Toward Natural Interaction in the Real World: Real-time Gesture Recognition

Ying Yin
MIT CSAIL
32 Vassar St, Cambridge MA, 02139 USA
yingyin@csail.mit.edu

Randall Davis
MIT CSAIL
32 Vassar St, Cambridge MA, 02139 USA
davis@csail.mit.edu

ABSTRACT

Using a new technology capable of tracking 3D hand postures in real-time, we developed a recognition system for continuous natural gestures. By natural gestures we mean those encountered in spontaneous interaction, rather than a set of artificial gestures chosen to simplify recognition. To date we have achieved 95.6% accuracy on isolated gesture recognition, and 73% recognition rate on continuous gesture recognition, with data from 3 users and twelve gesture classes. We connected our gesture recognition system to Google Earth, enabling real time gestural control of a 3D map. We describe the challenges of signal accuracy and signal interpretation presented by working in a real-world environment, and detail how we overcame them.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*input devices and strategies, interaction styles*

General Terms

Human Factors

Keywords

Multimodal interaction, natural human computer interaction, gesture recognition, interactive maps, digital table interaction

1. INTRODUCTION

In pursuit of more natural modes of human-computer interaction, we developed a new interface that enables real-time natural gestural interaction with a tabletop display, in a relatively uncontrolled environment. We discuss the difficulties presented by the environment and the steps needed to overcome them, and describe a system that goes beyond current practice by recognizing gestures from hand pose, as well as the traditional hand position and trajectory, potentially enabling a finer degree of gesture distinction.

ICMI-MLMI '10, In Proceedings of International Conference on Multimodal Interfaces and the Workshop on Machine Learning for Multimodal Interaction, pages 15:1–15:8, 2010.

2. NATURAL INTERACTION FOR USAR

While we believe that our work is broadly applicable, our current motivating task is urban search and rescue (USAR), which involves coordinating geographically distributed teams working under time pressure to search buildings and aid victims in the aftermath of a disaster. The task is both of considerable pragmatic utility and interesting as a research effort because it requires strategic assessment of large volumes of geographically-indexed information.

Computational tools currently available in USAR command centers are typically WIMP-based (window, icon, menu, pointing device), difficult to use, require substantial training, and may actually impede teamwork [21]. Interfaces that more closely conform to the way people naturally interact would have the potential to lower the users' cognitive load, allowing them to concentrate on the decision-making task.

3. USER STUDY

We did an exploratory user study to observe natural behavior under a USAR scenario, using a large tabletop display (described below). We chose a Wizard-of-Oz study in order to provide a realistic environment for eliciting natural multimodal input including speech, hand, and pen-based gesture. We summarize some of the main observations we made that are useful for designing a natural multimodal interface.

We found, not surprisingly, that there are variations in people's gestures. A single user may use different gestures for the same task, e.g., by gesturing either on or above the surface, using one hand or two hands, etc.

We observed users speaking and gesturing at the same time. As noted in previous work [6][20][12], the co-occurrence of the two modalities can be used to do mutual disambiguation.

Somewhat more interestingly, manipulative gestures are at times accompanied by adjectives and adverbs that refine the actions. As one example, we observed people naturally using terms like "more" or "smaller" as modifiers of their gestures. This shows how speech can naturally and usefully augment gestures when there is a limitation in the expressiveness of the gesture or in the physical space of the gesture.

Speech commands people use tend to be succinct and may not be grammatically correct. Many previous systems require users to conform to a predefined syntax (e.g., "put [point] that [point] there" [4]). 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 commands, reinforcing the need for natural, i.e., reasonably unconstrained interaction.

As the study made clear, the long-term solution requires a multimodal system capable of handling natural gestures accompanied by relatively unstructured speech.

While we have done extensive work on understanding hand-drawn sketches [16], sketching plus speech [1] and a variety of other multimodal systems, we focus in this paper on gesturing, as our first step toward a more complete system. As work in [17] has shown, gesture is particularly effective when dealing with spatial information, so it is well suited as a central modality of interaction for the USAR task.

4. HARDWARE

Large displays are an obvious choice for tasks like USAR that involve spatial information. In keeping with this, we constructed an interactive tabletop display (Fig. 1), modeled after the work in [2]. It uses four 1280 × 1024 pixel projectors connected to two NVIDIA GeForce 8600GT dual-headed graphics cards on a Dual-Core 2.4GHz desktop PC with 2GB of RAM, producing a seamless 2560 × 2048 image.

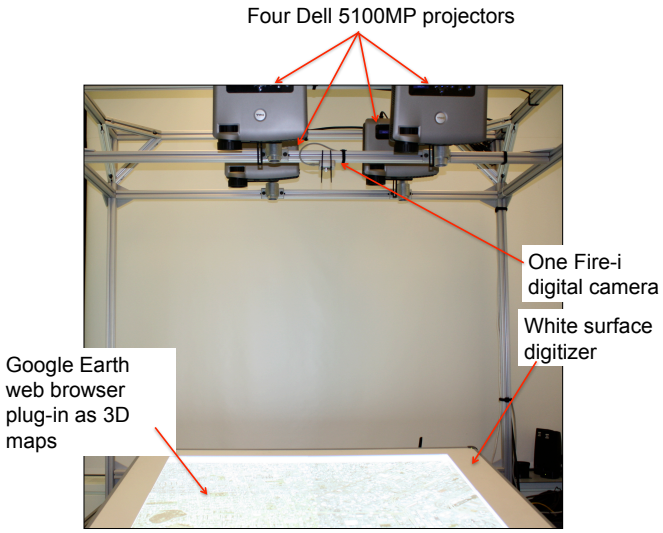


Figure 1: System setup

The image is projected onto a flat white surface digitizer (GTCO Calcomp DrawingBoard V), enabling us to capture writing and drawing on the image. A Fire-i™ camera (an ordinary off-the-shelf webcam) mounted above the center of the tabletop at the level of the projectors provides the view of the user interacting with the tabletop. Google Earth provides the 3D map displayed by the system.

4.1 Hand Tracking for Gesture

Accurate gesture understanding requires reliable, real-time hand position and pose data. Some previous efforts have used magnetic trackers and cybergloves, but these require wearing restrictive devices. Other approaches have relied on vision, but typically track only hand movement and finger tips, rather than 3D hand postures [7][19][15]. This limited data in turn often requires that artificial gestures be defined, in order to make position and pose tracking practical.

As our emphasis is on natural interaction, we want to be able to recognize gestures that people make spontaneously, and want minimal equipment on the user's hand.

We move toward this goal by using the system developed by Wang and Popović [25], which requires only an ordinary web camera and a cotton glove imprinted with a custom pattern (Fig. 2), and can track 3D hand postures in real-time. It provides data with 26 degrees of freedom: six for the hand position and four per finger. The glove is lightweight cotton, with no electronics or wires, opening up the possibility of developing more natural gestural interaction.

Figure 2: The colored glove

Given a video of the hand, Wang's system classifies each pixel in a frame as either background or one of the ten glove colors, using Gaussian mixture models trained from a set of hand-labeled images. The system then uses mean-shift with a uniform kernel of variable-bandwidth to crop the glove region. The region is size-normalized and the resultant image is used to look up the nearest neighbors in a large database of synthesized hand poses (see [25] for details).

We developed software to interface the tracker directly to the Fire-i camera, allowing the tracker to capture 640 × 480 RGB video at 15Hz with minimum latency.

4.2 Background Removal

Wang's system was developed originally for use in an office environment, i.e., a place with diffuse white illumination and plain backgrounds (e.g., a beige desk), where it can rely on the assumption that the glove is the only thing in the image with that particular set of bright colors.

Our tabletop environment, by contrast, with its projected, changing map, has both a complex and dynamic background, and non-uniform illumination. As the accuracy of the tracking relies on the accuracy of color classification of the glove image, our environment poses considerable additional challenges for Wang's system. Fig. 3 shows the camera's view of the table and user's hand. The complex background and illumination result in errors in color classification when the map contains colors close enough to those on the glove.

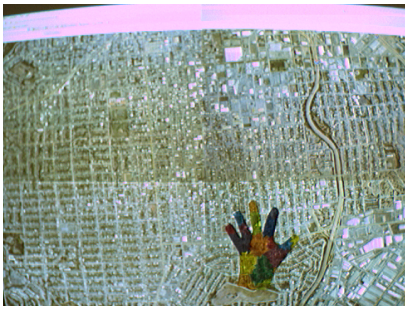


Figure 3: Camera image with complex background

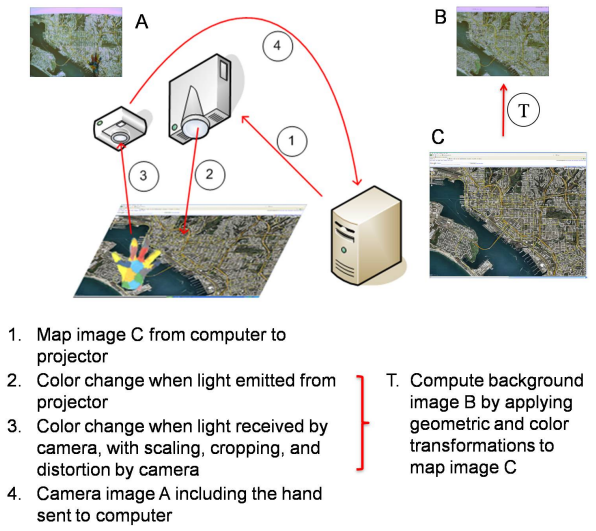


Figure 4: Transformation of the image pixels

Traditional background subtraction involves taking an image containing only the background or keeping a running average (a linear combination of the current image and the running average image [9]), then subtracting the background from the current image.

We have the advantage that the background is the map being displayed, so we know what the background is supposed to be. Unfortunately, what the Fire-i camera sees (I_C) is not what the graphics card is sending (I_G). Fig. 4 shows the physical processes that modify the pixels from the graphics card as they are projected to the tabletop, then captured by the camera. To determine what the image background looks like to the Fire-i, we need to characterize each of the processes in Fig. 4 through calibration.

4.2.1 *Geometric Calibration and Transformation*

Two steps of geometric calibration and transformation are needed. The first takes account of the barrel distortion produced by the Fire-i. We used the Camera Calibration Toolbox for Matlab[1] to get the intrinsic parameters of the camera, allowing us to remove the distortion in the camera image computationally. For real-time performance during tracking, we generate a lookup table that maps the x, y-coordinates between the distorted and rectified images.

The second step calibrates the degree to which the camera is rotated and/or translated relative to the tabletop display. We do this by projecting a checkerboard image (the target) on the tabletop and recording the x,y-coordinates of the grid corners of the target. We take a picture of target with the Fire-i, rectify the image, and record the corresponding coordinates of the grid corners. Using homogeneous coordinates, we find the least error transformation that maps the tabletop display coordinates to the camera image coordinates.

4.2.2 *Color Calibration and Transformation*

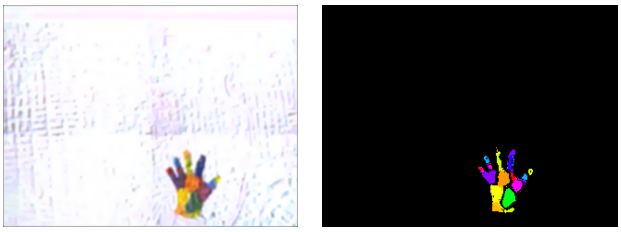
Next we need to understand how the colors in the map image (I_G) actually look to the Fire-i. Let $\underline{c}_p$ be the color projected and $\underline{c}_c$ be the color captured by the camera. We want to find a function $\boldsymbol{f} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ where $\boldsymbol{f} = [f_r, f_g, f_b]$ such that $\underline{c}_c = \boldsymbol{f}(\underline{c}_p)$.

We find $\boldsymbol{f}(\underline{c}_p)$ by using a color calibration palette with 80 colors, chosen because they are the colors found most frequently in Google Earth. The original palette and the color palette as seen by the camera supply a set of training examples $(\underline{c}_p^i, \underline{c}_c^i)$ for $i = 1 \ldots 80$. We explored a number of different models for the function, finding that the most effective model was a second degree polynomial in which each color channel also depends on all three channels. If x is one of the color channels, i.e., $x \in \{r, g, b\}$, $f_x(\underline{c}_p) = \underline{\theta}_x \cdot \underline{\phi}(\underline{c}_p)$, where $\underline{\theta}_x$ is the parameter vector and $\underline{\phi}(\underline{c}_p)$ 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 $\underline{\theta}_x$. The regularization term $\|\underline{\theta}_x\|^2$ is added to prevent over-fitting.

4.2.3 *Removing the Effect of Illumination*

Finally, because light from the projector illuminates the hand, the color signal reaching the camera from the hand is the multiplication of illumination and reflectance of the glove color for each pixel. To eliminate the effect of the illumination, we rectify the camera image (removing the distortion noted above), then divide it pixel-by-pixel by the transformed image from the graphics card (i.e., the background as seen by the camera).

Figure 5(a) shows an example of the result after this process, 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 removal (Fig. 5(b)) is considerably more robust.



(a) (b)

Figure 5: (a) Resultant image after background removal; (b) result of color classification after background removal

5. GESTURE RECOGNITION

The hand tracking system described above provides detailed 3D hand pose data. Our next step is to use that data to infer gestures.

5.1 Gesture Taxonomy

We adopt the taxonomy of hand movements proposed by [18], distinguishing gestures from unintentional hand movements (like beats), and further dividing gestures into manipulative and communicative. Manipulative gestures are used to act on objects, while communicative gestures have an inherent communicational purpose [18]. Our work to date has focused on recognizing single-hand manipulative gestures.

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

5.2 Isolated Gesture Recognition

We start by describing our approach to isolated gesture recognition, and then detail how we recognize continuous gestures.

5.2.1 Feature Vector

For tracking one hand, the tracker supplies data describing the hand location in x, y, z coordinates, and orientations of the hand and each finger joint in quaternions. There are three joints per finger in the model; the joint at the base of each finger has two degrees of freedom (DOFs) while the other two joints of each finger have one DOF each. 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 (except in painful circumstances) stays unchanged.

Our feature vector is composed of the velocity of hand in the x-y plane, the z position of the hand, the 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 data are also standardized across each feature to have a mean value of 0 and a standard deviation of 1. This ensures that the components of the feature vector have compatible magnitudes.

The feature vector produces a detailed description of the hand motion and pose, and as such provides more generality than would be available had we selected only those features that discriminated among a set of predetermined gestures. This approach gives us the flexibility to investigate and train the gestures that our user studies demonstrate are commonly employed, rather than being restricted to a predetermined set of gestures.

5.2.2 Hidden Markov Models

Previous work has suggested that the position and orientation of hand movement follows a first order Markov process [23]. Accordingly, we use a Hidden Markov Model (HMM) as our classification machinery, in particular a variety called the Bakis model, which is well suited to time-series signals [3]. 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 us to compensate for different gesture speeds [3].

Figure 6 shows an example of a four-state Bakis model with the transition probability from state s' to state s as $t(s|s')$ for $s, s' \in \{1, 2, \ldots, m\}$ where m is the number of states. We also add the non-emitting entry and exit states to the model, similar to what Young et al. [26] did for their speech recognition system. The entry state is added for easy representation of the initial state probability $t(s)$. 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 $P(\underline{x}_1, ...\underline{x}_t; \underline{\theta}_k)$ will be evaluated for all of the gesture models, with the classification 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:

$$\hat{k} = \arg\max_k \log P(\underline{x}_1, ...\underline{x}_t; \underline{\theta}_k). \quad (1)$$

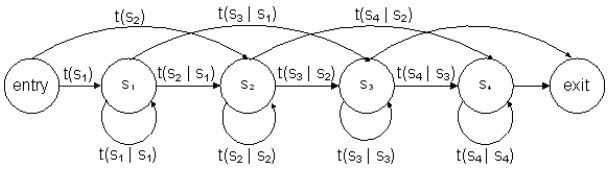


Figure 6: The state transition diagram of a four-state Bakis model with corresponding transition and initial state probabilities

5.3 Model Selection

5.3.1 Emission Probabilities

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

$$e(\underline{x} \mid s) = N(\underline{x}; \underline{\mu}_s, \Sigma_s). \quad (2)$$

We then generalize the method to accommodate the variance among multiple users by employing a Gaussian mixture model for each state. We assume 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 $z \in \{1 \ldots l\}$. Hence,

$$e(\underline{x} \mid s) = \sum_{z=1}^{l} q_s(z \mid s) N(\underline{x}; \underline{\mu}_{s,z}, \Sigma_{s,z}). \quad (3)$$

Note that Equation 2 is just a special case of Equation 3 when $l = 1$.

5.3.2 Model Size

Choice of model size (the number of states) is an important issue in implementing HMMs. An underlying state in our HMM represents a particular velocity, orientation and shape of the hand during a gesture. Hence, the states can be associated with the temporal phases that make up gestures. Psychological studies suggest that gestures have three phases: preparation, nucleus (peak or stroke [14]), 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" [13]. In the retraction phase, the hand either returns to the rest position or is repositioned for the next gesture. In keeping with this, our HMMs contain at least three (hidden) states [18].

Previous efforts have used from 4 [23] to 41 [24] states in their HMMS, a difference that may be due in part to variations in data sampling rates. While more states means a finer-grain discretization of the movement, which should lead to better accuracy, there is a trade-off between the complexity of the model and over-fitting the data. In addition, the number of states should also be related to the sampling rate (around 15 frame/sec in our case) which affects the number of frames per gesture. The smaller the number of frames per gesture, the smaller the number of states should be used.

We determined model size by training the system on 12 common gestures used for basic map manipulations, using

a MATLAB HMM toolbox[2] to get the maximum likelihood parameters of the models via expectation-maximization (EM). The gestures used were 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. Zooming in is spreading the fingers outwards; zooming out is the reverse action. Fig. 7 shows some examples (only left hand is used to date in training and testing). As the feature vector we use describes hand pose in general, we believe our results will be generalizable for other gestures.

Figure 7: Examples of some gestures used in the experiment

We collected a single user data set X_S over three different days, with 9 samples for each of the 12 gestures. Three samples per gesture were set aside for testing, and the remaining 6 samples per gesture were used for training. We used cross-validation to determine the choice of model size, and found that using between 3 and 5 states gave the highest testing accuracy.

We collected a similar data set X_M from three users. We used 14 samples per gesture for training, and 4 samples per gesture for testing (48 test samples in total). We experimented with different values of l in Equation 3, using k-means clustering for initializing the parameter estimates for Gaussian mixtures. Our results indicated that using 3 mixtures of Gaussian and 4 states gives the highest recognition accuracy, 95.6%, on the test data. There are several sources of possible errors. Although the hand tracker can identify challenging hand postures with significant self-occlusion [25], when the hand is rotated (e.g., in the pitch gesture, where the plane of the palm is almost perpendicular to the camera), and only a small section of the colored patches is visible, both hand tracking and gesture recognition accuracy are lower. The variations in different users also contribute to recognition errors. We need to test the trained model with more different users in future work.

5.4 Continuous Gesture Recognition

Continuous gesture recognition requires segmentation: detecting start and end points in a continuous gesture sequence in order to know what segment of the tracking signal to classify. Some previous systems have required the use of a specified artificial movement to indicate the start of a gesture [22], or they distinguished gestures from unintentional hand movements by restricting hand motion. For a more natural interface that a user can interact with more naturally, we needed a more flexible approach.

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

5.4.1 Segmentation

Gesture is signified by motion. We compute $\underline{d}_t$, the difference between the consecutive input data. As the first two elements of the feature vector $\underline{x}$ are velocities (already representing the difference), we keep these two unchanged, and calculate the difference for the remaining elements in $\underline{x}$:

$$\underline{d}_t = \begin{bmatrix} \underline{x}_t(1:2) \\ \underline{x}_t(3:26) - \underline{x}_{t-1}(3:26) \end{bmatrix}, \quad \forall t \in \{1 \dots T\}.$$

Note that there is no d_0. We then take the l^2-norm of the difference vector, i.e., $\|\underline{d}_t\|$, and use a two-state HMM to model the start and the end of a gesture using $\|\underline{d}_t\|$ as the observation sequence.

Fig. 8 shows an example of the segmentation 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, and the blue segments are gesture intervals resulting from HMM segmentation. When attempting to provide the manual labels, we discovered that it is hard to pinpoint the exact frame when the gesture starts and ends. Hence there may be errors in our "ground truth". In any case the differences in timing between the HMM segmentation and our ground truth are small – about 5 to 10 frames, or 0.3 to 0.6 seconds, meaning that the system's results are quite close to the manual labels.

The fluctuations in $\|\underline{d}_t\|$ are partly due to the noise in data from the hand tracker: the z value in the feature vector is especially noisy because there is no stereo imaging. The variation in the gesture speed is another factor. Even so, using an HMM is more robust than simply calculating the average $\|\underline{d}_t\|$, and using that as a threshold for gesture intervals, because an HMM takes the transition probabilities into account, making it less sensitive to high-frequency fluctuations. However, there still can be false detections in our current system, as Figure 8 shows.

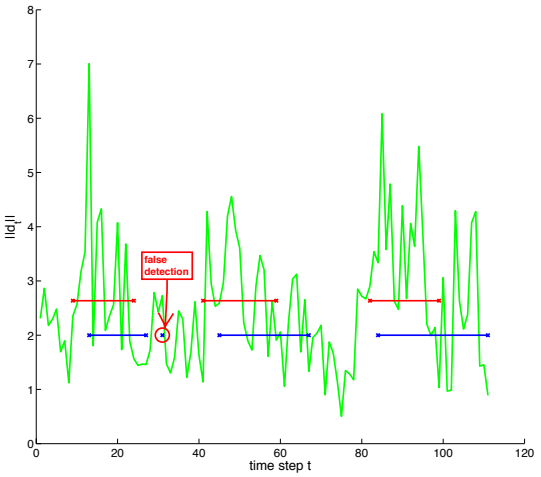


Figure 8: An example of gesture segmentation result

5.4.2 Real-Time Recognition

A final important issue is making recognition response fast enough to allow real-time interaction. Ideally we need

to recognize each 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 Eq. 1, we evaluate the feature vector at each time step, updating 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

$$P(x_{t_s}, \ldots, x_t; \underline{\theta}_k) = \sum_s P(x_{t_s}, \ldots, x_t, s_t = s; \underline{\theta}_k) = \sum_s \alpha[t, s]. \tag{4}$$

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

$$\alpha[t, s] = \sum_{s'} (\alpha[t-1, s'] \times t(s|s') \times e(x_t|s)). \tag{5}$$

As the probabilities become very small, our implementation uses logarithmic probabilities. Our approach is also efficient, requiring only an array to keep track of the α values and the log-likelihood value for each gesture model. As a result its 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 to the end of a gesture for the isolated gestures in the test data set X_M. As an empirical test, 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 often the highest even very early in the gesture. Figure 9 shows one such example: the actual gesture is pan down, and the log-likelihood of the observation sequence for pan down is the highest at the very beginning.

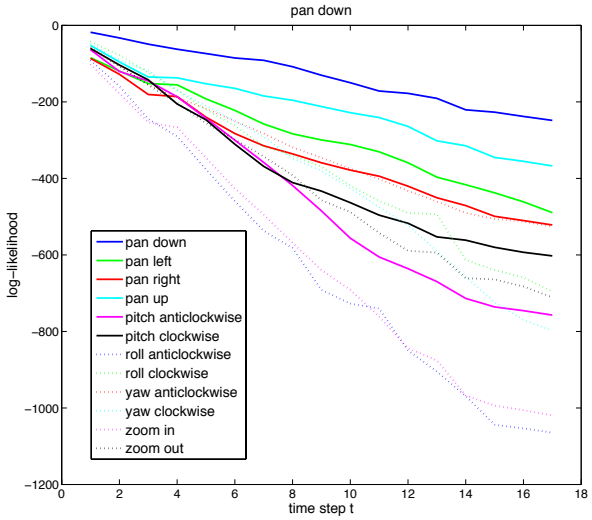


Figure 9: Change of log-likelihood values for 12 HMMs over time; the actual gesture is pan down

It is of course not always true that the correct gesture model will have the highest log-likelihood at the beginning: some gestures start out looking the same. Figure 10 is one such example: all the log-likelihood values are very close in the beginning.

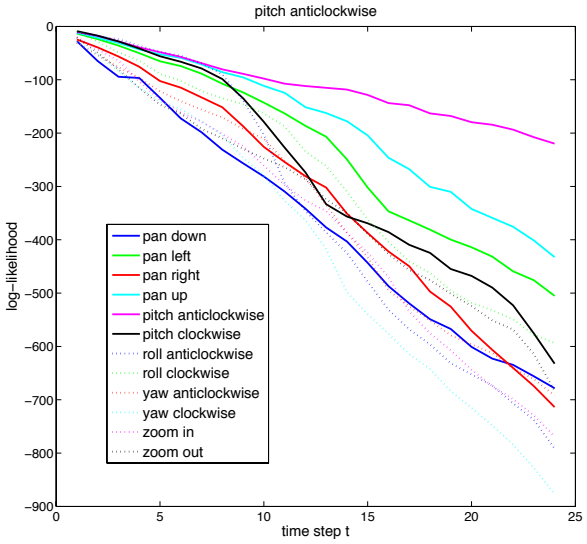


Figure 10: Change of log-likelihood values for 12 HMMs over time; the actual gesture is pitch anticlockwise

Our recognition decision is thus made only after the difference between the highest and second highest log-likelihood values are greater than an empirically derived threshold. We get the threshold by analyzing the test data in X_M, calculating the average maximum difference between the highest and second highest log-likelihood values when the higher one does not correspond to the correct gesture. Using this approach means we make a decision about the gesture as early as possible, i.e., as soon as it becomes clear which gesture is most likely.

5.4.3 *Experiments and Results*

We recorded a data set M_C containing sequences of continuous gestures, manually labeling the gestures, their start and end points. The test sequences contain different number of gestures, ranging from 2 to 6, with 49 gestures in total.

We measured the recognition rate – defined as the number of correctly recognized gestures over the number of actual gestures – without considering the false detections. 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 73%. If we consider the false detections of gestures, the rate is 60%. However, we observed that most false detections are very brief, typically about 2 – 3 time steps (i.e., 2 – 3 video frames, about 160 ms). As we explain below, such false detections have relatively little pragmatic consequence.

Figure 11 shows a graphical representation of the recognition process. It plots the log-likelihood values of 12 ges-

ture HMMs against time for a sequence of gestures. The log-likelihood values are set to 0 when there is no gesture detected. The upper horizontal line segments indicate the manually assigned 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 classification of the gestures is indicated by combinations of different line colors and line styles.

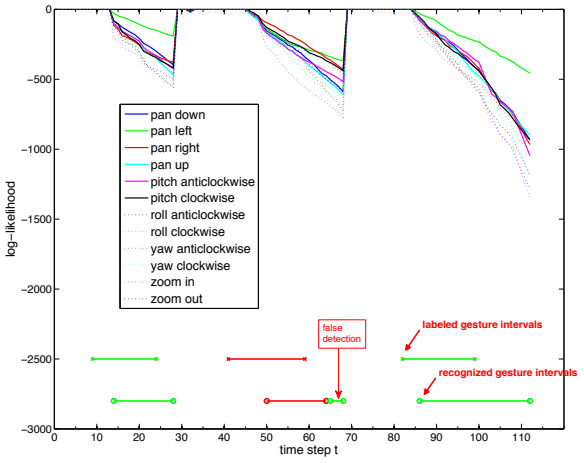


Figure 11: Change of log-likelihood values for 12 HMMs over time; continuous gestures

6. INTERACTING WITH GOOGLE EARTH

Using the Google Earth Plug-in and its JavaScript API, we embedded the Google Earth 3D map into a web page. We use the Java web browser object in the JDesktop Integration Components (JDIC) library to load the page. This allows us to augment the browser, i.e., make it respond to the gesture events from the gesture recognizer by providing 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. As noted, our continuous gesture recognition can have false detections. However, as long as the false detection period is short, it is unlikely to have a visible effect because the action the system will call for will be negligibly small.

The map is of course changed by each gesture, thereby changing the background displayed on the surface. In response, we re-compute the image needed for background removal.

7. RELATED WORK

There are other background removal methods for a projector-based system including, for example, polarizing filters or synchronized optical shutters [11]. As our method is purely software-based, it does not require additional hardware.

Much work on computer recognition of hand gestures has focused on synthetic gestures, most notably sign language recognition. Hernández-Rebollar et al. [10] used a glove-like device with accelerometers to recognize static postures of American Sign Language alphabets with a hierarchical classifier. Our research focuses more on gestures with dynamic change of postures. Bauer and Hienz [3] developed a video-based continuous sign language recognition system based on HMMs. They reported an accuracy of 91.7% based on a lexicon of 97 signs from a single user's input. While research in sign language recognition provides a good starting point on natural gesture recognition, there are some differences between the two. Sign language is well structured and has a defined grammar; natural gestures are free-form, and can occur in any order.

A variety of efforts have been made at 3D gesture. For example, Oblong Industries' g-speak spatial operating environment[3] allows free hand input in 3D space. Microsoft's Project Natal, a controller-free gaming and entertainment system, enables users to play video games using gestures by tracking full-body 3D motion. However, by using Wang's [25] hand pose tracking system, we can focus on a finer degree of 3D gesture distinction.

The work that is most related to ours is by Sharma et al. [21]. They did substantial work in analyzing the issues in designing speech-gesture driven multimodal interfaces for crisis management. They also did some work on continuous natural gesture recognition using HMMs [20]. Using their metrics, they reported a correct rate of 78.07% for three gesture categories: pointing, area and contour gestures with data from 6 different people. One main difference between their work and ours is that they tracked the hands as blobs, whereas we are recognizing finer degree gestures.

8. DISCUSSION AND FUTURE WORK

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

We defined the set of gestures used here for the purpose of testing the performance of the recognition method. We need to verify this performance with other sets of gestures, obtained through our user study, to test the generalizability of the system. We will also work on bimanual gestures in the future. First, we will test how sensitive the color-glove hand tracker is to two-hand occlusion, and take this into consideration when applying gesture recognition method.

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 (as in [20]). We want to be able to deal with the ungrammatical, and largely unrestricted speech we expect people to use spontaneously. Given current speech understanding technology, the plausible way to deal with this is to explore combining keyword spotting with gesture recognition, relying on research (e.g., [5]) showing that gestures and relevant classes of words (e.g., nouns) often appear contemporaneously, and repeated gestures often indicate coreference [8]. Some informal grammar, however, can also be incorporated

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

which may help inform the gesture recognition process.

9. ACKNOWLEDGEMENTS

We thank Robert Wang for assistance in understanding and using the hard-tracking system he developed.

This work was supported in part by a grant from Pfizer, Inc., by a grant from Foxconn Technology, and by Award IIS-1018055 from the National Science Foundation.

10. REFERENCES

[1] A. Adler and R. Davis. Symmetric multimodal interaction in a dynamic dialogue. In *2009 Intelligent User Interfaces Workshop on Sketch Recognition*. ACM Press, February 2009.

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

[3] 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, 2000.

[4] 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, 1980.

[5] 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.

[6] 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. ACM, 1997.

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

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

[9] 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.

[10] J. L. Hernández-Rebollar, R. W. Lindeman, and N. Kyriakopoulos. A multi-class pattern recognition system for practical finger spelling translation. In *ICMI '02: Proceedings of the 4th IEEE International Conference on Multimodal Interfaces*, page 185, 2002.

[11] 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.

[12] 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, 2003.

[13] 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.

[14] 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.

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

[16] T. Y. Ouyang and R. Davis. A visual approach to sketched symbol recognition. In *Proceedings of the 2009 International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1463–1468, 2009.

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

[18] 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.

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

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

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

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