Untethered Gesture Acquisition and Recognition for a Multimodal Conversational System

ABSTRACT

1. INTRODUCTION

By providing different input channels, multimodal interfaces allow a more natural and efficient interaction between user and machine. Recent years have seen the emergence of many systems using speech and gestures, where users interact with an application by talking to it, pointing (or looking) at icons and/or performing gestures. Research in multimodal interfaces and ubiquitous computing aim at building the tools to implement these abilities, in as natural and unobtrusive a manner as possible.

Several successful multimodal gesture systems have been developed which integrate speech input with pen and other haptic gestures [8, 18]; these generally use a physical stylus, or just a user's fingertip. For interaction with a kiosk [22], video wall [9], or conversational robots [3] it is desirable to have untethered tracking of full-body gesture. Full-body gesture processing consists of two components: *acquisition* to estimate the pose of the user (e.g. arm, body position and orientation) and *recognition* to recognize the gestures corresponding to sequences of poses.

To date, full-body gesture acquisition has been mainly developed around tethered interfaces because of their robustness and accuracy. Devices such as data gloves and magnetic position systems e.g. Flock of Birds have been successfully used for tasks such as map exploration [21]. Schemes with explicit markers attached to hands or fingers have also been proposed, as in systems for optical motion capture in computer animation. Unfortunately, the difficulty of use of these systems (e.g. attached wires, magnetic isolation of the room) prevents them from being generally usable by casual users. There has been many attempts to build untethered interfaces based on vision systems. However, to our knowledge, none of them has proven to be robust and fast enough to extract full articulated models for HCI purpose. In this paper we present an untethered interface based on the tracking of the user body using stereo cameras.

Many body pose gesture recognition systems use techniques adapted from speech recognition research such as Hidden Markov Models (HHMs) or Finite-State Transducers (FSTs). Such techniques consider consecutive poses of the user given by the acquisition system and estimates the most probable gesture. There are many difficulties with gestures. First, the inputs are highly dimensional (the dimension of body poses are usually greater than 20). Then the beginning and end of gestures are difficult to detect (contrary to speech where sentences are isolated by detecting surrounding silences).

In our multimodal system, speech is processed using the GALAXY system [26]. A vision system consisting of a stereo camera connected to a standard PC Pentium 4 (2Gz) captures images of the user and transfers them to the articulated body tracker that estimates the corresponding body pose (described in Section3.1). Sequences of body poses are used in the gesture recognition system to identify gestures (Section3.2. A rank order fusion algorithm is used to merge command recognition; parameters are estimated for each visual gestures (e.g., size or location.) Our overall architecture is shown in figure 1.

In the following sections we present the architecture of our multimodal system. Then we introduce our vision-based body pose acquisition technique. A framework for gesture recognition is then described. Finally we demonstrate its use for recognizing typical gestures and show some results.

2. PREVIOUS WORK

Many techniques for tracking people in image sequences have been developed in the computer vision community.

Techniques using cues such as contour and skin color detection have been popular for finger and hand tracking [15, 23, 16, 9] but are limited to planar interactions .

Articulated model-based techniques have been proposed to track people in monocular image sequences [31, 14, 20, 4, 32]. Due to the numerous ambiguities (usually caused by cluttered background or occlusions) that may arise while tracking people in monocular image sequences, multiple-hypothesis frameworks may be more suitable. Many researchers investigated stochastic optimization techniques such as particle filtering [27, 28]. Though promising, these approaches are not computationally efficient and real-time implementations are not yet available.

Stereo image-based techniques [19, 1] proved to give better pose estimates. [19] uses a generative mixture model to track body gestures with real-time stereo. However, the model used in this system was approximate and the system could only accurately detect arm configurations where the arm was fully extended.

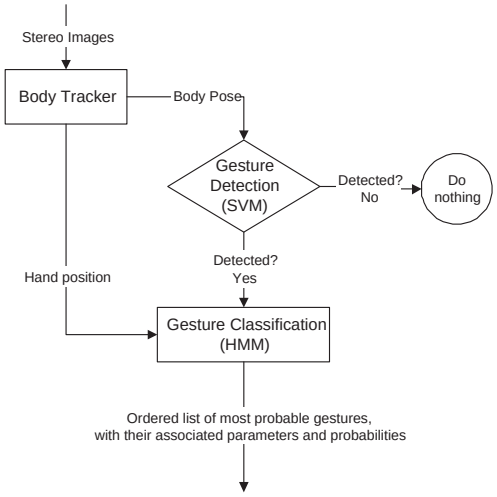

Figure 1: Vision Component.

The system described in [1] was successfully applied to detecting and classifying hand gestures in a conversational system [30, 6]. While the system was in real-time, it could sense only coarse "blob" features. Hence it was of limited use in tracking natural pointing gestures, although it was able to recognize parametric gestures defined by the relative position of both hands [30], using a variation of Hidden Markov Models. Other HMM approaches to gesture recognition include [17, 29].

Bobick and Davis [10] created a view-dependent approach to gesture recognition using temporal templates. It captures motion and time in one 2D image by varying the intensity of pixels where motion had occurred. Motion is captured by the change in the foreground image (the person) versus the background image. As time passes, the intensity of the pixels will decrease. This is then compared to the temporal templates to find the gesture that is most similar.

Previous systems using speech and gesture inputs [18, 8] have taken advantage of the progress of the research in the corresponding fields and the advent of new devices and sensors. Although some approaches such as [18] integrate speech and gesture at an early stage, most systems perform the recognition of speech and gesture separately and use a unification mechanism [8] for fusing the different modalities.

Speech processing usually consists of two components: *word recognition* to recognize individual words and *language understanding* to recognize full sentences (given a predefined grammar). Research in speech processing has known excellent progress in the last 10 years. Although there are still problems with crowds ("cocktail-party" problem) and noisy sound input (especially when the microphone is far from the speaker), natural speech recognition systems offer very high recognition rates.

3. ARTICULATED TRACKING AND GESTURE RECOGNITION

We propose a cascaded approach which efficiently and reliably recognizes a gesture. The system can be separated into two components, an articulated tracker and a gesture recognizer. We track articulated body motion and recognize full-body gestures of a user using a stereo-based technique. The body model used in this paper consists of a set of N rigid limbs linked with each other in a hierarchical system.

3.1 Tracking

Pose Π of a body is defined as the position and orientation of each of its N constituent limbs in a world coordinate system ($\Pi \in \mathcal{R}^{6N}$). We parameterize rigid motions using twists [4]. A twist $\boldsymbol{\xi}$ is defined as a 6-vector such that:

$$\boldsymbol{\xi} = \begin{pmatrix} \boldsymbol{t} \\ \boldsymbol{\omega} \end{pmatrix}$$

where t is a 3-vector representing the location of the rotation axis and translation along this axis. ω is a 3-vector pointing in the direction of the rotation axis.

Let Δ define a set of rigid transformations applied to a set of rigid objects. Δ is represented as a $6N$-vector such that:

$$\Delta = (\boldsymbol{\xi}_1, ..., \boldsymbol{\xi}_N)^\top \tag{1}$$

where N is the number of limbs in the body model.

Many algorithms have been proposed for articulated tracking using stereo data [11, 19, 12]. Such algorithms give an estimate of the body motion by minimizing an error function based on the distance between the 3D articulated model and the reconstructed 3D points.

In the case of articulated models, motions $\boldsymbol{\xi}_i$ are constrained by spherical joints. As a result, Δ only spans a manifold $\mathcal{A} \subset \mathcal{R}^{6N}$ that we will call *articulated motion space*. Efficient tracking is possible by enforcing the spherical joint constraints using a projection-based approach such as [12]. This technique consists in finding the motion $Delta^\star$ which minimize the Mahalanobis distance:

$$\begin{aligned} E^2(\Delta^\star) &= ||\Delta^\star - \Delta||^2_\Sigma \\ &= (\Delta^\star - \Delta)^\top \Sigma^{-1} (\Delta^\star - \Delta) \end{aligned} \tag{2}$$

with

$$\Delta = (\boldsymbol{\xi}_1, ..., \boldsymbol{\xi}_N)^\top \qquad \Sigma = diag(\Sigma_1, \Sigma_2, \dots)$$

where $\boldsymbol{\xi}_i$ is the rigid motion estimated by a 3D rigid object tracker (in our implementation, we used an ICP-based tracking algorithm) and Σ_i the corresponding uncertainty.

The approach presented in this paper can be considered as an extension of [12] accounting for non-linear constraints related to human body pose. Indeed, the human body is highly constrained due to various factors which are not possible to capture in a linear manifold (e.g., joint angles between limbs are bounded, some poses are unreachable due to body mechanics or behavior). To enforce these constraints we use a learning-based approach, and build a human body pose classifier using examples extracted from motion capture (mocap) data. We represent the space of valid poses defined by mocap data using a support vector machine (SVM) classifier.

SVMs classifiers have been very popular in the computer vision community for their ability to learn complex boundary between classes and also for their speed and efficiency. See [25, 5] for a detailed description of SVMs.

Given a data set $\{x_i, y_i\}$ of examples x_i with labels $y_i \in \{+1, -1\}$,

C	1	10	200	1000
ϵ	0.00072	0.00061	0.00065	0.00137
N_{sv}	1878	367	323	294

Table 1: Classification error rates ϵ and number N_{sv} of support vectors for SVMs trained with Gaussian kernels (σ=10) *vs.* error cost C.

σ	5	10	15	20	100
ϵ	0.00065	0.00061	0.00094	0.00047	0.0059
N_{sv}	479	367	570	842	4905

Table 2: Classification error rates ϵ and number N_{sv} of support vectors for SVMs trained with Gaussian kernels (C=100) *vs.* kernel size σ.

an SVM estimates a decision function $f(x)$ such that:

$$f(x) = \sum_i y_i \alpha_i k(x, x_i) + b \tag{3}$$

where b is a scalar and α_i some (non negative) weights estimated by the SVM. A subset only of the weights α_i are non null. Examples x_i corresponding to non zero α_i are the *support vectors*. The support vectors are the training examples that lie closest to the decision boundary. Their corresponding α_i defines its contribution to the shape of the boundary. $k(x, x_i)$ is the kernel function corresponding to the dot product of the non linear mapping of x and x_i in a (high dimensional) feature space. Linear, polynomial and Gaussian kernels are usually used. In this paper, we used a Gaussian kernel $k(x, x_i) = e^{-||x-x_i||^2/(2\sigma^2)}$.

In practice, an error cost C is introduced to account for outliers during the SVM training [25]. This allows for the noise in data that would cause classes to overlap. Once the SVM has been trained, new test vectors x are classified based on the sign of the function $f(x)$. In this work, we used the SVM implementation from the machine learning software library *Torch* [7].

3.1.1 Training

We trained a SVM classifier to model valid poses of human bodies. The features x used in the SVM are the relative orientation of the body with respect to the world coordinate system and the relative orientations of connected limbs.

Training data consisted of a collection of more than 200 mocap sequences of people walking, running, doing sports, *etc*, which amounts to about 150,000 body pose (positive) examples. The models used in these sequences describe the full body, including hands, fingers, and eyes. However, only the parameters used in our model (torso, arms, forearms and head) have been retained for the SVM training. Negative examples have been randomly generated. Because the space of valid poses is small compared to the space of all possible poses, a pose with randomly generated angles for each joint will most likely be invalid. From this and the fact that SVM can account for outliers, negative examples could safely be generated with this approach.

We experimented with different types of kernels (linear, polynomial, Gaussian) and varying error costs. The corresponding SVMs have been evaluated using standard cross-validation techniques: the classifiers have been trained using all-but-one sequences and the average mis-classification error ϵ of the sequence left has been estimated.

Results clearly show that linear and polynomial kernels very poorly model the human body poses ($\epsilon > 0.5$). Gaussian kernels, which are more local, give very good classification error rates. Tables 1 and 2 report the classification error rates ϵ as well as the number of support vectors N_{sv} for Gaussian kernels with varying kernel size σ and error cost C. The SVM used in the rest of the paper uses a Gaussian kernel with $\sigma = 10$ and $C = 100$, which provides a good trade-off between error rate and number of support vectors.

3.1.2 Tracking with SVMs

The tracking problem then consists of finding the motion transformation $\Delta^\star$ that maps the previous body Π_{t-1} pose to a body pose $\Pi^\star$ that is valid while minimizing:

$$\begin{aligned} E^2(\Delta^\star) &= ||\Delta^\star - \Delta||^2_\Sigma \\ &= (\Delta^\star - \Delta)^\top \Sigma^{-1} (\Delta^\star - \Delta) \end{aligned} \tag{4}$$

Articulated constraints are guaranteed by using the minimal parameterization $\Delta^\star = \mathbf{V}\delta^\star$ introduced in [12]. Let $\bar{\Delta} = \mathbf{V}\bar{\delta}$ be the (unconstrained) articulated transformation. The constrained minimization of criteria $E^2(\Delta^\star)$ is replaced with the one of $\bar{E}^2(\delta^\star)$:

$$\begin{aligned} \bar{E}^2(\delta^\star) &= ||\Delta^\star - \bar{\Delta}||^2_\Sigma \\ &= (\Delta^\star - \bar{\Delta})^\top \Sigma^{-1} (\Delta^\star - \bar{\Delta}) \\ &= (\delta^\star - \bar{\delta})^\top \mathbf{V}^\top \Sigma^{-1} \mathbf{V} (\delta^\star - \bar{\delta}) \end{aligned} \tag{5}$$

with the constraint $g(\delta^\star) = f(\Pi^\star) = f(T_{\Delta^\star}(\Pi_{t-1})) > 0$ where $f(.)$ is the decision function estimated by the SVM, as in eq.(3).

This is a standard constrained optimization problem that can be solved using Lagrangian methods or gradient projection methods [2]. Because of its simplicity, we implemented a variant of Rosen's gradient projection method described in [13].

3.2 Gesture Recognition

3.2.1 Detection

In trying to recognize a gesture, we start with the simpler problem of detecting its occurrence. We partition the space of possible poses into poses corresponding to gestures we wish to classify and those that do not. On examination of these spaces, it is clear that some individual gestures have overlapping poses, while others are clearly separated. Gestures that overlap are grouped in the same group.

We used SVM classifiers to learn the space of static poses corresponding to gestures. For each gesture group, a SVM is trained, using the static poses corresponding to all gestures in that gesture group as positive examples, and all static poses corresponding to non-gestures and other gesture groups. The feature x used in the SVM classifier are the 3D pose generated by the articulated tracker. As the articulated tracker tracks the body, the pose is tested against all SVM. When one is triggered, meaning a SVM has detected that the current pose is one of the static poses in the gesture group it is responsible for, we note the detection. With a perfect detector, we would immediately begin passing 3d hand positions to our recognizer, and stop when the SVM no longer triggers. Because we do not have a perfect detector, we ignore rapid oscillations in the signal, effectively running a low-pass temporal filter on the SVM decision function. For example, if a SVM has been triggering for a while and it stops for a frame, and starts detecting again, we assume the SVM has made an error, and continue sending 3D hand

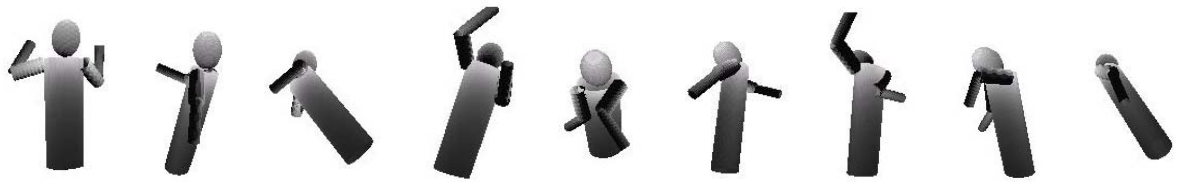

Figure 2: Body poses corresponding to 9 support vectors (out of 382) estimated by the SVM.

positions to our recognizer. The reverse is also true. If a SVM has not been triggering for a while, and it triggers for a frame, and stops triggering again, we assume the SVM has made an error, and do not send any information to our recognizer.

3.2.2 Recognition

Our recognizer is made up of continuous Hidden Markov Models which model specific gestures. As described in Rabiner and Juang [24], Hidden Markov Models deals well with the time element of gestures. One of the main advantages of using HMMs is the ability to know the probability of an incomplete observation sequence has of being produced by a particular model. This allows us to predict at any point in time what gesture might be occurring, and respond with the appropriate actions. Also we can identify various parts of a gesture, and easily extract various parameters.

Our recognizer runs only when a detection has occurred. When a SVM which models a specific gesture group triggers and starts passing 3d hand positions to the recognizer, each HMM corresponding to gestures within the gesture group starts computing the probability of that sequence being generated by that model. When the SVM stops triggering, our recognizer uses the probability of the sequence to classify the gesture. It orders the gestures within the gesture group according to the most probable, and calculates their corresponding parameters. This along with their associated probabilities are the input of the command recognizer.

4. APPLICATION

Not only can a vision interface supplement a speech interface, it can also provide an alternative way to convey the same information. Using gestures to do various actions like selecting an application, or playing a video stream, flipping through a photo album, the voice can be free, for various tasks like carry a conversation or give a talk.

A user can then select any application by pointing to the window or icon, and specify an action through speech, like saying ”Open” while pointing at an icon. If a user wants to resize or shrink a window, they can simply frame the window with their hands and resize to its desired size, with or without the aid of speech. This same gesture can be used in various context to accomplish related tasks. For example, this gesture can be used to zoom in or out of a image, or map. Rather than having to learn a new interface with different icons or ways of using the mouse, a user can do the most appropriate gesture or speech to get their point across.

For drawing applications, using vision and speech to interface with a computer can often be much easier than using a keyboard and mouse. A vision and speech interface allows us to get rid of all those icons that take up the screen, and limit your workspace, because we can just say ”edit”, ”drawing mode”, or ”add square”, or draw a square with our finger, resize or rotate an object. In general, we can express our 3d canvas using intuitive 3d motions, rather than learning to express 3d space with icons.

Action	Gesture	Speech
Select	Point	Select [object]
Select Region	Draw a path	Select [object]
Resize	Move hands along diagonal	Change size of [object] Enlarge [object] Shrink [object]
Next	Flip forward	Next Go forward
Back	Flip backward	Previous Go back Return

Table 3: Multimodal Actions

4.1 Implementation

In our multimodal system, speech is processed using GALAXY [26]. GALAXY is an architecture for integrating speech technologies to create conversational spoken language systems. The current version handles specialized servers for word recognition, language understanding, database access and speech synthesis.

The vision system consists of a stereo camera connected to a standard PC Pentium 4 (2Gz). The stereo camera captures images of the user and transfers them to the articulated body tracker that estimates the corresponding body pose (Section3.1. Sequences of body poses are used in the gesture recognition system to identify gestures (Section3.2.

Our implementation focused on the gesture recognition aspect of our multimodal interface. We narrowed down the actions a user can perform when interfacing with the computer to an experimental set of actions, that is smaller is size while maintains a most of the complexity of the problem. In Table 3, the set of actions we experimented on are listed.

4.1.1 Training Our Detector

We partitioned the gestures into two groups: one-handed gestures and two-handed gestures. The first two gestures, ”point” and ”draw a path”, are grouped into the one-handed gestures group. The rest are in the two-handed gestures group.

We collected on average 25 sequences of each gesture across a sample space of 8 people. Each sequence is a collection of images of size 320 x 240, provided by a stereo camera. The 3D model used in the experiments consists only of the upper body parts (torso, arms, forearms and head). The torso has been purposely made long to

σ	5	10	15	30
1	0.0270	0.0837	0.1311	0.1951
100	0.0103	0.0181	0.0286	0.0477
200	0.0094	0.0160	0.0237	0.0317
300	0.0104	0.0144	0.0237	0.0317

Table 4: Classification error rates ϵ for one-handed SVM trained with Gaussian kernels with varying error cost C and kernal size σ

σ	5	10	15	30
1	0.0216	0.0405	0.0478	0.0567
100	0.0116	0.0168	0.0225	0.0293
200	0.0108	0.0141	0.0198	0.0257
300	0.0112	0.0137	0.0190	0.0233

Table 5: Classification error rates ϵ for two-handed SVM trained with Gaussian kernels with varying error cost C and kernal size σ

compensate for the lack of hips and legs in the model. The complete tracking algorithm can run on a Pentium 4 (2GHz) processor at a speed ranging from 6Hz to 10Hz.

We trained two SVM classifier to model our two groups of gestures. The features x used in the SVM are the relative orientation of the body with respect to the world coordinate system and the relative orientations of connected limbs. Each image in our data collection is labelled as either a pose corresponding to a one-handed gesture, a two-handed gesture or a neutral position.

The SVMs have been evaluated using standard cross-validation techniques: the classifiers have been trained using 90% of the training data, and the average mis-classification error ϵ on the remaining training data is calculated. Tables 4 and 5 report the classification error rates $\epsilon$for Gaussian kernels with varying kernel size σ and error cost C. The SVM used in the rest of the paper uses a Gaussian kernel with $\sigma = 5$ and $C = 200$.

4.1.2 Training Our Recognizer

Each gesture in the two-handed gesture group has its own model. The model for "resize" allows for different ways of performing this gesture, different starting positions, and has large spatial variations. This action is modelled with a 4 state HMM, shown in 4. The first state consists of both hands moving to frame the window. This can either be done by selecting the upper left corner and the lower right corner, or selecting the upper right corner and the lower left corner. The next state can be resizing along either diagonal, allowing the choose of two states. Both states return to the same final state to return to the neutral position.

The action "next" and the action "previous" have a more compli-

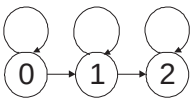

Figure 3: Select and Select Region HMM.

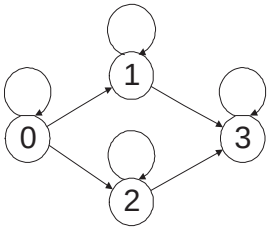

Figure 4: Resize HMM.

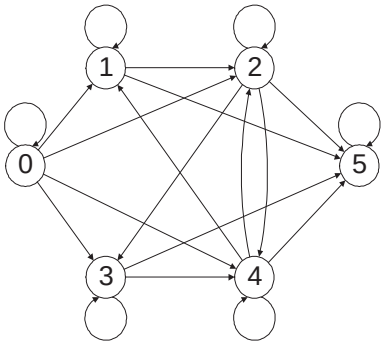

Figure 5: Next and Previous HMM.

cated structure, shown in figure 5. The first state consists of both hands moving out in front of the user, with the freedom of placing the hands together or apart. The next states follow this path, starting with two hands together: 1) the hands move apart. 2) the hands move together. This can be repeated as many times as a user wants, resulting in high temporal variation, specifying the number of times the computer should respond, for example by flipping through a sequence of images or webpages.

While the two-handed gesture group demonstrates how successfully we can classify different gestures, we defined only one HMM for both gestures to demonstrate the flexibility of our system. Rather than focusing on the probability of a sequence of frames being generated by a particular HMM, we focus on its ability to parse a sequence into discrete states. The action *select* and emphselect region are modelled with the same 3 state left-to-right HMM. The three states are moving from a neutral position to the object to be selected (state 0), pointing at the selected object (state 1), and moving back to a neutral position (state 2). The corresponding state diagram is shown in figure 3.

The feature vector for two-handed gestures and one-handed gestures are different. Because we know that one of the hands in a one-handed gesture contains no useful information towards classification, we use a feature vector that uses only information from one hand. Because we do not know which hand the gesture is performed on, we test two sequence of feature vectors corresponding to either hand with our HMM, and use the results pertaining to the model that generates the highest probability.

The feature vectors for both two-handed gestures and one-handed gestures are directly computed from the 3d hand positions given by the articulated tracker. Given a 3d hand position, we estimate velocity, $\overrightarrow{v}$, by finding the difference between two frames. The

	F1	F2	F3	F4
previous	.1666	.2500	.3500	.2000
next	.0625	.0625	.0625	0
enlarge	0	.0535	.0870	.0435
average	.0764	.1220	.1665	0.0812

Table 6: Cross validation error

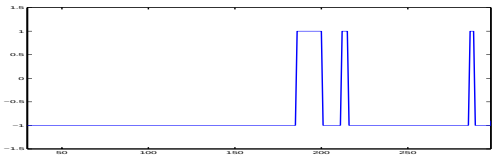

Figure 6: SVM decision function for one-handed gestures.

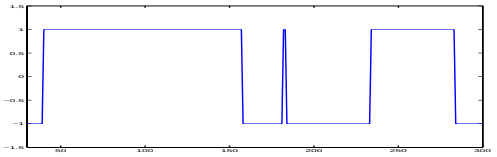

Figure 7: SVM decision function for two-handed gestures.

velocity is decomposed into its unit vector, $\widehat{v}$ and its magnitude, $|v|$. The relative position, $\overrightarrow{p_r}$, is computed by subtracting the right hand position from the left hand position. The relative velocity, $\overrightarrow{v_r}$, uses this newly calculated position, to estimate the local relative velocity between the left and right hand. Their unit vector, $\widehat{p_r}$ and $\widehat{v_r}$, and magnitude, $|p_r|$ and $|v_r|$, respectively, are computed as well.

To test which feature vectors were actually effective in distinguishing two-handed gestures, we ran 4 cross-validation tests and computed the classification rate:

1. F1 uses all the features described above,
2. F2 used $\overrightarrow{v}$, $\overrightarrow{p_r}$, and $\overrightarrow{v_r}$,
3. F3 used $\widehat{v}$, $\widehat{p_r}$, and $\widehat{v_r}$,
4. and F4 uses both unit vectors, $\widehat{v}$, $\widehat{p_r}$, and $\widehat{v_r}$, and magnitudes, $|v|$, $|p_r|$ and $|v_r|$.

Table 6 shows the cross validation error when trained with the four different feature vectors described above. Each model was trained 8 times, leaving out a few sequences of each type for each time. The error is to the ratio of how many times a sequence was classified as a different gesture versus how many sequences of each gesture was tested. This happens when the probability for that sequence is higher in a model that does not correspond its actual gesture. We can see the best overall classifier is F1, which used the feature vector with all calculated features. Another good classifier was F4, actually having a better classification rate for the action *previous*, than F1 did. The results shown in Table 6 using a Gaussian as a model for their observation probability. Mixtures of Gaussians were tested for each feature case, but resulted in terrible classification rates.

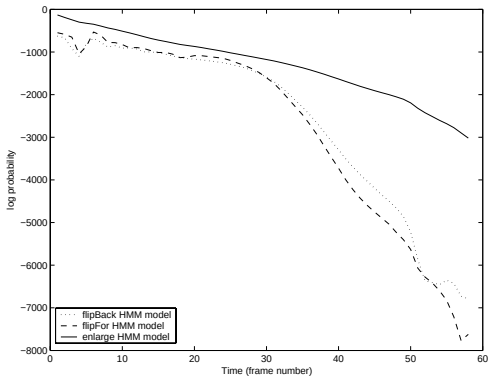

Figure 8: The log probability at each frame of an observation of the observations so far was created by each hmm model.

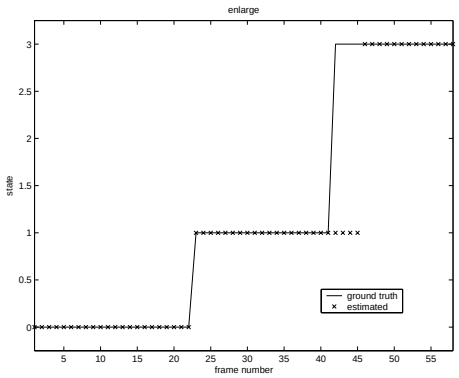

Figure 9: The states estimated by the resize HMM model compared to the ground truth states.

	Vision Only	Speech Only	Vision and Speech
previous	.1666	.0500	0
next	.0625	0	0
enlarge	0	.1000	0
average	.0764	.0500	0

Table 7: Classification error with different modes.

4.1.3 A sample sequence

The results shown here show a sample run through our gesture detection system. The sequence consists of a user performing the *next* gesture, then the *select* gesture, and finishing with the *resize*. At each captured frame, the SVM classifiers for one-handed and two-handed gestures determine whether or not it is a gesture. Shown in figure 6 and 7, when the decision function $f(x) = 1$, a gesture has been detected. As mentioned before, the signal filters out bursts in the signal. Because of the short duration of the detections around frame 210 and 280 for the one-handed gesture SVM, and frame 180 in for the two-handed gestures, they would be ignored and not trigger the recognizer.

When the SVM classifier for the two-handed gestures detects a gesture at frame numbers, 40 and 265, our recognizer will test the collection of subsequent frames that the SVM classifier detects as part of a two-handed gesture. We test this sequence of frames using the HMMs that correspond to the two-handed gesture: next, previous, and resize. When the SVM classifier for the one-handed gesture detects a gesture at frame number 180, the system will parse the pointing gesture into its appropriate states.

At frame number 245, the two-handed gesture happens to be resize. In figure 8, the log probability for each model in the two-handed gesture group is shown. As time passes, we can see a sharp decrease in the log probability corresponding to the *next* and *previous* HMMs in contrast with the the *resize* HMM. In figure 9, the way the HMM partitioned the gesture is compared to the ground truth partitioning. Although the estimated and ground truth don't match up exactly for the transition from state 2 to state 3, the different in position is actually quite small. In this way, we can extract the parameters that are relevant to this this particular gesture. For resize, we need the position of hands at the beginning of state 2 to know which window or object to resize, and the new size of the window or object with the position of the hand at end of state 2.

4.1.4 Command Recognizer Results

In Table 7, we show preliminary results on a small data set how using different modes effect the classification error of our system. A speech utterance was captured corresponding to each gesture in our data set. The n-best list of both the speech recognizer and the gesture recognizer are compared against each other, and the best overall is used to calculated the 3rd column. When using both vision and speech, the results are promising, although we expect the error to increase as we more thoroughly test the system.

5. FUTURE WORK

Although much progress has been made in the development of interfaces that use speech and gesture recognition, a great deal still remains to be done.

Rather than using effectively a low pass temporal filter on the SVM decision function, we would implement a multiple hypothesis method. For the original start of the detection, we run a recognizer on the gesture. We can estimate results for a gesture that ends when the SVM ends, while continuing to estimate the probabilities and parameters for a gesture that continues. Also, we can start another pass through the recognizer using this new start as the start of the gesture. After the gesture sequence has been completed, we can compare the results of all the possible segmentation of a sequence, and summarize this information to pass into our command recognizer.

There are other implementations we would like to evaluate. One of them is replacing the HMMs and simply using SVM by extracting key features from the entire gestures to classify. We would also like to integrate a symbol recognizer which would use the path extracted from the one-handed gestures into the system.

While we have described a system that combines speech and vision after the understanding of either modes are determined, we feel we could improve our accuracy by using information to assist recognition. Because speech can help limit the range of possible gesture, we can create new gesture groups that use common language as a criteria for different groups rather than similar poses.

6. CONCLUSION

Multimodal interfaces allow users to interact with machines in a natural and intuitive manner. In order to explore this emerging area of research, we developed an interface that incorporates recognition of both gestures and speech. We believe this system is flexible enough to cover a range of possible interactions for human-computer interfaces and allows for easy additions of different modules.

7. REFERENCES

[1] A. Azarbayejani, C. Wren, and A. Pentland. Real-time 3-d tracking of the human body. In *IMAGE'COM*, 1996.

[2] M. Bazaraa, H. Sherali, and C. Shetty. *Nonlinear programming: theory and algorithms*. Wiley, 1993.

[3] C. Breazeal. Towards sociable robots. *Robotics and Autonomous Systems*, pages 167–175, 2003.

[4] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *CVPR'98*, 1998.

[5] C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[6] J. Cassell. Nudge nudge wink wink: Elements of face-to-face conversation for embodied conversational agents. In *Embodied Conversational Agents*, 2000.

[7] R. Collobert, S. Bengio, and J. Marithoz. Torch: a modular machine learning software library.

[8] A. Corradini, R. Wesson, and P. Cohen. A map-based system using speech and 3d gestures for pervasive computing. In *IEEE International Conference on Multimodal Interfaces (ICMI'02)*, pages 191–196, 2002.

[9] T. Darrell, P. Maes, B. Blumberg, and A. Pentland. A novel environment for situated vision and behavior. In *IEEE Workshop on Visual Behaviors*, 1994.

[10] J. W. Davis and A. F. Bobick. The recognition of human movement using temporal templates. In *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.23, No. 3*, 2001.

[11] Q. Delamarre and O. D. Faugeras. 3d articulated models and multi-view tracking with silhouettes. In *Proceedings of ICCV'99*, pages 716–721, 1999.

[12] D. Demirdjian and T. Darrell. 3d articulated pose tracking for untethered deictic reference. In *International Conference on Multimodal Interfaces*, 2002.

[13] P. Fua and C. Brechbuhler. Imposing hard constraints on soft snakes. In *ECCV'96*, pages 495–506, 1996.

[14] D. Gavrila and L. Davis. 3d model-based tracking of humans in action: A multi-view approach. In *CVPR*, 1996.

[15] D. Hall, C. Le Gal, J. Martin, O. Chomat, and J. L. Crowley. Magicboard: A contribution to an intelligent office environment. In *Intelligent Robotic Systems*, 2001.

[16] M. Isard and A. Blake. Icondensation: Unifying low-level and high-level tracking in a stochastic framework. In *ECCV'98*, 1998.

[17] Y. A. Ivanov and A. F. Bobick. Recognition of visual activities and interactions by stochastic parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), Aug. 2000.

[18] M. Johnston and S. Bangalore. Finite-state multimodal parsing and understanding, 2000.

[19] N. Jojic, M. Turk, and T. Huang. Tracking articulated objects in dense disparity maps. In *International Conference on Computer Vision*, pages 123–130, 1999.

[20] I. Kakadiaris and D. Metaxas. 3d human body model acquisition from multiple views. *International Journal of Computer Vision*, 30(3), 1998.

[21] D. Koons, C. Sparrell, and K. Thrisson. Integrating simultaneous input from speech, gaze and hand gestures. *Intelligent Multimedia Interfaces, ed. by M. Maybury, MIT Press*, pages 257–276, 1993.

[22] N. Krahnstoever, S. Kettebekov, M. Yeasin, and R. Sharma. A real-time framework for natural multimodal interaction with large screen displays. In *Proc. of Fourth Intl. Conference on Multimodal Interfaces (ICMI 2002), Pittsburgh, PA, USA*, October 2002.

[23] K. Oka, Y. Sato, and H. Koike. Real-time tracking of multiple fingertips and gesture recognition for augmented desk interface systems. In *IEEE International Conference on Automatic Face and Gesture Recognition*, 2002.

[24] L. Rabiner and B. Juang. An introduction to hidden markov models. *IEEE ASSP Magazine*, Jan. 1986.

[25] B. Scholkopf, C. Burges, and A. Smola. *Advances in Kernel Methods*. 1998.

[26] S. Seneff, E. Hurley, R. Lau, C. Pao, P. Schmid, and V. Zue. Galaxy-ii: A reference architecture for conversational system development. In *ICSLP*, volume 3, pages 931–934, Sydney, Australia, 1998.

[27] H. Sidenbladh, M. J. Black, and D. J. Fleet. Stochastic tracking of 3d human figures using 2d image motion. In *ECCV (2)*, pages 702–718, 2000.

[28] C. Sminchisescu and B. Triggs. Covariance scaled sampling for monocular 3d body tracking. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Kauai, Hawaii, USA*. IEEE Computer Society Press, Dec 2001.

[29] C. Vogler and D. Metaxas. Parallel hidden markov models for american sign language recognition. In *International Conference on Computer Vision*, Kerkyra, Greece, Sept. 1999.

[30] A. Wilson and A. Bobick. Parametric hidden markov models for gesture recognition. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1999.

[31] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *PAMI*, 19(7):780–785, July 1997.

[32] M. Yamamoto and K. Yagishita. Scene constraints-aided tracking of human body. In *CVPR*, 2000.