Virtual Reality manuscript No.
(will be inserted by the editor)

Untethered Gesture Acquisition and Recognition for Virtual World Manipulation

David Demirdjian, Teresa Ko, Trevor Darrell

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139, USA

The date of receipt and acceptance will be inserted by the editor

Abstract Humans use a combination of gesture and speech to interact with objects and usually do so more naturally without holding a device or pointer. We present a system that incorporates user body-pose estimation, gesture recognition and speech recognition for interaction in virtual reality environments. We describe a vision-based method for tracking the pose of a user in real time and introduce a technique that provides parameterized gesture recognition. More precisely, we train a support vector classifier to model the boundary of the space of possible gestures, and train Hidden Markov Models on specific gestures. Given a sequence, we can find the start and end of various gestures using a support vector classifier, and find gesture likelihoods and parameters with a HMM. A multimodal recognition process is performed using rank-order fusion to merge speech and vision hypotheses. Finally we describe the use of our multimodal framework in a virtual world application that allows users to interact using gestures and speech.

1 Introduction

By providing different input channels, multimodal interfaces allow a more natural and efficient interaction between user and virtual world. 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 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, 20]; these generally use a physical stylus, or just a user's fingertip. For interaction with a kiosk [25], video wall [10], or conversational robot [3], it is desirable to have untethered tracking of full-body gestures. 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 virtual reality [22] and map exploration [24]. Schemes with explicit markers attached to hands or fingers have also been proposed, as in systems for optical motion capture in computer animation. Unfortunately, these systems' difficulty of use (e.g. attached wires, magnetic isolation of the room) prevents them from being generally usable by casual users. There have been many attempts to build untethered interfaces based on vision systems, but, to our knowledge, none of them has proven to be robust and fast enough to extract full articulated models for virtual reality purposes. 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 (HHM) or Finite-State Transducers (FST). Such techniques consider consecutive poses of the user given by the acquisition system and estimate the most probable gesture. There are many challenges in gesture recognition: the inputs are highly dimensional (the dimension of body poses are usually greater than 20); and the beginning and end of gestures are difficult to detect (contrary to speech where sentences are isolated by detecting surrounding silence).

In our multimodal system, speech is processed using the GALAXY system [29]. A stereo camera captures images of the user and transfers them to the articulated body tracker that estimates the corresponding body pose (described in Section 3.1). Sequences of body poses are used in the gesture-recognition system to identify gestures (Section 3.2). A rank-order fusion algorithm is used to merge command recognition; parameters are estimated for each visual gesture (e.g., size, location).

In the following sections, we present the architecture of our multimodal system for virtual reality. We then introduce our vision-based body-pose acquisition technique. A framework for gesture recognition is then described. Finally we demon-

strate its use for recognizing typical gestures in a virtual world environment and show some results.

2 Previous work

Systems based on tethered interfaces (e.g., datagloves) have been used for virtual reality. However untethered interfaces are more usable and provide more natural interactions. In this review of the literature, we focus exclusively on untethered body tracking techniques.

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 [17, 26, 18, 10] but are limited to planar interactions.

Articulated model-based techniques have been proposed to track people in monocular image sequences [34, 16, 23, 4, 35]. 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 such stochastic optimization techniques as particle filtering [30, 31]. Though promising, these approaches are not computationally efficient and real-time implementations are not yet available.

Stereo image-based techniques [21, 34] have given better pose estimates. Jojic, Turk and Huang [21] use 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.

The system described in [34] was successfully applied to detecting and classifying hand gestures in a conversational system [33, 6]. While the system worked 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 [33] using a variation of Hidden Markov Models. Other HMM approaches to gesture recognition include [19, 32].

Bobick and Davis [11] 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 [20,8] 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 [20] integrate speech and gesture at an early stage, most systems perform the recognition of speech and gesture separately and use a unification mechanism [8] to fuse 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 (the "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, particularly in conversational systems [29]. Inspired by [8], we fuse speech and gesture data after recognition.

3 Articulated Tracking and Gesture Recognition

We propose a cascaded approach to efficiently and reliably recognize gestures. The system can be separated into two components, an articulated tracker and a gesture recognizer. We describe each in turn. 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

A 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, and 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 [12,21,13,14]. 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 $\mathcal{A} \subset \mathcal{R}^{6N}$, which we will call the *articulated motion space*. Efficient tracking is possible by enforcing the spherical joint constraints using a projection-based approach such as [14]. This technique consists in finding the motion, $\Delta^\star$, which minimizes 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

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

where $\boldsymbol{\xi}_i$ is the rigid motion estimated by a 3D rigid object tracker and Σ_i the corresponding uncertainty. In our implementation, we used a tracker based on the Iterative Closest Point (ICP) algorithm [2]. The ICP algorithm was originally designed to register 3D surfaces. We use it to estimate attraction forces between 3D shapes (such as limbs) and 3D point clouds observed by the stereo camera.

The approach presented in this paper can be considered as an extension of [14] accounting for nonlinear 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.

SVM classifiers have been very popular in the computer vision community for their ability to learn complex boundaries between classes and also for their speed and efficiency. See [28,5] for a detailed description of SVM.

Given a data set, $\{x_i, y_i\}$, of examples x_i with labels $y_i \in \{+1, -1\}$, 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 (nonnegative) weights estimated by the SVM. A subset only of the weights, α_i,are nonnull. Examples x_i corresponding to nonzero α_i are the *support vectors*. The support vectors are the training examples that lie

Fig. 1 Body poses corresponding to 9 support vectors (out of 382) estimated by the SVM.

closest to the decision boundary. Each α_i defines the contribution of the corresponding support vector to the shape of the boundary. $k(x, x_i)$ is the kernel function corresponding to the dot product of the nonlinear 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 [28]. This allows for 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 (positive) body pose 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) were retained for the SVM training. Negative examples were 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.

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.

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

Results clearly show that linear and polynomial kernels model human body poses very poorly ($\epsilon > 0.5$). 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 pose, Π_{t-1}, to a body pose, $\Pi^\star$, 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 [14]. Let $\bar{\Delta} = \mathbf{V}\bar{\delta}$ be the (unconstrained) articulated transformation. The constrained minimization of criteria $E^2(\Delta^\star)$ is replaced with

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

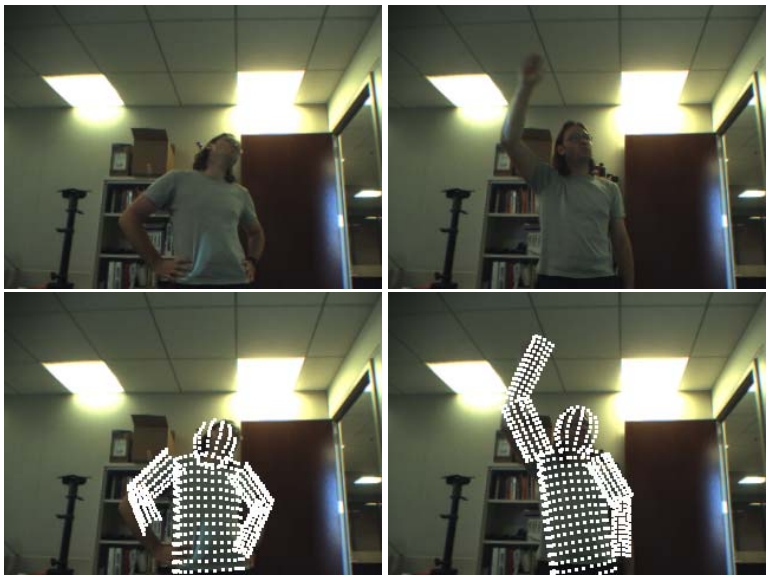

Fig. 2 Images (top) and corresponding articulated model (bottom) extracted from a tracking sequence.

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 (3).

This is a standard constrained optimization problem that can be solved using Lagrangian methods or gradient projection methods [1]. Because of its simplicity, we implemented a variant of Rosen's gradient projection method described in [15]. Figure 2 shows two images extracted from a video sequence and their corresponding articulated models estimated by our tracking algorithm.

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 nongestures and other gesture groups as negative examples. The feature x used in the SVM classifier is the 3D pose generated by the articulated tracker. As the articulated tracker tracks the body, the pose is tested against all SVM classifiers. 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 then starts detecting again, we assume the SVM has made an error, and continue sending 3D hand 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. Hidden Markov Models deal well with the time element of gestures [27]. One of the main advantages of using HMMs is the ability to know the probability of an incomplete observation sequence 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,is the input of the command recognizer.

4 Interactions in a Virtual Environment

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, playing a video stream, or flipping through a photo album, leaves the voice free for such tasks as carrying on a conversation or giving a talk.

Fig. 3 User interacting with the *Virtual Studio* application, which allows manipulation of objects with natural gestures and no wires or special markers. A stereo camera (above the projection screen) gives visual information to the tracking system, which estimates the position of the arms of the user (in red on the projection screen) and detects pointed objects (e.g. yellow pyramid).

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 contexts 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, limiting workspace, because we can just say “edit”, “drawing mode”, “add square”, draw a square with our finger, or resize or rotate an object. In general, we can access our 3D canvas using intuitive 3D motions, rather than learning to express 3D space with icons.

4.1 Virtual Studio

We used our multimodal infrastructure to create a virtual studio application that allows a user to edit a virtual world and navigate around it using speech and gesture inputs. The physical setup consists of a stereo camera that observes the user, a projector that displays the 3D virtual world on a 2-m wide screen and a microphone array to capture audio signals from the user while removing a considerable amount of background noise.

The stereo camera provides dense disparity maps. The disparity maps are grayscale images whose pixel intensities (disparities) are inversely proportional to the depth of the corresponding point. In our framework, the disparity maps are further processed to recover the approximate position of the user using background subtraction techniques [9]. Given the parameters of the stereo camera (focal lengths, baseline) a full 3D reconstruction of the user is computed and used as input for our articulated tracking algorithm.

A set of commands has been defined to manipulate (create, delete, edit, move, ...) different geometric shapes and objects (cubes, spheres, ...). Moreover, we take deictic references and pronouns into account by using anaphora resolution rules. Therefore our system can handle requests such as:

- "Make a blue sphere here [user pointing at the screen]"
- "Make a red cube in the middle of the screen"
- "Move it [user pointing at the sphere] there [user pointing at the screen]"
- "Resize the cube" [user making a resizing gesture]
- etc.

Figure 3 shows a user interacting with the *Virtual Studio* application. Full videos showing the system can be seen at:

`http://www.ai.mit.edu/~demirdji/movie/multimodal/`

4.2 Implementation

In our multimodal system, speech is processed using GALAXY [29]. 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 2-GHz Pentium 4 PC. The stereo camera captures images of the user and transfers them to the articulated body tracker that estimates the corresponding body pose (Section 3.1). Sequences of body poses are used in the gesture-recognition system to

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

identify gestures (Section 3.2). We constrained the set of actions a user can perform when interfacing with the computer to an experimental set of actions that is smaller in size while maintaining most of the complexity of the problem. In Table 3, the set of actions we experimented on are listed.

Finally, a multimodal fusion algorithm integrates speech and gesture recognition (see Section 4.3) and generates the corresponding commands/requests to the virtual world (rendered using Java/Java3D).

4.2.1 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 an average of 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 compensate for the lack of hips and legs in the model.

We trained two SVM classifiers 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 labeled 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 misclassification error, ϵ, on the remaining training data is calculated. Ta-

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 a one-handed SVM trained with Gaussian kernels with varying error cost, C, and kernel 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 kernel size, σ

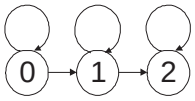

Fig. 4 Select and Select Region HMM.

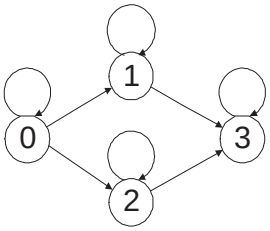

Fig. 5 Resize HMM.

bles 4 and 5 report the classification error rates, ϵ, 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 $C = 200$.

4.2.2 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 Figure 5. The first state consists of

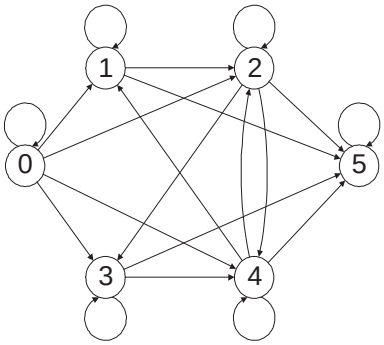

Fig. 6 Next and Previous HMM.

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 choice 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 complicated structure, shown in Figure 6. 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 *select 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 4.

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 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 hands. Their unit vectors, $\widehat{p_r}$ and $\widehat{v_r}$, and magnitudes, $|p_r|$ 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 their classification rates:

1. F1 used 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 10 times, leaving out a few sequences of each type each time. The error is the ratio of how many times a sequence was classified as a different gesture and how many sequences of each gesture were tested. This happens when the probability for that sequence is higher in a model that does not correspond to 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 used a Gaussian as a model for their observation probability. Mixtures of Gaussians were tested for each feature case, but resulted in poor classification rates.

4.3 Command Recognizer Results

In Table 7, we show some results on a small data set of 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 calculate the 3rd column. When using both vision and speech,

	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 of the gesture recognizer with the feature vectors F1, F2, F3 and F4. The values reported correspond to the average error rates obtained by running a standard leave-one-out test 10 times.

	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. The values reported correspond to the average error rates obtained by running a standard leave-one-out test 10 times.

the results are promising, although we expect that the error may increase as we generalize to larger task vocabularies.

5 Conclusion and Future Work

Multimodal interfaces allow users to interact with a virtual world in a natural and intuitive manner. In order to explore this emerging area of research, we developed an untethered body tracker interface that incorporates recognition of both gestures and speech. We believe this system is flexible and robust enough to cover a range of possible interaction styles for interacting with virtual environments.

The tracking system works well in contexts such as the one corresponding to the use of our virtual studio application (person standing, facing the large display, moderate motion speed). However the performance of the tracking system may decrease when used in a more general context (e.g., any movement allowed) or if the user performs very fast motions. The tracking system may then provide inaccurate body pose estimates, start losing track and provide incorrect gesture recognition. The visual feedback present in our system (projection of the estimated arm position in the virtual scene) provides a way to recover from tracking failure: In such cases the user can detect tracking failures and ask the tracking system to reinitialize the articulated body model. Current work consists of integrating a single-frame pose

estimator to automatically detect tracking failure and reinitialize the articulated body model.

This paper describes a multimodal recognition approach with a small set of actions. Some preliminary experiments have been performed on a similar system containing about 10 actions and show that the performance of the system is comparable to the results reported in this paper.

There are many avenues of future work for this research, including the recognition of natural gestures, the use of machine learning techniques to make the interface adaptive to individual users and the integration of natural language dialog systems to allow conversational interaction. Yet another great challenge for future work is to study multiuser interaction and navigation in virtual environments. We believe that in multiuser situations passive and untethered sensing of users' pose and gesture becomes even more valuable.

References

1. M. Bazaraa, H. Sherali, and C. Shetty. *Nonlinear programming: Theory and algorithms*. John Wiley and Sons, 1993.
2. P. Besl and N. MacKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:239–256, 1992.
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 *Proceedings of Computer Vision and Pattern Recognition (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 J. Cassell, S. Prevost, J. Sullivan, and E. Churchill, editors, *Embodied Conversational Agents*. MIT Press, 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 *Proceedings of International Conference on Multimodal Interfaces (ICMI'02)*, pages 191–196, Pittsburgh, PA, USA, 2002.
9. T. Darrell, D. Demirdjian, N. Checka, and P. Felzenszwalb. Plan-view trajectory estimation with dense stereo background models. In *Proceedings of International Conference on Computer Vision (ICCV'01)*, Vancouver, Canada, July 2001.
10. T. Darrell, P. Maes, B. Blumberg, and A. Pentland. A novel environment for situated vision and behavior. In *IEEE Workshop on Visual Behaviors*, 1994.

11. J. W. Davis and A. F. Bobick. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):257–267, 2001.
12. Q. Delamarre and O. D. Faugeras. 3D articulated models and multi-view tracking with silhouettes. In *Proceedings of International Conference on Computer Vision (ICCV'99)*, pages 716–721, 1999.
13. D. Demirdjian. Enforcing constraints for human body tracking. In *Proceedings of Workshop on Multi-Object Tracking*, Madison, Wisconsin, USA, 2003.
14. D. Demirdjian and T. Darrell. 3D articulated pose tracking for untethered deictic reference. In *Proceedings of International Conference on Multimodal Interfaces (ICMI'02)*, Pittsburgh, PA, USA, 2002.
15. P. Fua and C. Brechbuhler. Imposing hard constraints on soft snakes. In *Proceedings of European Conference on Computer Vision (ECCV'96)*, pages 495–506, 1996.
16. D. Gavrila and L. Davis. 3D model-based tracking of humans in action: A multi-view approach. In *Proceedings of Computer Vision and Pattern Recognition (CVPR'96)*, 1996.
17. 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.
18. M. Isard and A. Blake. Icondensation: Unifying low-level and high-level tracking in a stochastic framework. In *Proceedings of European Conference on Computer Vision (ECCV'98)*, 1998.
19. 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.
20. M. Johnston and S. Bangalore. Finite-state multimodal parsing and understanding. In *In Proceedings of International Conference On Computational Linguisitics*, pages 369–375, 2000.
21. N. Jojic, M. Turk, and T. Huang. Tracking articulated objects in dense disparity maps. In *International Conference on Computer Vision*, pages 123–130, 1999.
22. E. Kaiser, A. Olwal, D. McGee, H. Benko, A. Corradini, X. Li, S. Feiner, and P. Cohen. Mutual disambiguation of 3d multimodal interaction in augmented and virtual reality. In *Proceedings of International Conference on Multimodal Interfaces (ICMI'03)*, pages 12–19, Vancouver, BC, November 2003.
23. I. Kakadiaris and D. Metaxas. 3D human body model acquisition from multiple views. *International Journal of Computer Vision*, 30(3), 1998.
24. D. Koons, C. Sparrell, and K. Thrisson. Integrating simultaneous input from speech, gaze and hand gestures. *Intelligent Multimedia Interfaces*, pages 257–276, 1993.
25. N. Krahnstoever, S. Kettebekov, M. Yeasin, and R. Sharma. A real-time framework for natural multimodal interaction with large screen displays. In *Proceedings of International Conference on Multimodal Interfaces (ICMI'02)*, Pittsburgh, PA, USA, October 2002.

26. 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.
27. L. Rabiner and B. Juang. An introduction to hidden markov models. *IEEE ASSP Magazine*, 3(1):4–16, Jan. 1986.
28. B. Scholkopf, C. Burges, and A. Smola. *Advances in Kernel Methods*. MIT Press, 1998.
29. 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.
30. H. Sidenbladh, M. J. Black, and D. J. Fleet. Stochastic tracking of 3D human figures using 2d image motion. In *Proceedings of European Conference on Computer Vision (ECCV'00)*, pages 702–718, 2000.
31. C. Sminchisescu and B. Triggs. Covariance scaled sampling for monocular 3D body tracking. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR'01)*, Kauai, Hawaii, USA, Dec 2001.
32. C. Vogler and D. Metaxas. Parallel hidden markov models for american sign language recognition. In *International Conference on Computer Vision*, Kerkyra, Greece, 1999.
33. A. Wilson and A. Bobick. Parametric hidden markov models for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(9):884–900, 1999.
34. C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):780–785, July 1997.
35. M. Yamamoto and K. Yagishita. Scene constraints-aided tracking of human body. In *Proceedings of Computer Vision and Pattern Recognition (CVPR'00)*, 2000.