RECT}, and then divide its values pixel-by-pixel by the screen capture values after transformation (I_B). We use the RGB color model, so each color pixel is a triplet with values representing the intensities of red, green and blue channels. We use the subscripts r, g, b to denote the three channels through out this chapter. Let $\underline{p}_A$ be a pixel in I_{A_RECT} with RGB values (p_{Ar}, p_{Ag}, p_{Ab}), and $\underline{p}_B$ be a pixel in I_B, then a pixel $\underline{p}_R$ in the resultant image I_R is calculated as

$$\underline{p}_R = (p_{Rr}, p_{Rg}, p_{Rb}) = (\frac{p_{Ar}}{p_{Br}}, \frac{p_{Ag}}{p_{Bg}}, \frac{p_{Ab}}{p_{Bb}})$$

Figure 3-5 shows an example of the result after division, where the background is mostly white and the glove colors are closer to the actual colors. The result is not perfect, due to imperfections in the geometric and color calibrations, but color classification of the glove colors based on the image after this background elimination is more robust.

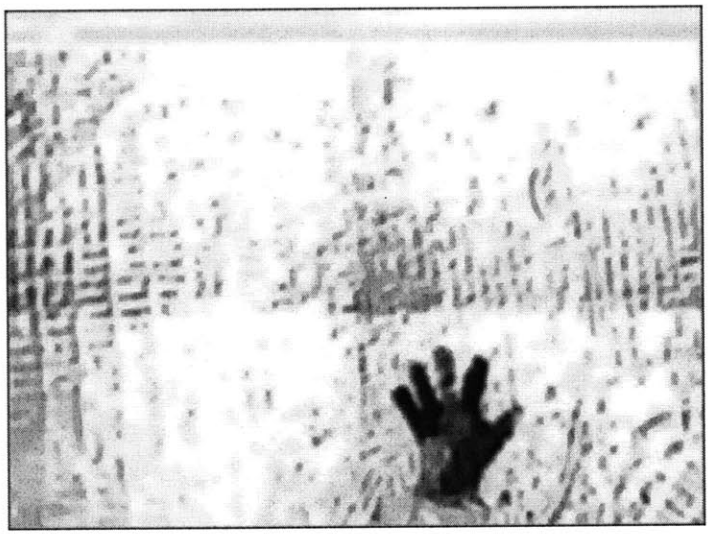


Figure 3-5: Resultant image after background elimination

3.3.1 Geometric Calibration and Transformation

There are two steps of geometric calibration and transformation we need to perform. The first one is rectifying the camera image; it has noticeable barrel distortion (see Figure 3-6(a)). We used the Camera Calibration Toolbox for Matlab[3] to obtain the intrinsic parameters (focal length, principal point and distortion coefficients) of the camera.

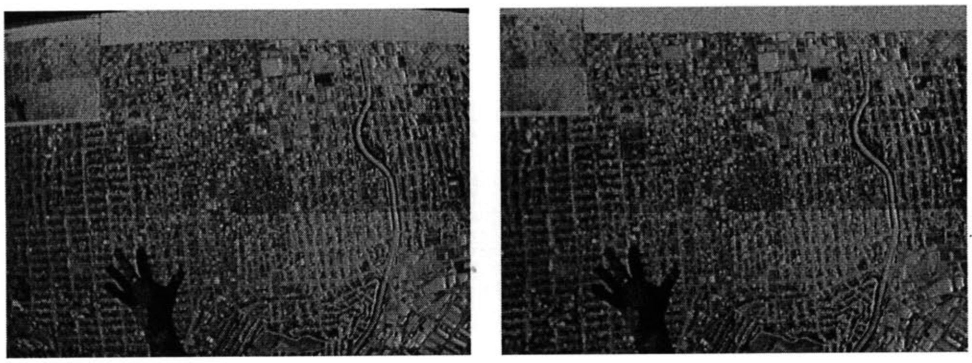
For realtime performance, we pre-generate a lookup table (LUT_{RECT}) that maps the x, y-coordinates in the rectified image to the coordinates in the original distorted image. Let I_{RECT} and $I_{DISTORT}$ be the rectified and the distorted images respectively,

[3] `http://www.vision.caltech.edu/bouguetj/calib_doc/`

then the pixel at (x, y) of I_{RECT} is

$$I_{RECT}(x, y) = I_{DISTORT}(LUT_{RECT}(x, y)).$$

Figure 3-6 shows the result of rectifying the original image.



(a) (b)

Figure 3-6: (a) Distorted image captured by the camera; (b) rectified image

The second step is calibrating the extrinsic parameters (rotation and translation) of the camera relative to the tabletop display. This step is necessary for transforming I_C (screen captured the image) to I_B in Figure 3-4.

The calibration process involves displaying a checkerboard image (target image) on the tabletop display, and recording the x,y-coordinates of the grid corners of the checkerboard image. We take a picture of the tabletop with the target image displayed on it. Then we rectify the camera image and record the corresponding coordinates of the grid corners on the image. Using homogeneous coordinates, we find the least error transformation Q that maps the tabletop display coordinates to the camera image coordinates.

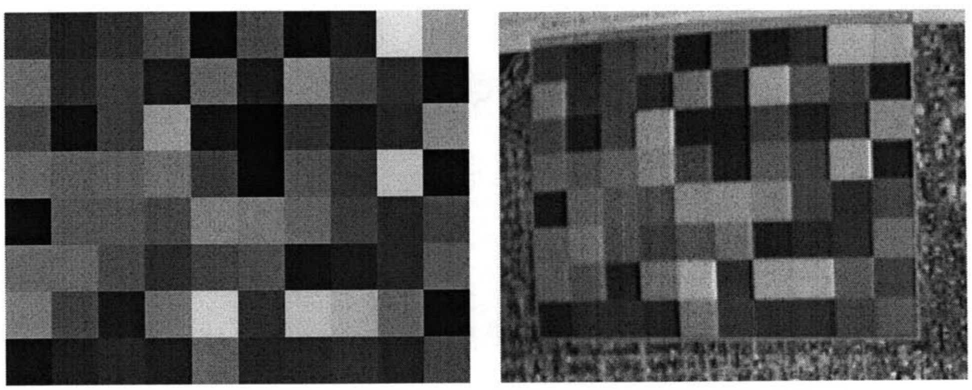
We also use a lookup table to improve the speed of the transformation. Let I_{SCREEN} and I_{TR} be the screen captured and the transformed image respectively. Then the pixel at (x, y) of I_{TR} is

$$I_{TR}(x, y) = I_{SCREEN}(LUT_{TR}(x, y))$$
$$\text{where } LUT_{TR}(x, y) = Q^{-1}((x, y)).$$

3.3.2 Color Calibration and Transformation

For background subtraction, we also need to transform the projected colors to the colors captured by the camera. Let $\underline{c}_p$ be the projected color and $\underline{c}_c$ be the camera captured color. We want to find a function $\boldsymbol{f} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ where $\boldsymbol{f} = [f_r, f_g, f_b]$ such that $\underline{c}_c = \boldsymbol{f}(\underline{c}_p)$.

To find function $\boldsymbol{f}$, we did a color calibration using a palette with 80 colors (see Figure 3-7(a)). The colors are chosen because they are the most frequent colors in Google Earth. Figure 3-7(b) shows the same color palette as seen from the camera. From these two images, we obtained a set of training examples $(\underline{c}_p^i, \underline{c}_c^i)$ for $i = 1 \ldots 80$.



(a) (b)

Figure 3-7: (a) Color palette for color calibration; (b) the same palette captured by the camera

We experimented with three methods of finding the mapping function $\boldsymbol{f}$. Method 1 assumes the transformation is linear, so for example, the R value $c_{cr} = a_{r3}c_{pr} + a_{r2}c_{pg} + a_{r1}c_{pb} + a_{r0}$. The equations for G and B values are similar.

Method 2 assumes the transformation is a second degree polynomial, but each color channel is independent, i.e., $c_{cr} = a_{r2}c_{pr}^2 + a_{r1}c_{pr} + a_{r0}$. We find the parameters using least-square error methods.

For Method 3, we also assume a second degree polynomial model, but each color channel depends on all three color channels. Let x be one of the color channels, i.e., $x \in \{r, g, b\}$, and $f_x(\underline{c}_p) = \underline{\theta}_x \cdot \underline{\phi}(\underline{c}_p)$ where $\underline{\phi}(\underline{c}_p)$ contains all polynomial terms up to degree 2 (e.g. $c_{pr}, c_{pg}, c_{pr}^2, c_{pr}c_{pg}$ etc.). We use regularized least-squares regression to find the optimal parameter $\underline{\theta}_x$. The regularization term $||\underline{\theta}_x||$ is added to prevent over-fitting.

To evaluate the three color transformation methods, we use four 640×480 pixel testing background images (with different levels of details and a variety of colors like water bodies and forests) in both projected form and camera captured form. We color transform the projected images using the three methods, and divide the camera captured image by the color transformed image pixel by pixel in each R, G, B value. Ideally, we would obtain a white image after the division. However, the color calibration is not perfect. Hence, the evaluation metric we use measures how much each pixel in the resulting image deviates from a neutral (gray) color. Any neutral color is distinct from the colors on the glove; it need not be pure white. With a neutral background, the accuracy of color classification of the glove can be improved. If n is the number of pixels in image I, (p_{ir}, p_{ig}, p_{ib}) the RGB values of pixel p_i, and $\bar{p}_i$ the average value of (p_{ir}, p_{ig}, p_{ib}), the metric M for image I can be expressed as

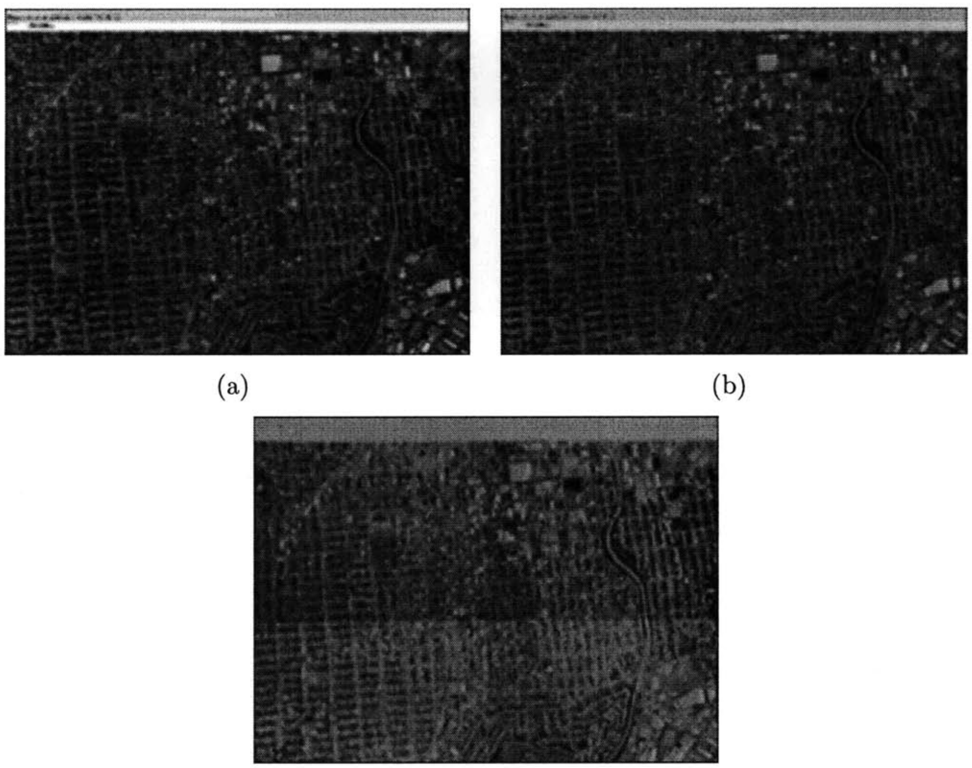
$$M(I) = \sum_{i=1}^{n} (p_{ir} - \bar{p}_i)^2 + (p_{ig} - \bar{p}_i)^2 + (p_{ib} - \bar{p}_i)^2.$$

Table 3.1 shows the result of average $M(I)$ for the four testing images. Method 3 has the lowest value which means less deviation from neutral colors. As a result Method 3 is used for color transformation. Figure 3-8 shows the result of color transformation using Method 3 (all images are blurred to eliminate the high frequency noise).

Method	**Average $M(I)$**	**Classification Accuracy**
1. linear	666	0.9986
2. polynomial, independent channels	600	0.9979
3. polynomial, dependent channels	497	0.9988

Table 3.1: Evaluation metrics for the three color transformation methods

As the goal of background elimination is to improve the the accuracy of color classification, we also evaluate the three methods based on the result of color-classification. We tested with the same four different background images and took several color classified images to calculate the average accuracy. The ground truth classification is done



(a) (b) (c)

Figure 3-8: (a) Scaled background image in its original color through screen capture; (b) background image after color transformation; (c) rectified camera image manually. Table 3.1 also shows the result of the average accuracy using the three different color transformation methods. The accuracies are all very high, but Method 3 gives the best result. Figure 3-9 shows the result of color classification when using Method 3 to transform the color of the screen captured background.

Figure 3-9: Result of color classification after background removal using Method 3 for color transformation

3.4 Discussion

We applied a new hand-tracking technology in our tabletop environment. Despite the complex lighting environment and background, we can obtain good tracking results by reducing the adverse effects computationally.

We developed a new background removal method for a projector-based system. With geometric and color calibrations, the background removal can be done computationally and relatively accurately. Although there may be other methods, for example, using polarizing filter or synchronized optical shutters [14], our method is purely software-based, and hence, does not require additional hardware setup.

Chapter 4

Gesture Recognition

The hand tracking system described in Chapter 3 provides us with detailed 3D hand pose data. In this chapter, we describe the methods we use to infer gestures from that data.

4.1 Gesture Taxonomy

We adopt the taxonomy of hand movements proposed by Pavlović et al. [25], which distinguishes gestures from unintentional hand movements (like beats). They then further divid the gestures into manipulative and communicative.

Manipulative gestures are used to act on objects, while communicative gestures have an inherent communicational purpose [25]. In a natural environment, communicative gestures are usually accompanied by speech. Hence, for manipulative gestures, classification is based on the hand states, while for communicative gestures, both hand and speech recognitions will be combined for recognition. In this chapter, we focus on manipulative gesture recognition where speech is not involved.

4.2 Temporal Modeling of Gestures

Human gestures are a dynamic process. If recognition is to work, it is important to understand the temporal (dynamic) characteristics of gestures. Psychological studies of gestures provide us with some insights to the properties of gesture intervals. Kendon [16] calls the gesture interval a "gesture phrase". It has been established that three phases make a gesture: preparation, nucleus (peak or stroke [20]), and retraction. The preparation phase consists of preparatory movement that sets the hand in motion from a resting position. The nucleus of a gesture has some "definite form and enhanced dynamic qualities" [16]. In the retraction phase, the hand either returns to the rest position or is repositioned for the next gesture.

4.3 Isolated Gesture Recognition

4.3.1 Feature Vector

The output from the hand tracker is a sequence of data describing the translations in x, y, z coordinates, and orientations in quaternions of the hand and each finger joint. Figure 4-1 shows the coordinate system we use. There are three joints per finger in the model. The joint at the base of each finger has 2 DOFs while the other two joints of each finger have 1 DOF each. From the tracker output, we derive the feature vector at each time step. For each finger, we are interested in the bending angles of the joints, so we convert the joint orientation in quaternions to Euler angles. The translation of the joint is irrelevant because it is relative to the hand and stays the same. The feature vector we use fro recognition includes the velocity of hand movement in the x-y plane (obtained from the global translation of the hand), z position of the hand, roll, pitch and yaw of the hand, and four angles for each finger (one angle for each of the first two joints and two angles for the base joint). The result is a 26-dimensional feature vector $\underline{x}$.

The feature vector produces a description of the hand motion and pose, and as such provides more generality than would be available from a set of features chosen because they discriminated among a set of predetermined gestures. Our feature vector gives us the flexibility of investigating and training different gestures that can be obtained through a user-centered approach instead of prescribing a rigid set of gestures.

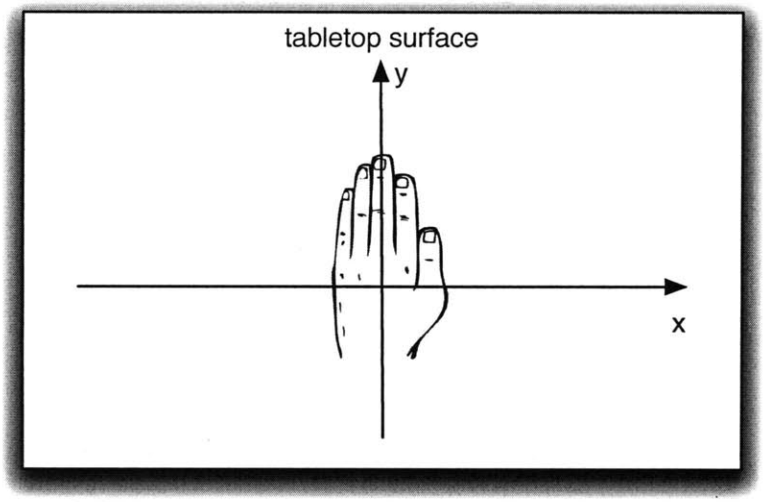


Figure 4-1: The coordinate system we use: the table surface is the x-y plane, with z axis coming out from the table surface

4.3.2 Hidden Markov Models

The position and the orientation of the hand through time can be assumed to follow the first order Markov process [35]. Hence, we use Hidden Markov Models (HMMs) to classify gestures. There exist many kinds of HMMs [28]. One that can model time-series signals whose characteristics change successively over time is called the Bakis model [2] or the Left-Right model [27], often used in speech recognition systems [2]. The Bakis model allows transitions to the same state, the next state, and the one after the next state. It is particularly useful for our task because it allows different gesture speeds to be compensated [2].

Figure 4-2 shows an example of a four-state Bakis model with transition probability from state s' to state s as $t(s|s')$ for $s, s' \in \{1, 2, \ldots, m\}$. We also add the non-emitting entry and exit states to the model, similar to what Young et al. [40] did for their speech recognition system. The entry state is added to represent the initial state parameters $t(s)$. Only the first two states can be the initial state, and only the last two states can transit to the exit state.

There is one HMM ($\underline{\theta}_k$) trained for each gesture k. The probability of an observed sequence $P(\underline{x}_1, \ldots \underline{x}_t; \underline{\theta}_k)$ will be evaluated for all competing models, with the classifi-

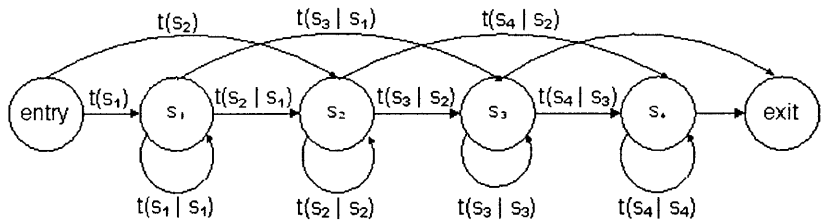


Figure 4-2: The state transition diagram of a 4-state Bakis model with corresponding transition probabilities

cation based on the model that gives the highest log-likelihood. More formally, the classification for an observation sequence $\underline{x}_1, ..., \underline{x}_t$ is:

$$\hat{k} = \arg\max_k \log P(\underline{x}_1, ... \underline{x}_t; \underline{\theta}_k). \tag{4.1}$$

4.3.3 Model Selection

Emission Probabilities

We start with the simple case of recognizing isolated gestures from one single user. We define emission probability e using a simple Gaussian distribution whose parameters depend on its underlying state. More specifically,

$$e(\underline{x} \mid s) = N(\underline{x}; \underline{\mu}_s, \Sigma_s) \tag{4.2}$$

We then generalize the method to accommodate multiple users by using a Gaussian mixture model for each state to account for the variance among different people. It is assumed that each of the m states has its own set of l mixtures, so there are $l \times m$ mixture components. Let $q_s(z|s)$ specify a distribution over the l possible mixture components, which depends on the underlying state s, so $z \in \{1 \dots l\}$. Hence,

$$e(\underline{x} \mid s) = \sum_{z=1}^{l} q_s(z \mid s) N(\underline{x}; \underline{\mu}_{s,z}, \Sigma_{s,z}). \tag{4.3}$$

Note that Equation 4.2 is just a special case of Equation 4.3 when $l = 1$.

Model Size

The choice of the size of a model (the number of states) is an important issue in implementing HMMs. An underlying state in HMMs represents a particular velocity, orientation and shape of the hand of the user during the continuous movement. Hence, the states can be associated with the temporal gesture phases described in Section 4.2. As a result we can say that a gesture HMM should contain at least (and usually more than) three (hidden) states [25].

A literature review shows that people have used different numbers of states in their Bakis model for gesture recognition. Starner and Pentl [35] used four-state HMMs for all signs, while von Agris et al. [37] used an average of 41 states. The discrepancy may lie in the differences of the data sampling rates. Starner and Pentl [35] used a 5 frame/sec rate for data recording, while von Agris et al. [37] used a 25 frame/sec rate.

Intuitively, using more states means discretizing the whole movement further, which should lead to better accuracy. However, there is also a trade-off between the complexity of the model and over-fitting the dataset. In addition, the number of states should also be related to the sampling rate (around 12 frame/sec in our case) which affects the number of frames per gesture. We use cross-validation to determine the choice of model size.

4.3.4 Experiments and Results

To test the performance of our recognition method, we obtained training data for 12 gestures chosen for basic map manipulations. The gestures include pan left, right, up and down; pitch, roll and yaw in clockwise and anticlockwise directions; and zoom in and out. Panning is moving the hand left, right, up or down, with the hand flat on the tabletop surface. Pitch, roll and yaw gestures are rotating the hand about the x, y, and z axes respectively (see Figure 4-1 for the coordinate system). Zooming in is spreading the fingers outwards, while zooming out is the reverse action, i.e., moving the fingers inwards. Figure 4-3 shows the examples of some gestures we used for training. As the feature vector we use is not influenced by these gestures, we believe our the results will be generalizable for other gestures.

Figure 4-3: Examples of some gestures used in the experiment

We used a MATLAB HMM toolbox [1] for obtaining the maximum likelihood parameters of the models using the expectation-maximization (EM) algorithm.

Single User

We collected a single user data set X_S over three different days. There are 9 samples for each of the 12 gestures. Three samples per gesture are set aside for testing, and the remaining six samples per gesture used for training.

A 3-fold cross-validation on the training data is performed to evaluate the effect of the number of states (m) on performance (Table 4.1). The results show that using 3 states give the highest accuracy rate. However, the results for using 4 or 5 states are also very close.

m	Accuracy Rate
3	0.736
4	0.722
5	0.708
6	0.611

Table 4.1: Cross-validation with different number of states

We then perform the training on all the training data, and the resulting models are tested with the remaining 36 test samples. Table 4.2 shows the results with different

[1] http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html

numbers of states. Using 3-5 states gives the best result.

m	Test Accuracy	Training Accuracy
3	0.972	1.000
4	0.972	1.000
5	0.972	1.000
6	0.917	1.000

Table 4.2: Test accuracy and training accuracy with different number of states

Multiple Users

As it is important to test whether the models generalize well to different users, we collected a second data set X_M with three different users. We use the Gaussian mixture model for the emission probability for each state and experiment with different values of l in Equation 4.3. We use k-means clustering for initializing parameter estimates for a mixture of Gaussians. Table 4.3 shows the performance when tested with different users. The training data includes 14 samples per gesture, and the test data includes 4 samples per gesture (48 test samples in total with data from three different users).

	$m = 3$	$m = 4$	$m = 5$
l=1	0.938	0.917	0.917
l=2	0.896	0.896	0.896
l=3	0.938	0.958	0.917
l=4	0.938	0.875	0.917

Table 4.3: Test accuracy on data from three different users with different number of mixtures (l) and different number of states (m)

Table 4.3 shows the performance when using different numbers of mixtures (l) for the emission probabilities, and different number of states (m). Using 3 mixtures of Gaussians and 4 states gives the highest recognition accuracy of 95.6%.

4.4 Continuous Gesture Recognition

The next important step is continuous online gesture recognition. Some previous systems distinguish gestures from unintentional hand movements by restricting the hand motion or defining some arbitrary gesture to indicate the start of a gesture [34]. For an interface that a user can interact with more naturally, we need a more flexible approach.

There are two issues we need to address. One is gesture segmentation, i.e., we need to detect the start and the end points in a continuous gesture sequence in order to use the HMMs trained for isolated gestures. The other issue is real-time recognition with minimum delay.

4.4.1 Segmentation

Gesture is signified by motion. We compute $\underline{d}_t$, the difference between the feature vectors of consecutive time steps, as

$$\underline{d}_t = \underline{x}_t - \underline{x}_{t-1}, \quad \forall t \in \{1 \dots T\}.$$

Note that there is no d_0. We then take the 2-norm of the difference vector, i.e., $||\underline{d}_t||$. A larger value in $||\underline{d}_t||$ signifies bigger motion. We use a two-state HMM to model the start and the end of a gesture (see Figure 4-4) using $||\underline{d}_t||$ as the observation sequence.

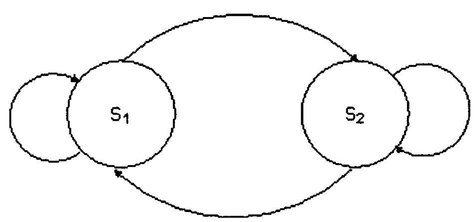


Figure 4-4: State machine for HMM: s1 = gesture starts, s2 = gesture ends

Figure 4-5 shows an example of the segmentation of a series of continuous gestures using the trained HMM. The green line is the $||\underline{d}_t||$ value. The red segments are the manually labeled gesture intervals. The blue segments are the gesture intervals resulted from HMM segmentation. Note that the manual labels are not perfect either because it is hard to determine the exact point in time a gesture starts or ends.

There are many fluctuations in the $||\underline{d}_t||$ values. Several factors may contribute to this. One factor is the noise in the feature vectors coming from the hand tracker. The z value in the feature vector is especially noisy and fluctuates greatly because there is no stereo imaging. The variation in the speed of the hand movement is another factor. Using an HMM is more robust than simply calculating the average value of $\|\underline{d}_t\|$ for gesture intervals, and using that as a threshold, because an HMM takes the transition probabilities into account, making it less sensitive to high-frequency fluctuations. However, there still can be false detections like the one shown in Figure 4-5.

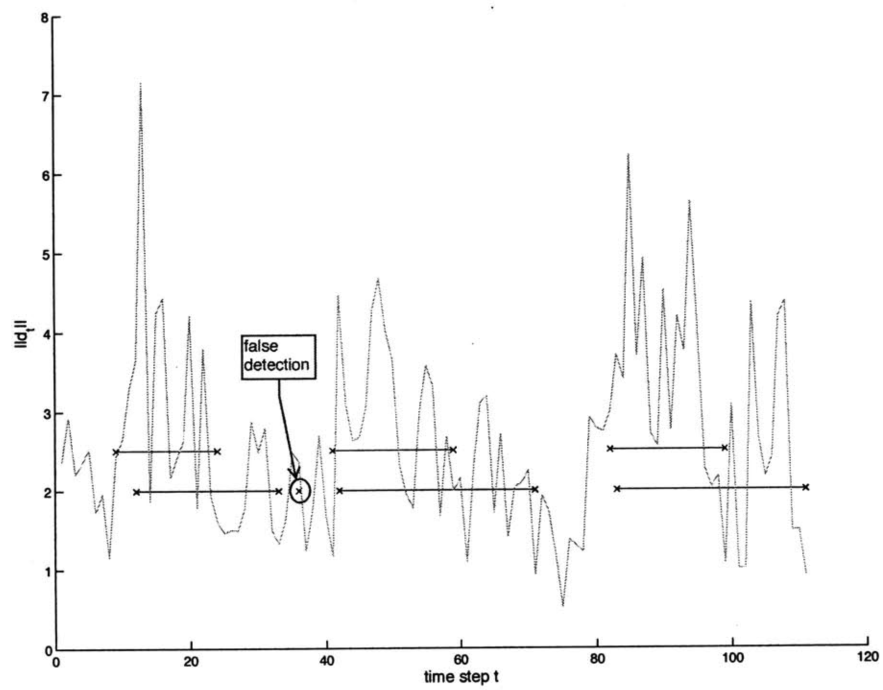


Figure 4-5: An example of gesture segmentation result

Figure 4-6 shows the system level structure of the continuous gesture recognition module. We have two levels of HMMs: the first level is used to detect the starting and ending points of gesture intervals; the second level is used to classify the gesture interval using the HMMs trained for isolated gestures. The results can be used to drive the GUI of an application, for example, Google Earth.

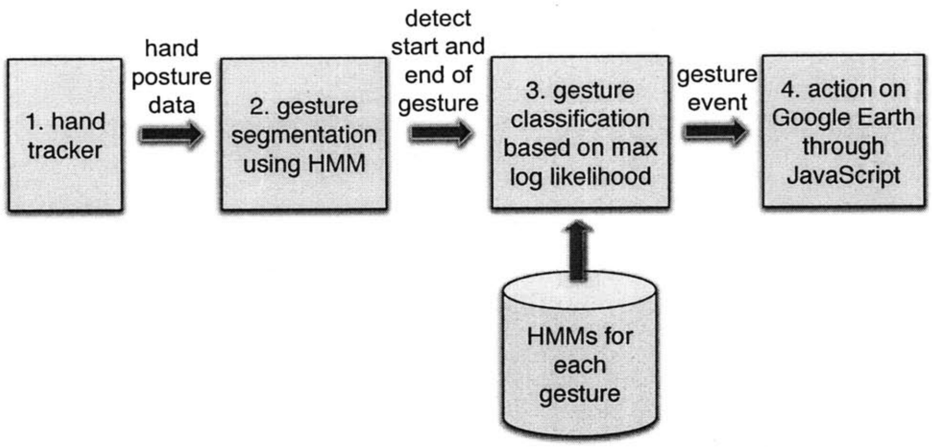


Figure 4-6: Continuous gesture recognition system

4.4.2 Real-Time Recognition

Achieving minimum time delay of gesture recognition is necessary for a real-time interactive system. To have a responsive system, we will ideally need to recognize the gesture even before it ends. Therefore, instead of evaluating the log-likelihood of a gesture HMM at the end of the gesture interval and finding the model with the maximum value, as in Equation 4.1, we evaluate the feature vector at each time step as it comes from the tracker, and update the estimated likelihood of each gesture.

Let t_s and t_e be the starting and ending time steps of a gesture. At time step $t_s \leq t \leq t_e$, we calculate the probability of the observation sequence $x_{t_s}, \ldots, x_t$ for the gesture model $\underline{\theta}_k$ as

$$P(x_{t_s}, \ldots, x_t; \underline{\theta}_k) = \sum_s P(x_{t_s}, \ldots, x_t, s_t = s; \underline{\theta}_k) = \sum_s \alpha[t, s]. \tag{4.4}$$

$\alpha[t, s]$ can be calculated using the forward algorithm:

$$\alpha[t, s] = \sum_{s'} [\alpha[t-1, s'] \times t(s|s') \times e(x_t|s)] \tag{4.5}$$

As the probabilities will become very small, our implementation uses logarithmic probabilities. We only need an array to keep track of the α values and the log-likelihood value for each gesture model. Hence, the storage requirement is $O(1)$ and time requirement is $O(t)$.

We analyzed the change of log-likelihood values of 12 gesture HMMs from the start of a gesture until the end for the isolated gestures in the test data set X_M. So for each gesture observation sequence $(\underline{x}_1, \ldots, \underline{x}_T)$, we plotted $log(P(\underline{x}_1, \ldots, \underline{x}_t; \underline{\theta}_k))$ against time t where $t = 1 \ldots T$ and $k = 1 \ldots 12$. We observed that the log-likelihood of the correct gesture model is the highest even very early in the gesture. This means that we can make an early decision about the gesture based on the highest log-likelihood value. Figure 4-7 shows one such example: the actual gesture is pan down, and the log-likelihood of the observation sequence with the gesture model parameter $\underline{\theta}_{\text{pan down}}$ is the highest at the very beginning.

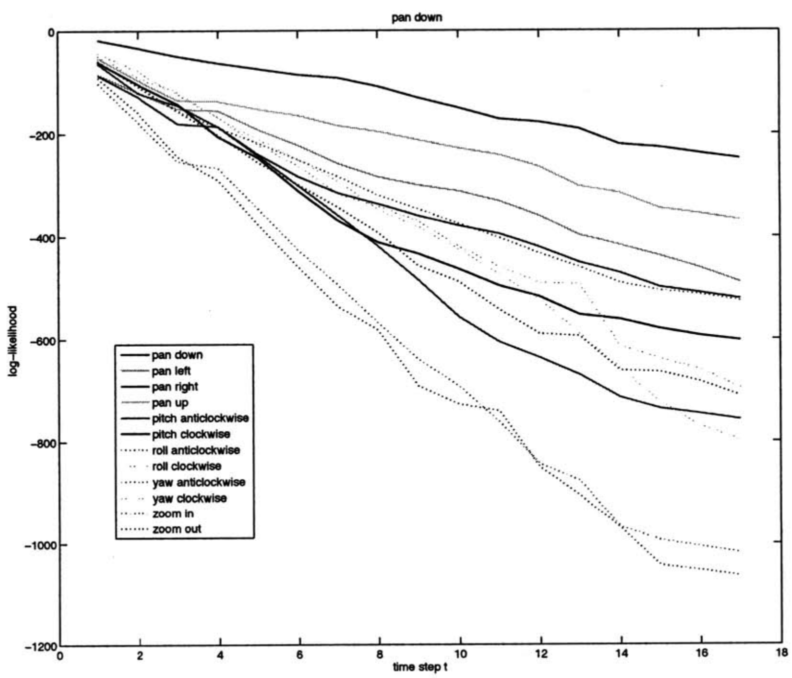


Figure 4-7: The change of log-likelihood values for 12 HMMs over time, the actual gesture is pan down

However, it is not true that the correct gesture model will always have the highest log-likelihood at the very beginning, because some gestures are similar at the start.

Figure 4-8 is one such example; the correct gesture model does not have the highest log-likelihood during the first few time steps, with all the log-likelihood values very close in the beginning.

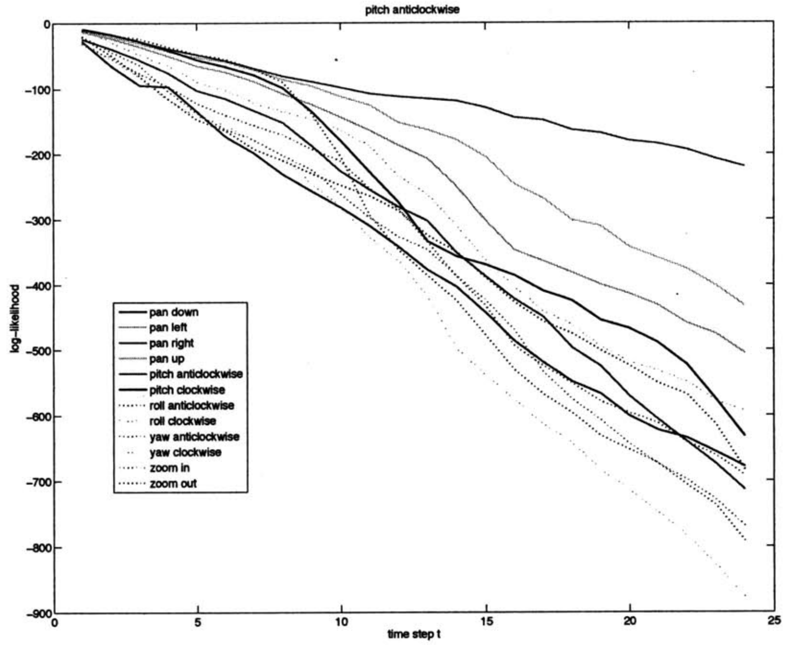


Figure 4-8: The change of log-likelihood values for 12 HMMs over time, the actual gesture is pitch anticlockwise

Eickeler et al. [11] used a similar method (calculating the probability of the feature vector sequence at each time step t) for continuous gesture recognition. For real-time recognition, they defined the condition for the moment of recognition as the following: if the probabilities of the observation sequence for all the models are decreasing during a short interval, the most likely model at the end of this interval is the recognition result. The reason behind this choice of the condition seems empirical, but we did not observe the same pattern as described by them. They did not report the recognition accuracy rate either.

Instead, we make the recognition decision only after the difference between the highest and second highest log-likelihood values are greater than an empirically derived threshold value. The reason is that greater difference in the probabilities means greater confidence that the highest one is the correct gesture. We obtain the threshold value through analysis of the test data in the isolated gesture data set, and calculate the average maximum difference between the highest and second highest log-likelihood values when the higher one does not correspond to the correct gesture.

4.4.3 Experiments and Results

We recorded a data set X_C containing sequences of continuous gestures, and manually labeled the gestures, and their start and end points to get the ground truth. We use four examples to train the HMMs for segmentation. As the data is fully labeled, we can get the maximum likelihood parameters directly instead of using the EM algorithm. The examples contains different number of gestures, ranging from 2 to 9, with 76 gestures in total.

We measure the recognition rate defined as the total number of correctly recognized gestures over total number of gestures (without considering the false positives). Note that Sharma et al. [32] used a similar metric for the performance of their recognition system, however they did not mention whether the timing of the recognition was also considered or whether they just compared the output gesture sequences with the true gesture sequences. For the recognition to be useful for an interactive interface, we define a correct recognition as one in which the label is correct and the output gesture interval overlaps with the true gesture interval. To date, we are able to obtain a recognition rate of 74%.

Figure 4-9 shows a graphical representation of the recognition process. It plots the log-likelihood values of 12 gesture HMMs against time for a sequence of gestures. The log-likelihood values are set to be 0 when there is no gesture detected. The upper horizontal line segments indicate the manual labels of the gestures in those time intervals. The lower line segments are the outputs of the recognition algorithm indicating the recognized gesture intervals. The lines with different styles correspond to different gestures.

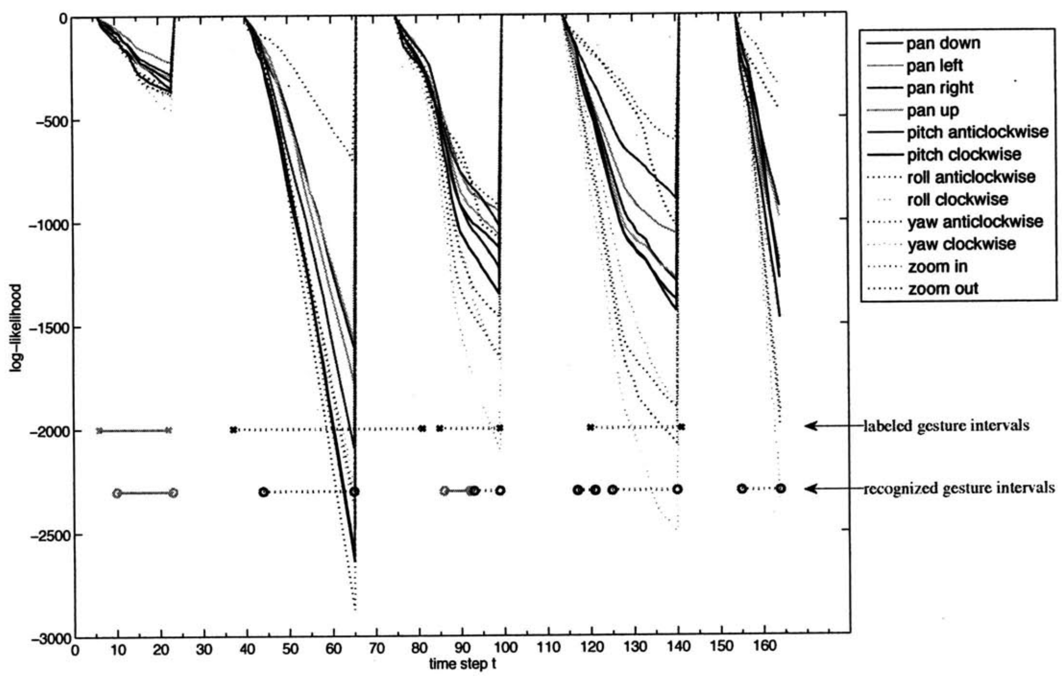


Figure 4-9: The change of log-likelihood values for 12 HMMs over time, the actual gesture sequence is pan left, yaw anticlockwise, zoom in, zoom in

4.5 Real-Time Interaction with Google Earth

We implemented the continuous gesture recognition algorithm in Java to make a real-time interactive system. Using the Google Earth Plug-in and its JavaScript API, we embedded Google Earth, a 3D digital globe, into a web page. We use the Java web browser object in the JDesktop Integration Components (JDIC) library to load the web page. In this way, we can augment the browser and make it respond to the gesture events from the gesture recognizer and provide the corresponding actions (panning, rotation, tilting and zooming) to the map. The actions on the map are written in JavaScript, invoked through the Java browser object.

When a gesture is recognized, we calculate the parameters needed for the corresponding action on the map. For instance, if a panning gesture is recognized, we calculate the translation of the hand to determine the amount of panning needed. The continuous gesture recognition described above is not perfect and can have false positives. However, as long as the false detection period is short, it may not have an effect on the action resulted from the gesture because the actual change in the parameters is negligible.

After an action is executed, the map is changed, and so is the background displayed on the tabletop surface. As a result, we need to re-capture the screen display to compute the background image for background removal.

4.6 Discussion

We obtained a good recognition rate for isolated gestures even with different users. For the continuous gesture recognition, the recognition rate is lower because there are two more sources of potential errors: the detection of start and end of the gestures, and the classification of the gesture even before the gesture ends. There are also false positives due to unintentional movement or repositioning of the hand. In the future work, we plan to add the hand retraction movement to the training of the isolated gestures to take this into account.

We defined the set of gestures only for the purpose of testing the performance of the recognition methods. We need to verify this performance with another set of gestures, probably obtained through our user study, to test the flexibility and generalizability of the system.

Chapter 5

User Study

An important feature of a natural interface would be the absence of predefined speech and gesture commands [33]. As our motivating application domain is natural interaction in USAR Command and Control, we need first to understand what kinds of spontaneous speech, drawings, and gestures people make in such scenarios. In this chapter, we describe the exploratory user study we conducted under an incident command system (ICS) environment using our tabletop display system, and summarize the main observations and insights we obtained that will be useful for designing and developing a natural multimodal interface.

5.1 Related Work

We need to learn from the multimodal data in order to develop a natural multimodal interface. However, the lack of an existing fully functional multimodal system that allows natural gesture and speech input creates a "chicken-and-egg" problem [33]. Various techniques have been used to address this problem.

Sharma et al. [32] studied the analogous domain of the TV weather broadcast to analyze the gesture/speech inputs by a subject under natural conditions. The most frequent gestures in that environment are pointing, contour and area gestures. While these are useful observations, they are still limited in terms of the large variety of ways users can possibly interact with a large display.

Micire et al. [21] conducted an experiment to determine the gestures that people would naturally use for controlling robots on multi-touch tabletop surfaces, rather than the gestures they would be instructed to use in a pre-designed system. They used a Mitsubishi DiamondTouch tabletop surface as the display, but with no visual or audio feedback to the participants. They found that prior experience of the participants introduced some bias into the gestures they wanted to use. In particular, selection and movement gestures were heavily influenced by standard mouse paradigms. Participants who had used iPhones used significantly more pinch gestures for zooming. They mentioned that the lack of visual audio feedback eliminated any potential biasing, but at the same time also removed any indications that the participant might be providing inconsistent or nonsensical input.

Salber and Coutaz [30] applied the Wizard of Oz (WOz) technique to the study of multimodal systems. The WOz technique is an experimental evaluation mechanism. It allows the observation of a user operating an apparently fully functioning system whose missing services are supplemented by a hidden wizard. They state that in the absence of generalizable theories and models, the WOz technique is an appropriate approach to the idenfication of sound design solutions. They showed the WOz technique can be extended to the analysis of multimodal interfaces, and developed Neimo, a multimodal Wizard of Oz platform using mouse, speech and facial expression as inputs. Their focus is formulating the requirements for a generic multimodal WOz platform rather than the findings from such an user experiment.

Oviatt [23] did a WOz study on using speech and pen-based input for interacting with map displays. They simulated a realistic task that asked users to select real estates using dynamic maps, and had an assistant to make the response in minimum delay. They videotaped the user's actions, and then later did a quantitative analysis that showed combining speech and pen input decreased the disfluencies and task completion time. Our experiment is similar to Oviatt [23]'s except that we are also interested in gesture input and our application domain is different.

5.2 Experiment Design

We conducted a WOz user study in which the experimenter viewed and interpreted user actions, then simulated system response as we believe this creates a more realistic environment to elicit natural user behavior. The system setup is the same as described in Section 1.2. Google Earth was used to provided a dynamic interactive map interface. We added markups to the map to simulate an earthquake scenario in a city where many major roads were impassable, and a fire had broken out in a chemical plant (see Figure 5-1).

Figure 5-1: Map interface with markups using Google Earth

Experimental subjects played the role of command center personnel charged with receiving event reports coming from the field, and making appropriate updates to the map display. As the user study is for bootstrapping an experimental testbed, we want minimum overhead in terms of system development. Hence, no attempt was made in this exercise to process the drawings, speech or gestures. Instead, as mentioned, the experimenter interpreted all user input and produced a contemporaneous response from the system. We videotaped the subjects and recorded their gestures and speech, then analyzed the results to find commonalities.

Nineteen people participated in the study. Six experiments involved two people working together, seven involved a single person working alone. We had two kinds of experiments to observe the difference in the interaction pattern when people work alone versus collaborate with others. All participants were from 18 to 40 years old; eleven were female. Seventeen of them were right-handed and two were left-handed.

Fourteen participants reported prior experience with tablet-based PCs, and one participant used it regularly. Seventeen had experience with the Apple iPhone or similar touch screen phones, and twelve of them used such phones on a daily basis. Seven have used multi-touch tabletop displays once or a few times. Eight participants had some experience with speech-enabled interfaces, but only one of them used it on a regular basis.

5.3 Procedures

Each participant was first shown a short video showing the capabilities of the map interface. The video only showed the outcome of the manipulation, for example, that the map could be tilted and rotated to see the 3D buildings and places could be marked. It did not show what inputs (gestures) were needed to achieve the output. This was done to avoid biasing the participants in the ways they could interact with the interface. They were then briefed on the experiment scenario and tasks. Participants were told they could use speech, gestures (both touch- and non-touch-based ones), a stylus pen or any combination of these to achieve the goal of the task. Other than specifying the available input modalities, an effort was made not to influence the manner in which people expressed themselves. The participants did not know that there was a "wizard" behind the scene that actually controlled the system.

In a questionnaire survey with 12 emergency managers in various states done by Sharma et al. [33], the results showed the need for a GIS-based emergency response system to support zoom, pan, buffer, display, and spatial selection of geospatial data. Based on this, the tasks we used are designed to cover some of the most frequent use cases required by a GIS-cased emergency response system, including pan, zoom, rotation in 3D, selection, and indicating a path. There are seven tasks in total in the experiment (see Appendix A).

The study began with the system prompting the participant(s) with a field report and a task. Each task involved some manipulation or updates to the digital map. When the task was done, the system prompted the participant(s) with the next one. The tasks were not disjoint, but formed a coherent sequence of actions that a commander would encounter in such a scenario. The reason for making the tasks realistic was to elicit natural behaviour. The experimenter sat behind the participant(s) at some distance. She observed the participant(s)' action through a video feed, and then controlled the map through a wireless keyboard correspondingly.

5.4 Results

As this is an explorative user study, we focus on a qualitative analysis of the commonalities in the observed user behavior instead of a quantitative analysis. We use the same gesture taxonomy mentioned in Section 4.1 for the analysis.

5.4.1 Manipulative Gesture

We consider panning, rotation and zooming as manipulative gestures, as they act on the map object. One commonality among all participants is that they move the map instead of moving the position of the camera when asked to pan or rotate. This is expected because the map is a more obvious object in front of them for manipulation where the camera viewpoint is only implicit.

Panning

The gestures participants used for panning were relatively similar, with some small variations. Most participants used one hand on or above the surface, moving the map in the desired direction. The majority used multiple fingers, while some used the index finger or the palm. One user also used the stylus pen to move the map.

Two participants also used both hands to gesture. These variations also occurred within the same participants. Figure 5-2 shows some examples of common panning gestures.

(a) (b) (c)

Figure 5-2: Panning gestures, (b) and (c) shows different gestures from the same participant

Rotation

There are two kinds of rotation: one is 2D rotation about the axis perpendicular to the tabletop surface; the other is 3D tilting.

For 2D rotation, some people rotated their hands on or above the surface as if grabbing the map to rotate it; some people used two hands above or on surface to do a circular motion. A few people used the index finger or the stylus pen to draw a curve on the surface (see Figure 5-5).

(a) (b) (c)

Figure 5-3: Examples of 2D rotation gestures

For tilting, we observed the most variety of gestures mainly because there is no existing interface that allows users to do 3D gestures. As a result, participants are not preconditioned to any existing common gestures. Some participants used two hands in 3D space to make a circular motion (Figure 5-4(a)); some used one hand to do that (Figure 5-4(b)); some curled the fingers into a ball shape and rotated the hands (Figure 5-4(c)); some used one or two hands in 3D space to represent the tilted surface (Figure 5-4(d)); some participants used one or two hands to do a lifting action (Figure 5-4(e)). Some participants initially still had the mental model that the gesture needed to be touch-based, but then realized that it was hard to do that for the tilting gesture. Eventually they started to do some non-touch-based gestures. There were also a few participants who did touch-based gestures, for example, by putting two hands on the surface, moving in the opposite directions (Figure 5-4(f)) or the same direction. These gestures seem artificial because they do not resemble tilting in any ways. However, the participants still tried to make them distinct from other gestures. One participant resolved to give a speech command by speaking "angled view of 30 degree" to the system.

Zooming

The most common gesture for zooming is using two hands with index fingers or multiple fingers or palm moving towards or away from each other (Figure 5-5(a)). This is very much the same as the gesture used for zooming in the existing tabletop interfaces like Microsoft Surface. Some participants used an iPhone like gesture (a pinch gesture with thumb and index finger) (Figure 5-5(b)). Some put their hands on the surface, while some put hands above (Figure 5-5(c)). The choice of using touch or non-touch gestures was not consistent within the same user either. Some also used different gestures for zoom in and out.

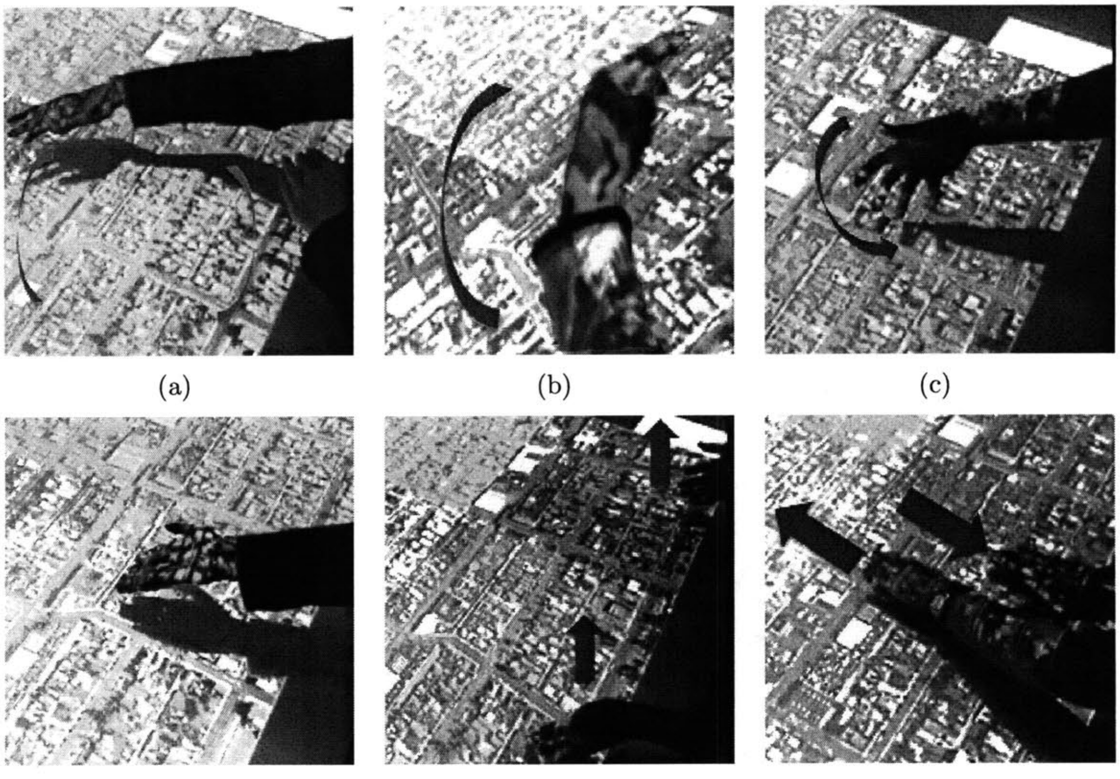


Figure 5-4: Examples of tilting gestures

Figure 5-5: Different zoom gestures used by the same participant: (a) zoom out; (b) zoom in; (c) zoom out

5.4.2 Communicative Gesture

Communicative gestures can be further classified into acts and symbols. This classification is suggested by Pavlović et al. [25] which is in turn based on Quek [26]'s proposed gesture taxonomy. In act gestures, the movements performed relate directly to the intended interpretation. Act gestures may be divided into two classes: mimetic and deictic [26]. Mimetic gestures are characterized by their iconicity. They are usually performed as pantomimes of the referent [26]. Deictic (or pointing) gestures are of great interest for computer input, and are our focus of analysis, because they are the most common gesture for our application. Symbol gestures are a kind of motion shorthand, serving in a linguistic role. Sign languages, for example, are largely symbolic.

Deictic Gestures

Deictics can refer to actual physical entities and locations, or to spaces that have previously been marked as relating to some concept and idea [6].

In our scenario, the deictic gestures were used for specifying a location, selection of an area and indicating a path.

For specifying a location, the most common gesture is circling the target with the index finger or the stylus pen. Other common gestures are pointing with the index finger, and tapping with multiple fingers. One participant also marked cross signs over places with the index finger. The gestures were always accompanied by speech such as "here", "inform this hospital and this one", "there is one shelter there", "this residential area go to this shelter", "shelter here, shelter there".

When pointing with the index finger, people did not alwasy touch the tabletop surface. For example, one participant said "there is 31st street" while pointing the index finger in a general direction without touching the surface.

For selecting an area and indicating a path, there were not many varieties in the gestures. All participants used the stylus pen or the index finger to make an outline of the area of interest or follow the lines of the path. Some participants also said "give me the shortest route to ..." when indicating the path.

Symbol Gestures

The only symbol gesture we observed was a "stop" gesture made by one participant when asking the interface to stop tilting further. The participant put two hands together to form a "T" shape to symbolize stop. The gesture is also accompanied with a speech command "stop". To support symbol gesture, we must recognize that they are usually arbitrary in nature [26] and are often culture specific. However, its occurrence with speech allows the opportunity for adaptive learning.

5.4.3 Speech and Gesture

Because of the inherent communicational purpose of the communicative gesture, we observed that when there were two participants working together collaboratively, there were more communicative gestures accompanied by speech. In the single participant case, there were a few participants who did not use any speech commands. However, the two-user case resembles the real world command and control center more closely where there are many people working together.

One participant used speech for most of the tasks. For example, for locating a place, the participant said "show me the intersection of Commercial Street and 29th street". The other commands used were "zoom in/out", "look straight down", "show me the radius of 3 street blocks from here". Notice that instead of saying the actual location of the chemical plant, she substituted that with a pointing gesture and the adverb "here". This participant also had used a speech-enabled interface a few times in the past.

Occasionally, manipulative gestures are also accompanied with speech. Some used speech commands such as "pan left", "rotate clockwise", "zoom in", or "tilt up" while they were gesturing. Some participants said "smaller" when gesturing zooming in. Some said "faster" or "more" when doing the tilting gesture. These words are adjectives or adverbs to describe the actions.

5.5 Discussion

We summarize some of the main observations that will be useful for designing a natural multimodal interface.

1. There are variations in people's gestures, some are small while some are significant. Prior experience with multi-touch devices such as iPhone or Microsoft Surface influence the gestures people use and make the gestures more uniform. Within a single user, the gestures may change for the same action. This is understandable because there are more than one natural way to perform curtain tasks. This demands great flexibility when developing an interface that can accommodate such variations. It may not be possible or necessary to accommodate all possibilities, however, the system can be bootstrapped with several common gestures, and then continue to learn new variations adaptively while in use. The system should also be easily customizable. For example, the user can give some examples of the gestures at the beginning, and the system can learn the gestures defined by the user.

2. People use one hand or two hands to gesture interchangeably, so it is important that the system support bimanual input. For the same task, for example zooming out or rotation, the same user may use different gestures either on or above the surface. Hence, supporting non-touch based gesture input will greatly improve the flexibility and naturalness of the interface. Some users used their right hand to hold the stylus pen and left hand to gesture, while some used the same hand to hold the stylus and gesture. This means both hand gesture and stylus can be incorporated as input, and people will naturally select the modality that suits the task best. The stylus pen is useful for high resolution input, while finger tips can be more convenient for tasks that do not require high resolution.

3. Users may speak and the gesture at the same time. As mentioned by some previous works [8][32][15], we can utilize the co-occurrence of the two modalities

for mutual disambiguation. The co-occurrence can alos be used for adaptive learning. For example, if the system recognizes the speech command "stop" and does not recognize the gesture, it can then associate the the gesture with the "stop" action.

4. Sometimes manipulative gestures are accompanied by adjectives and adverbs that describe the actions. This shows that speech not only can reinforce the gesture, but also can augment the gestures when there is a limitation in the expressiveness of the gesture or in the physical space of the gesture. The physical space can either be limited by the working volume of the system or human anatomy. Schlattmann and Klein [31] mentioned about the problem of limited work space for their bimanual 3D gesture interaction system. If a virtual object needs to be moved through a long distance, the hands may move outside the work volume, i.e., out of the view of the cameras. When this occurs, they suggested the user decrease the distance between the hands. An example limitation in human anatomy is that when doing a tilting gesture, there is a limit in the degree the hand can rotate. One way to overcome this is to repeat the gesture. This is analogous to repeating the mouse actions, for instance, repeating the scrolling action for zooming. However, a simpler way is to use speech, which we observed in our study: people naturally use modifiers like "more" or "smaller" to specify actions.

5. Speech commands people use tend to be succinct and may not be grammatically correct or complete. Many previous systems require users to conform to a predefined syntax such as "put [point] that [point] there" [3] or "build <object> here [point]" [36]. While we observed certain high frequency words such as "here" and "there" that are often accompanied by pointing gestures, there are also a variety of ways people express the commands. One common pattern is that people will mention the name of the target of interest, for instance, "hospital" or "shelter" and words for location. Sharma et al. [32] used a keyword spotting algorithm to combine speech and gesture, and was able to improve

the continuous gesture recognition rate. The key-word spotting algorithm allows a more flexible way to understand speech, and we plan to explore these in our future work.

6. Using gestures and speech commands allows us to move away from the WIMP paradigm because we need fewer controls to complete the tasks. This allows us to design more natural and intuitive user interfaces. During the study, only one participant wanted to have menus and buttons to click to do tilting and zooming.

Chapter 6

Conclusion

We have described the development of a gesture recognition module that is a necessary part of an overall multimodal system, and an exploratory user study for analyzing natural behaviours with a large tabletop display under USAR scenarios. In this chapter, we discuss the lessons learned and the direction for future work.

6.1 Lessons Learned

This thesis work demonstrates one possible valuable application of the hand-tracking system developed by Wang and Popović [38]. Despite the complex lighting and background, we can obtain good tracking results by reducing the adverse effects computationally.

We have also shown that background removal in a projector-based system can be done computationally and relatively accurately. The advantage of this method is that it is purely software-based, and hence, does not require additional hardware setup.

We obtained 96% recognition rate on isolated gestures for different users using data from the new hand-tracking system. To date, we achieved 74% correct rate on continuous gesture recognition with data from 3 different users and 12 gesture classes. Continuous gesture recognition is more difficult because we need to detect the start and end points of the gesture accurately. The ambiguities in the gestures, for example, whether a hand movement is part of the gesture or is just a retraction, create more challenges to the problem. We need to include different retraction cases in our training examples.

We obtained many insights through our user study for multimodal interaction under a natural condition. We observed a need for 3D gestures specially when interacting with 3D data. This shows the usefulness of a gesture recognition system that goes beyond a touch-based system. The co-occurrence of gesture and speech also provides possibility to further improve the continuous gesture recognition by modality fusion and mutual disambiguation.

6.2 Future Work

Our user study showed that speech in natural interaction is highly unstructured and rarely grammatical. We plan to use an off-the-shelf speech recognizer for key-word spotting [32]. Unlike the design of some previous multimodal systems, which restrict speech to short, specific verbal commands, we want to be able to deal with the informal, ungrammatical, and largely unrestricted speech we expect people to use spontaneously while performing this task. Given current speech understanding technology, the only plausible way to deal with this is to explore keyword spotting, relying on research (e.g., Cohen et al. [7]) showing that gestures and relevant classes of words (e.g., nouns) often appear contemporaneously, and repeated gestures often indicate coreference [12].

We plan to support bi-manual input because our study shows that people will naturally use both hands for interaction. Currently the hand tracker can not obtain the height of the hand relative to the table very accurately. We plan to improve the accuracy by stereo imaging so that we can enable a variety of touch sensitivity. This will permit a rich interaction environment with different input modalities: high resolution input from the stylus on the digitizer table, low resolution input from the finger tips, 3D gestures, and speech.

Once the sketch, gesture, and speech capabilities are operational, we will measure the impact of these on human efficiency and performance on the task. With a preliminary working prototype, we can use more quantitative metrics to measure the performance of a natural multimodal interface, which, as mentioned in the introduction, includes learnability, flexibility, memorability, convenience and efficiency.

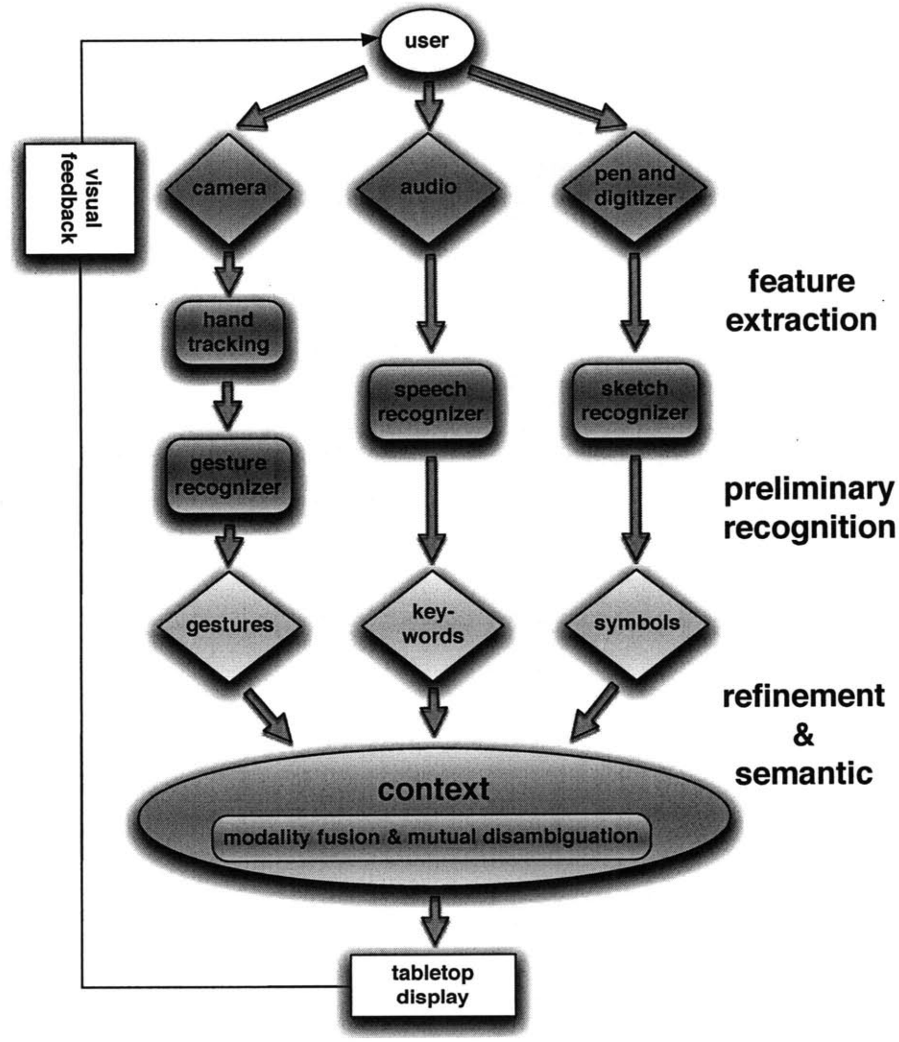


Figure 6-1: System architecture for the multimodal interface prototype

The longer term vision of our research involves building a multimodal interface for natural interaction that can serve as a testbed for conducting user studies to understand the human factors (human capabilities, e.g., attention, use of multiple information channels) involved in multimodal HCI. Based on the results of studies, we want to formulate a more principled system design framework for natural multimodal human-computer interaction. The modalities we will consider include hand gesture, speech, and pen-based input (i.e., sketches). Figure 6-1 shows the system architecture for developing the multimodal interface prototype.

Appendix A

User Study Tasks

Waiting for reports ...

1. Report: There is a big explosion at the chemical plant at the intersection of Commercial Street and 29th Street.

   Task: Locate the chemical plant on the map.

2. Task: Get an angled view of the chemical plant area.

3. Task: Zoom in and take a look around the surroundings at different angles.

4. Task: Zoom out and get back to the initial view looking straight down.

5. Report: The liquid chemicals are flowing steadily towards the surrounding residential areas. Radius of about 3 street blocks from the chemical plant is considered at high risk..

   Task: Indicate this high risk zone surrounding the chemical plant on the map.

6. Report: Local hospitals and shelter facilities need to be informed about the incident.

   Task: Indicate the closest two hospitals and shelters.

7. Report: The leaked toxic chemicals from the explosion cannot be prevented from moving into the residential area. People need to be evacuated in time.

Task: Work out some routes for evacuating people to the nearest hospitals and shelters. Watch out for places that are blocked due to damaged roads.

Bibliography

[1] M. Ashdown and P. Robinson. Escritoire: A personal projected display. *IEEE Multimedia*, 12(1):34–42, 2005.

[2] B. Bauer and H. Hienz. Relevant features for video-based continuous sign language recognition. In *FG '00: Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition 2000*, page 440, Washington, DC, USA, 2000. IEEE Computer Society.

[3] R. A. Bolt. "Put-That-There": Voice and gesture at the graphics interface. In *SIGGRAPH '80: Proceedings of the 7th annual conference on Computer graphics and interactive techniques*, pages 262–270, New York, NY, USA, 1980. ACM.

[4] I. Brewer. Lessons learned from a work domain analysis for the design of a collaborative, multimodal emergency management gis. presented at the GIScience Conf., 2002.

[5] S. K. Card, T. P. Moran, and A. Newell. *The Psychology of Human-Computer Interaction*, page 3. Lawrence Erlbaum Associates, 1st edition, February 1983.

[6] J. Cassell. A framework for gesture generation and interpretation. In *Computer Vision in Human-Machine Interaction*, pages 191–215. Cambridge University Press, 1998.

[7] P. Cohen, D. McGee, and J. Clow. The efficiency of multimodal interaction for a map-based task. In *Proceedings of the sixth conference on Applied natural language processing*, pages 331–338. Association for Computational Linguistics, 2000.

[8] P. R. Cohen, M. Johnston, D. McGee, S. Oviatt, J. Pittman, I. Smith, L. Chen, and J. Clow. Quickset: multimodal interaction for distributed applications. In *MULTIMEDIA '97: Proceedings of the fifth ACM international conference on Multimedia*, pages 31–40, New York, NY, USA, 1997. ACM.

[9] B. Dasarathy. Sensor fusion potential exploitation: Innovative architectures and illustrative applications. 85(1):24–38, January 1997.

[10] D. Demirdjian, T. Ko, and T. Darrell. Untethered gesture acquisition and recognition for virtual world manipulation, 2003.

[11] S. Eickeler, A. Kosmala, and G. Rigoll. Hidden markov model based continuous online gesture recognition. In *In Int. Conference on Pattern Recognition (ICPR)*, pages 1206–1208, 1998.

[12] J. Eisenstein. *Gesture in automatic discourse processing.* PhD thesis, Massachusetts Institute of Technology, 2008.

[13] W. Freeman and C. Weissman. Television control by hand gestures. In *Proc. of Intl. Workshop on Automatic Face and Gesture Recognition*, pages 179–183, 1995.

[14] S. Izadi, S. Hodges, S. Taylor, D. Rosenfeld, N. Villar, A. Butler, and J. Westhues. Going beyond the display: a surface technology with an electronically switchable diffuser. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*, pages 269–278. ACM, 2008.

[15] E. Kaiser, A. Olwal, D. McGee, H. Benko, A. Corradini, X. Li, P. Cohen, and S. Feiner. Mutual disambiguation of 3d multimodal interaction in augmented and virtual reality. In *ICMI '03: Proceedings of the 5th international conference on Multimodal interfaces*, pages 12–19, New York, NY, USA, 2003. ACM.

[16] A. Kendon. Current issues in the study of gesture. In P. P. J.-L. Nespoulous and A. R. Lecours, editors, *The Biological Foundations of Gestures: Motor and Semiotic Aspects*, pages 23–47. Lawrence Erlbaum Assoc., New Jersey, 1986.

[17] S. Kettebekov and R. Sharma. Toward natural gesture/speech control of a large display. In *Lecture Notes in Computer Science*, pages 133–146. Springer Verlag, 2001.

[18] N. Krahnstoever, S. Kettebekov, M. Yeasin, and R. Sharma. A real-time framework for natural multimodal interaction with large screen displays. In *Proceedings of the 4th IEEE International Conference on Multimodal Interfaces-Volume 00*, page 349. IEEE Computer Society, 2002.

[19] S. Kumar, P. R. Cohen, and R. Coulston. Multimodal interaction under exerted conditions in a natural field setting. In *ICMI '04: Proceedings of the 6th international conference on Multimodal interfaces*, pages 227–234, New York, NY, USA, 2004. ACM.

[20] D. McNeil and E. Levy. Conceptual representations of in language activity and gesture. In J. Jarvella and W. Klein, editors, *Speech, place and action: Studies in deixis and related topics.* Wiley, 1982.

[21] M. Micire, M. Desai, A. Courtemanche, K. M. Tsui, and H. A. Yanco. Analysis of natural gestures for controlling robot teams on multi-touch tabletop surfaces. In *ITS '09: Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, pages 41–48, New York, NY, USA, 2009. ACM.

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

[23] S. Oviatt. Multimodal interfaces for dynamic interactive maps. In *CHI '96: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 95–102, New York, NY, USA, 1996. ACM.

[24] S. Oviatt. Ten myths of multimodal interaction. *Communications of the ACM*, 42(11):74–81, 1999.

[25] V. I. Pavlović, R. Sharma, and T. S. Huang. Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:677–695, 1997.

[26] F. Quek. Toward a vision-based hand gesture interface. In *Virtual reality software & technology: proceedings of the VRST'94 Conference, 23-26 August 1994, Singapore*, page 17. World Scientific Pub Co Inc, 1994.

[27] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. pages 267–296, 1990.

[28] L. R. Rabiner and B. H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, pages 4–15, January 1986.

[29] I. Rauschert, P. Agrawal, I. R. Pyush, and R. Sharma. Designing a human-centered, multimodal GIS interface to support emergency management, 2002.

[30] D. Salber and J. Coutaz. Applying the wizard of oz technique to the study of multimodal systems. In *In Human Computer Interaction, 3rd International Conference EWHCI93*, pages 219–230. Springer Verlag, 1993.

[31] M. Schlattmann and R. Klein. Efficient bimanual symmetric 3d manipulation for markerless hand-tracking. In *Virtual Reality International Conference (VRIC)*, Apr. 2009.

[32] R. Sharma, J. Cai, S. Chakravarthy, I. Poddar, and Y. Sethi. Exploiting speech/gesture co-occurrence for improving continuous gesture recognition in weather narration. *Automatic Face and Gesture Recognition, IEEE International Conference on*, 0:422, 2000.

[33] R. Sharma, M. Yeasin, N. Krahnstoever, I. Rauschert, G. Cai, I. Brewer, A. Maceachren, and K. Sengupta. Speech-gesture driven multimodal interfaces for crisis management, 2003.

[34] M. C. Shin, L. V. Tsap, and D. B. Goldgof. Gesture recognition using bezier curves for visualization navigation from registered 3-d data. *Pattern Recognition*, 37(5):1011 – 1024, 2004.

[35] T. Starner and A. Pentl. Visual recognition of american sign language using hidden markov models. In *In International Workshop on Automatic Face and Gesture Recognition*, pages 189–194, 1995.

[36] E. Tse, S. Greenberg, C. Shen, and C. Forlines. Multimodal multiplayer tabletop gaming. *Comput. Entertain.*, 5(2):12, 2007.

[37] U. von Agris, D. Schneider, J. Zieren, and K. Kraiss. Rapid signer adaptation for isolated sign language recognition. In *Computer Vision and Pattern Recognition Workshop, 2006 Conference on*, pages 159–159, 2006.

[38] R. Y. Wang and J. Popović. Real-time hand-tracking with a color glove. *ACM Transactions on Graphics*, 28(3), 2009.

[39] A. Wilson. Simulating Grasping Behavior on an Imaging Interactive Surface. In *Proceedings of Interactive Tabletops and Surfaces 2009*, 2009.

[40] S. Young, D. Kershaw, J. Odell, D. Ollason, V. Valtchev, and P. Woodland. *The HTK Book Version 3.2.1.* Cambridge University Press, 2002.