Visual Modeling of Dynamic Gestures Using 3D Appearance and Motion Features

Guangqi Ye, Jason J. Corso, and Gregory D. Hager

Computational Interaction and Robotics Laboratory
The Johns Hopkins University
grant@cs.jhu.edu
jcorso@cs.jhu.edu
hager@cs.jhu.edu

We present a novel 3-D gesture recognition scheme that combines the 3-D appearance of the hand and the motion dynamics of the gesture to classify manipulative and controlling gestures. Our method does not directly track the hand. Instead, we take an object-centered approach that efficiently computes 3-D appearance using a region-based coarse stereo matching algorithm. Motion cues are captured by differentiating the appearance feature. An unsupervised learning scheme is carried out to capture the cluster structure of these features. Then, the image sequence of a gesture is converted to a series of symbols that indicate the cluster identities of each image pair. Two schemes, i.e., forward HMMs and neural networks, are used to model the dynamics of the gestures. We implemented a real-time system and performed gesture recognition experiments to analyze the performance with different combinations of the appearance and motion features. The system achieves recognition accuracy of over 96% using both the appearance and motion cues.

1 Introduction

Gestures have been one of the important interaction media in current human-computer interaction (HCI) systems [3, 4, 11, 12, 14, 16, 18, 22, 25, 26, 28]. Furthermore, for 3-D virtual environments (VE) in which the user manipulates 3-D objects, gestures are more appropriate and potentially powerful than traditional interaction media, such as a mouse or a joystick. Vision-based gesture processing also provides more convenience and immersiveness than those based on mechanical devices.

We are interested in modeling manipulative and controlling gestures [14] for direct manipulation and natural interaction. These gestures have a temporal nature that involves complex changes of hand configurations. Furthermore,

human hands and arms are highly articulate and deformable objects. As a result, gestures normally consist of complex 3-D global and local motion of the hands and arms. The complex spatial properties and dynamics of such gestures render the problem extremely difficult for pure 2-D (e.g. template matching) methods. Ideally, we would capture the full 3-D information of the hands to model the gestures [11]. However, the difficulty and computational complexity of visual 3-D localization prompts us to question the necessity of doing so for gesture recognition.

Most reported gesture recognition work in the literature (see Section 1.1) relies heavily on visual tracking and template recognition algorithms. However, general human motion tracking is well-known to be a complex and difficult problem [8, 17]. Additionally, while template matching may be suitable for static gestures, its ability to capture the spatio-temporal nature of dynamic gestures is in doubt. Alternatively, methods that attempt to capture the 3-D information of the hand [11] have been proposed. However, it is well-known that, in general circumstances, the stereo problem is difficult to solve reliably and efficiently [19].

To that end, we present a novel scheme to model and recognize 3-D temporal gestures using 3-D appearance and motion cues without tracking and explicit localization of the hands. Instead, we follow the site-centered computation fashion of Visual Interaction Cues (VICs) paradigm [3, 25].

We propose that interaction gestures can be captured in a local neighborhood around the manipulated object based on the fact that the user only initiates manipulative gestures when his or her hands are close enough to the objects. The advantages of this scheme are its efficiency and flexibility. The dimension of the volume of the local neighborhood around the manipulated object can be adjusted conveniently according to the nature of the particular interaction environment and the applicable gestures. For example, in a desktop interaction environment, the interaction elements are represented as small icons on a flat panel. Manipulative gestures are only initiated when the user's hand is near the surface of the panel, so we only need to observe a small volume above the panel with the icon sitting at the center of the bottom. The height and diameter of the volume is also limited to be able to capture enough visual cues to carry out successful gesture recognition.

The remainder of this paper is structured as follows. In Section 2 we present a novel method to efficiently capture the 3-D spatial information of the gesture without carrying out a full-scale disparity computation. We discuss how to learn the cluster structure of the appearance and motion features via an unsupervised learning process in Section 3. Two ways to model the dynamics of the gestures — forward Hidden Markov Models (HMMs) [10, 20] and multilayer neural networks [6] — are also presented. In Section 4 we demonstrate our real-time system that implements the proposed method and present the results of gesture recognition. Section 5 concludes the paper.

1.1 Related Work

[23] gives a general overview of the state of the art in gesture analysis for vision-based human computer interaction. Robust hand localization and tracking, modeling the constraints of hand motion and recognizing temporal gesture patterns are among the most difficult and active research areas. Compared to other techniques, such as neural network and rule-based methods [14], HMMs [24, 25] and its extension [2] are a popular scheme to model temporal gestures.

Many HCI systems [12, 14, 16, 22, 23] have been reported that enable the user to use gestures as a controlling or communicative media to manipulate interaction objects. The hand or fingertips are detected based on such cues as visual appearance, shape, human body temperature via infrared cameras. A variety of algorithms have been applied to track the hand [23], such as the Kalman filter and particle filter [5].

With a model-based approach [1, 13], it is possible to capture the gesture in higher dimensionality than 2-D. In [1] the 3-D hand model is represented as a set of synthetic images of the hand with different configurations of the fingers under different viewpoints. Image-to-model matching is carried out using a Chamfer distance-based computation. One of the difficulties of this approach is constructing a good 3-D model of the hand that can deal with variance between different users. Furthermore, efficient algorithms are necessary to handle the matching between models and input images. Another approach to capture 3-D data is to use special cameras [11], such as 3-D cameras or other range sensors. However, the hardware requirement limits its application to general HCI systems.

2 Capturing 3D Features of Manipulative Gestures

Manipulative and controlling gestures have a temporal 3-D nature involving the interaction between human hands and other objects. Example subjects include the tools and toys in a VE, interaction elements in an HCI interface, and so forth. One of the most difficult problems in visual modeling of gestures is data collection and feature representation [23]. We propose an efficient scheme to capture 3-D gesture appearance and motion in an object-centered fashion. We use the Visual Interaction Cues (VICs) [27] paradigm in this work. We provide a brief summary of the paradigm in Section 2.1. Under the VICs paradigm, we are able to make the assumption that the content of a manipulative gesture can be captured in a local region around the manipulated object. This assumption is valid in many HCI scenarios [27], such as a WIMP-style interface [21].

2.1 The Visual Interaction Cues Paradigm

As we discussed earlier, manipulative and controlling gestures involve the interaction between human hands and objects in the environment. Typical methods for vision-based interaction attempt to perform continuous, global user tracking to model the interaction. Such techniques are computationally expensive, prone to error and the re-initialization problem, prohibit the inclusion of arbitrary numbers of users, and often require a complex gesture-language the user must learn.

However, under the VICs paradigm [27], we focus on the components of the interface itself instead of on the user. The VICs paradigm is a methodology for vision-based interaction operating on the fundamental premise that, in general vision-based human computer interaction settings, global user modeling and tracking are not necessary. There are essentially two parts to the VICs paradigm.

First, we define and maintain a mapping between the interface components and their respective projections in the images (Figure 1). Let $\mathcal{I}$ be an image defined by a set of pixel locations (points in $\mathbb{R}^2$). Let $\mathcal{W}$ be the space in which the components of the interface reside. In general, $\mathcal{W}$ is the 3D Euclidean space $\mathbb{R}^3$ but it can be the Projective plane $\mathbb{P}^2$ or the Euclidean plane $\mathbb{R}^2$. Define an interface component mapping $M{:}\mathcal{C} \rightarrow \mathcal{X}$, where $\mathcal{C} \subset \mathcal{W}$ and $\mathcal{X} \subset \mathcal{I}$. In Figure 1, we show an example of this concept for stereo cameras. In this case, two mappings are required with one for each image. Intuitively, the mapping defines a region in the image to which an interface component projects.

Second, if, for each interface component and the current images, a map is known, detecting a user action reduces to analyzing a local region in the image. This is a fairly general statement and the subject of this paper. We provide a simple example here for expository purposes. Let the interface component be a standard push-button. Then, to detect a button-press by a user, we expect a certain sequence of *interaction cues* to occur in the image region. An example of such cues might be *motion* $\rightarrow$ *skin-color* $\rightarrow$ *finger-shape* $\rightarrow$ *finger pauses* $\rightarrow$ *motion and absence of skin-color*. Such cues may be heuristically defined or learned as in this paper.

2.2 3D Gesture Volume

Given a pair of stereo images of a scene, a disparity map can be computed using a standard correspondence search algorithm. Since we only care about the local neighborhood around the object, we can constrain the stereo search to a limited 3D space around the object. This brings about two advantages: first, we only care about the small patch of the image centered at the object; second, we only need to search through a small number of disparities (depths), which is a limited range around the depth of the object. To simplify the computation, we carry out the stereo matching process for a discrete number of image patches, not for each pixel position.

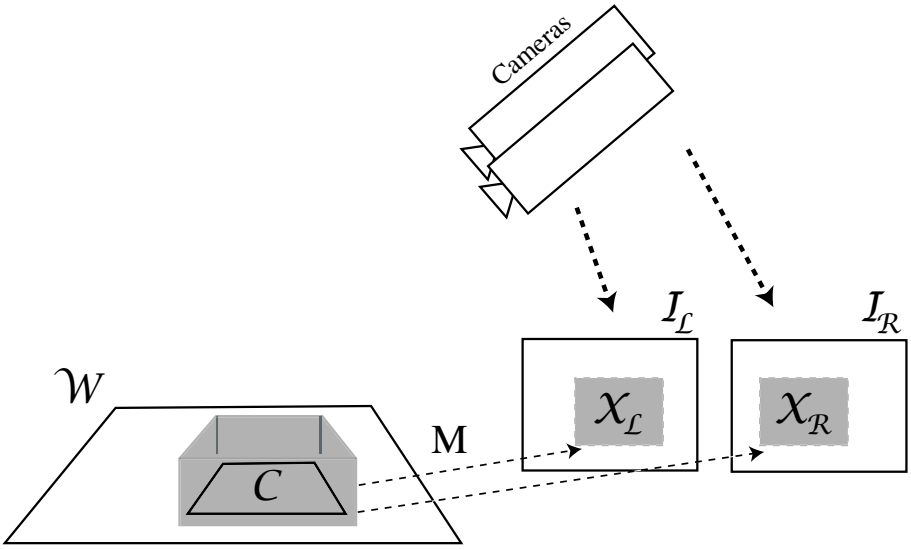


Fig. 1. Schematic explaining the principle of local image analysis for the VICs paradigm: M is the component mapping that yields a region of interest in the stereo images $\mathcal{I}_{\mathcal{L}}$ and $\mathcal{I}_{\mathcal{R}}$ for analyzing actions on component $\mathcal{C}$

.

Formally, let I_l and I_r be a pair of images of the scene. We split the images into tiles of equal size of $w \times h$. Here w and h refer to the width and height of the tile, respectively. Suppose we only consider a local area of size of $m \times n$ patches, starting at patch (x_0, y_0). Given a discrete parallax search range of $[0, (p-1) \times w]$, we can characterize the scene using a $m \times n \times p$ volume V as:

$$V_{x,y,z} = SIM(I_{l(x_0+x,y_0+y)}, I_{r(x_0+x+z,y_0+y)}) \quad (1)$$
$$x \in \{0, ..., m-1\}, y \in \{0, ..., n-1\}, z \in \{0, ..., p-1\}$$

Note that in the previous equation, the image index indicates a patch of the image, not a particular pixel. SIM is a similarity measurement between the two image patches. Example measurements include the sum of absolute of differences and sum of squared differences.

We convert the color images into hue images to reduce the impact of changes in lighting intensity because hue is a good color-invariant model [9]. Furthermore, we perform a comprehensive color normalization process [7] on each image to overcome the variance of illumination and lighting geometry across different interaction sessions. These techniques ensure the relative stability of the appearance feature under different imaging conditions.

Following this scheme, we can extract the features of the image as a very simple vector with the size of $m \times n \times p$. The typical size of the extracted

appearance vector is from 125 to 1000. In contrast, the size of the original image is 640×480 and the size of the local image around a typical object in our experiments is approximately 150×150. Thus, this feature extraction scheme significantly reduces the size of the the input data.

Figure 2 shows examples of the stereo image pair and the extracted 3D features of the scene. It can be seen that the extracted feature volume characterizes the different configuration of the user's hand with respect to the target interaction subject.

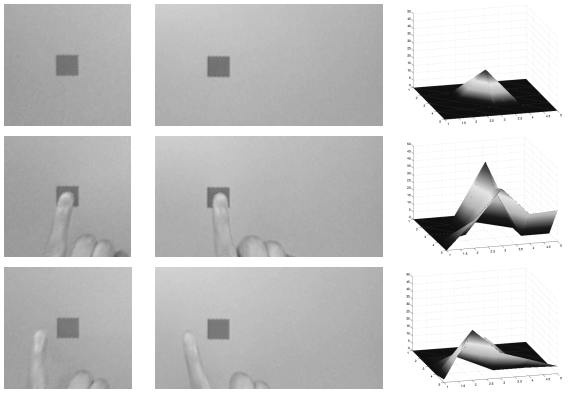


Fig. 2. Examples of the image pair and extracted appearance feature. The left and middle columns display left images and right images of the scene, respectively. The right column shows the bottom layer of the feature volume (i.e., $V_{x,y,z}$ with $z = 0$).

2.3 Motion by Differencing

Since we represent the 3D appearance of the gesture images using feature vectors, one simple way to capture the motion information of the gestures is to compute the the displacement in this feature space. In our real-time system, the change between consecutive frames is normally very small because of the high frame rate. Thus we compute the difference between the appearance feature of the current frame and that of several frames before.

$$Motion_i = V_i - V_{i-k}, \quad i = k+1, ..., M \tag{2}$$

One way to combine the appearance feature and the motion feature is to concatenate the two vectors to form a larger vector. This new vector contains both the static and temporal information of the gesture.

2.4 Analysis of the 3D Features

Given an image sequence that contains a particular manipulative gesture, we convert the sequence into a series of vectors, or points in the appearance

or motion space. Thus, the gesture can be conceptualized as a directed path connecting these points in the appropriate order. Intuitively we can model the gesture by learning the parameters of such a path. However, this appearance or motion space is still a relatively high-dimensional space, making the learning and recognition difficult to handle.

Furthermore, for a set of a 3-D appearance or motion feature points that are extracted from a dataset of gesture sequences, we can expect that there will be much redundancy of information. The reason is that the training set contains repeatable gestures and there are only a limited number of gestures in the set. To analyze the data redundancy, we use principal components analysis (PCA) technique on a dataset that consists of 622 gesture sequences. We experiment with representing the 125-dimensional appearance feature space using different numbers of principal components. Figure 3 shows the relationship between the average reconstruction error and the number of principal components. It can be seen that, using the first 25 principal components, we can achieve an average reconstruction error of less than 5%. Therefore, we expect to be able to characterize the appearance or motion feature of the gestures using data of much lower dimensionality without losing the capability to discriminate between them.

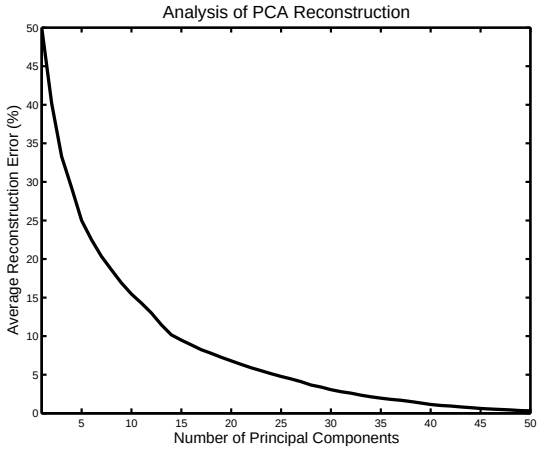


Fig. 3. PCA analysis on the 125-dimensional appearance feature space.

In the next section (3), we will discuss the techniques to learn the cluster structures of the 3D features and to model the dynamics of the temporal gestures in a feature space of reduced dimensionality.

3 Learning the Gesture Structure

3.1 Unsupervised Learning of the Cluster Structures of 3D Features

One of the popular ways to model temporal signals is to learn a statistical model [6]. However, the size of training data needed for statistical learning normally increases exponentially with the dimensionality of input features. This curse of dimensionality is one of the reasons that visual modeling of gestures is difficult. Thus we propose to reduce the dimensionality of the 3D feature by learning its cluster configuration.

We propose an unsupervised method to learn the cluster structure of the high-dimensional raw feature. Basically, we implement a K-means algorithm to learn the centroid of each of the clusters of the feature set. Then, the feature vectors are clustered using a Vector Quantization (VQ) [10] algorithm. We represent each feature v_i with one of the k clusters $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ based on nearest-neighbor criterion.

$$VQ(v_i) = \arg\min_{C_j \in \mathcal{C}} Dist(C_j, v_i) \tag{3}$$

Here, $Dist$ is computed as the Euclidean distance between two feature points.

The choice of the number of clusters to initialize the VQ algorithm is a difficult model selection problem. We handle this problem based on the analysis of the average representation error. The representation error is defined as the distance between feature point v_i and its corresponding cluster centroid $VQ(v_i)$. In theory, as the number of clusters k increases, the average representation error decreases. On the other hand, our aim of feature dimensionality reduction prefers smaller k. A trade-off is achieved by increasing the number of clusters until the average representation error only decreases slightly as k grows larger.

Figure 4 shows an example of the relationship between k and the representation error for a dataset of 125-D appearance features. We can see that, when the cluster number is larger than 8, increasing the cluster number can only slightly reduce the average error. Thus, we can select the number of clusters to be 8. In Section 4, we include an experimental validation of this analysis.

This learning scheme allows different ways of combining appearance and motion cues. Let V^{Appr} and V^{Mot} denote the extracted appearance and motion feature, respectively. The first way is to normalize each visual feature to the same scale and then concatenate the feature vectors to form a new feature (V^{Appr}, V^{Mot}). The dimensionality of this new feature space is the sum of the dimensionality of the individual feature spaces. Then we can carry out VQ on this new feature space. The second way to combine these two visual cues is to carry out VQ on each feature space separately. Let VQ_{Appr} and VQ_{Mot} denote the VQ projection in the appearance and motion feature space, respectively. The overall representation of the visual features thus can be expressed

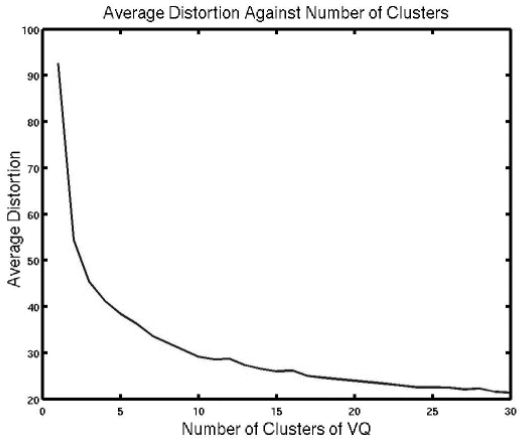


Fig. 4. The average representation error against number of clusters.

as a discrete vector $(VQ_{Appr}(V^{Appr}), VQ_{Mot}(V^{Mot}))$. Furthermore, since we know the number of clusters in each feature space, which is equivalent to the dimensionality of corresponding element of the 2-D discrete vector, we can further convert the 2-D vector into a scalar.

3.2 Gesture Modeling Using HMM

We use typical forward HMMs to model the dynamics of the temporal gestures. The input to the HMMs is the gesture sequence represented as a series of symbols with each symbol indicating the cluster identity of current frame. The basic idea is to construct a HMM for each gesture and learn the parameters of the HMM from the training sequences that belong to this gesture using the Baum-Welch algorithm [10, 15]. The probability that each HMM generates the given sequence is the criterion of recognition. The gesture sequence is recognized as the class with the highest probability. Rejection of invalid gestures is based on the thresholding of the best probability. If the highest probability that a sequence achieves on all HMMs is lower than a threshold, the sequence will be rejected. This threshold is chosen to be smaller than the lowest probability that each HMM generates the sequences that belong to that class in the training set.

In our experiment, we use a 6-state forward HMM to model each of the six manipulative gestures. Figure 5 shows the topology of the HMMs.

The choice of the number of the states in the forward HMM is based on the intuitive analysis of the temporal properties of the gestures to be modeled. In our current experiment, each of the gestures can be decomposed into less than 6 distinct stages. For example, if we use 3 spatial layers to represent the vicinity of a manipulated object, the gesture of swiping an icon to the left

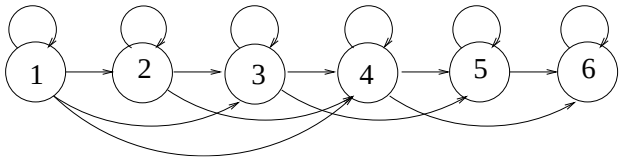


Fig. 5. HMM structure for the interaction gestures

can be viewed as such a configuration sequence of the hand: (1) entering the outer layer of the vicinity of the icon, (2) entering the inner layer (3) touching the icon to select it and (4) swiping the icon by moving the finger to the left side of the icon. Ideally, each of the distinct stages can be modeled by a certain state of the forward HMM. The parameter sets of the trained HMMs verify our expectation, in which the observation probability of each symbols of a gesture is high in one of the states and very small in the other states. Generally speaking, a dynamic process with n stages can be modeled using an n-state forward HMM with similar topology. For example, in [20], four-state HMMs are used to recognize American Sign Language.

3.3 Gesture Modeling Using A Multilayer Neural Network

Another way to learn gestures is to use multilayer neural networks. The input to the neural network is the entire gesture sequence, which is now a sequence of symbols. The output is the identity of the gesture. To meet the requirement of the neural network, we need to fix the length of each input sequence. We align each sequence to a fixed length by carrying out sub-sampling on those sequences that are longer than the predefined length and interpolation on those that are shorter. The parameters of the network are also learned from training data using the standard backpropagation algorithm [6].

In our current system, the neural network consists of 3 layers, i.e., the input and output layer and the hidden layer. The number of nodes in the hidden layer is chosen to be 50.

4 Experimental Results

4.1 Experimental Setup

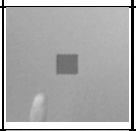
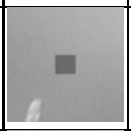
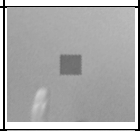
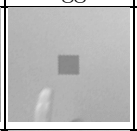
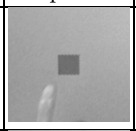
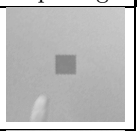
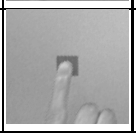
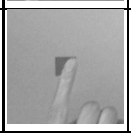
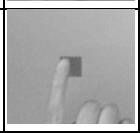
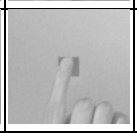
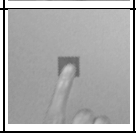
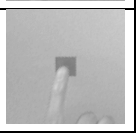
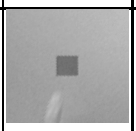
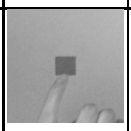
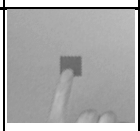
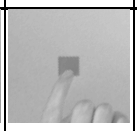
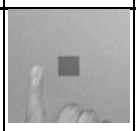
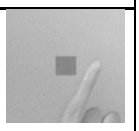
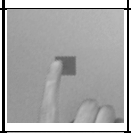
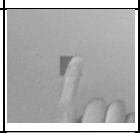
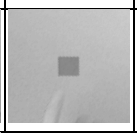
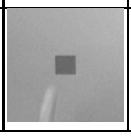
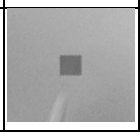
We use the 4-D Touchpad [3] as our experimental platform. We use a pair of color cameras to observe the interaction desktop which is presented as a flat panel on which the interaction elements are rendered. The system is calibrated using a pair of homographies thereby defining the required interface component mappings between the rendered image and the images captured from the cameras. The user interacts with the objects on the panel using

Fig. 6. The 4D Touchpad HCI platform.

manipulative and controlling gestures. Figure 6 shows the configuration of our experiment platform.

In our current experiments, we collect gesture sequences consisting of 6 interactive gestures, i.e., pushing a button, twisting a dial clockwise, twisting a dial anti-clockwise, toggling a switch, swiping an icon to the left and swiping an icon to the right. Table 1 shows several snapshots of the typical image sequences of the gestures.

Table 1. Example images of the gestures.

Gesture	Push	Twist	Twist Anti	Toggle	Swipe Left	Swipe Right
Stage 1						
Stage 2						
Stage 3						
Stage 4						
Stage 5						

We implement the system on a PC with dual Pentium III processors. The system achieves real-time speed; the processing is limited by the cameras (30Hz). The system processes the continuous video in the following fashion. For each captured image pair, the appropriate appearance and/or motion features are extracted and the corresponding cluster identity of current features is computed based on trained cluster centroids. We define an "empty scene" as the configuration without the user's hand in the vicinity of the object. We can check whether current frame is an empty scene by comparing the cluster identity of the current frame with that of an empty configuration. We begin the processing of a sequence when the system observes a non-empty scene, which means the hand has entered the vicinity of the object. We carry out the recognition of the current sequence and notify the user when a valid gesture is recognized. The recording of the current sequence is then terminated and the system enters a new cycle. Another case for ending the current sequence is that the system continuously observes the empty configuration for several frames.

4.2 Gesture Recognition Results

To perform training of the HMMs and the neural network, we record over 100 gesture sequences for each of the 6 gestures. A separate test set contains over 70 sequences of each gesture.

We carry out the training and testing on several feature sets. These different sets are characterized by the dimensionality of our 3-D gesture volume described in Section 2 and different combination of the appearance and motion cues.

In our experiment, we record the data as subimages around the manipulated object. The size of the area is 220×210 pixels. When computing the appearance volume using Equation 1, we split this local area into tiles of appropriate dimension according to the choice of the dimensionality of the appearance volume, i.e., m and n. For example, in the first appearance dataset, we choose $m = n = 5$. So the dimensionality of the image patches is $44(220/5) \times 42(210/5)$. For convenience, in all the datasets, we set $m = n = p$.

1. **Appearance Only** (125-D)
   In this set, we only use the appearance feature. We set $m = m = p = 5$ and thus the dimensionality is $5 \times 5 \times 5 = 125$.

2. **Appearance Only** (1000-D)
   Similar to the first set, except that the dimensionality of the appearance feature is $m(= 10) \times n(= 10) \times p(= 10) = 1000$.

3. **Motion Only** (1000-D)
   We compute the motion feature by taking the difference between two 1000-D appearance vectors.

4. **Concatenation of Appearance and Motion**
   In this set, we concatenate the 125-D appearance feature with the 1000-D motion vector to form a 1125-D vector.

5. **Combination of Appearance** (125-D) **and Motion**
   We carry out K-means on the 125-D appearance feature and 1000-D motion features separately. Then each frame is represented as a 2-D discrete vector containing both the appearance cluster identity and motion cluster character.

6. **Combination of Appearance** (1000-D) **and Motion**
   Similar to the previous setting except that we use the 1000-D appearance feature.

We perform the training and testing on these sets for the HMM models and the neural network. For the neural network, we align each gesture sequence to the fixed length of 20. For the HMM models, we also carry out comparison experiments between using the same aligned sequences as the neural network and applying the raw unaligned sequence. Table 2 shows the gesture recognition results for all the feature sets and both gesture models. For each model we report both the recognition accuracy on the training set and on the test set. We also present the number of clusters used to carry out the VQ.

Table 2. Gesture recognition results for different feature spaces

Set	Clusters	HMM		NN		Unaligned	
Appearance(125-D)	8	99.5	99.5	100.0	98.8	99.4	99.4
Appearance(1000-D)	8	99.5	100.0	98.4	94.4	98.4	98.0
Motion(1000-D)	15	98.4	98.1	97.7	86.3	97.9	98.8
Concatenation	18	98.9	99.0	98.9	87.7	96.7	96.1
Combination 1	120	100.0	100.0	100.0	96.6	98.2	97.3
Combination 2	120	99.8	99.8	99.8	97.1	99.2	99.5

The results show that aligning the sequences to the same length improves the recognition accuracy. It can also be seen that the motion feature alone seems to perform slightly worse than those with appearance cues. However, combining appearance features with the motion features achieves the best recognition accuracy for our current gesture set.

Another interesting comparison between the HMM model and neural network shows that our multilayer neural network tends to over-train on the feature sets. The neural network model achieves equivalent or higher accuracy on the training set as the HMM model, but performs worse on the test set. During the training of the HMMs, the Baum-Welch algorithm runs for less

than 5 iterations before the overall system entropy reaches a local minimum. During the neural network training process, the backpropagation algorithm typically runs for over 1000 iterations. We stop the the procedure when the decrease of the output error between consecutive runs is lower than a threshold, which is typically a very small number such as 0.00001.

Alternatively, one could stop the backpropagation algorithm interactively by measuring the performance on a validation set after each iteration and halting the training process if the classification on this validation set degenerates. However, we choose a fixed threshold to preserve the generality of the method and keep the training process automatic.

We also compare the gesture modeling using HMMs based on the raw sequences and those using collapsed sequences. Each raw sequence containing a gesture is packed in such a way that we only record a symbol if it is different from its previous one. In essence, we only record the order of the appearance of each feature, excluding the duration in the original temporal sequence. This is similar to the rule-based and state-based gesture modeling [2, 23]. Table 3 shows the gesture recognition results based on the datasets of collapsed sequences.

Table 3. Gesture recognition results for collapsed sequences

Feature Sets	Training	Test
Appearance(125-D)	89.3%	88.8%
Appearance(1000-D)	88.3%	86.1%
Motion(1000-D)	98.4%	96.6%
Concatenation	90.8%	89.0%
Combination 1	94.2%	96.8%
Combination 2	99.8%	98.8%

Compared to the results using raw sequences, the gesture recognition using collapsed sequences performs slightly worse. Still, for the combination of the appearance and the motion features, this scheme of gesture modeling based only on key frames achieves very good recognition performance.

We also carry out the HMM recognition experiments using different numbers of clusters for the VQ algorithm discussed in Section 3. Table 4 and Table 5 summarize the results. For the two datasets where we combine the clustering result of appearance and motion into a 2-D vector, in Table 5, we display the number of clusters for both the appearance and motion features in the format of (appearance clusters, motion clusters).

It can be seen that, the recognition accuracy generally increases with the growth of the number of clusters. The reason is that gestures are represented by trajectories in a feature space of higher dimensionality, so that more characteristic detail of the gestures will be modeled. However, when the number of clusters increases beyond a certain limit, the HMMs tend to overtrain on

Table 4. Gesture recognition results using different numbers of clusters

Set	4		8		16		32		64		128	
App.(125-D)	64.1	70.2	99.4	99.4	99.8	99.0	99.8	92.4	97.4	67.2	98.7	49.6
App.(1000-D)	83.3	83.6	98.9	98.3	99.5	97.8	97.9	92.4	98.7	58.7	99.8	27.1
Motion	64.1	74.2	93.8	94.6	99.2	97.3	99.5	98.0	100	94.6	100	83.9
Concat.	62.0	70.7	95.3	96.1	98.5	97.3	99.8	98.3	100	96.1	100	84.8

Table 5. Results for different combinations of appearance and motion features

Set	(4, 4)		(8, 4)		(8, 8)		(16, 8)		(16, 16)	
Combination 1	93.7	97.3	100	97.3	100	93.9	100	92.9	100	89.5
Combination 2	92.0	97.1	99.4	98.3	99.8	93.2	100	88.5	100	86.6

the training data and the accuracy on the test set deteriorates. A trade-off between training and testing accuracy must be made.

5 Conclusions

In this paper we present a novel real-time 3-D gesture recognition system that combines the 3-D appearance of the hand and the motion dynamics of the gesture to classify manipulative and controlling gestures. Instead of tracking the user's hand, we capture the 3-D appearance of the local volume around the manipulation subject. Motion is computed as the difference of the appearance features between frames. We reduce the dimensionality of the 3-D feature by employing unsupervised learning. We implemented a real-time system based on the 4-D Touchpad platform and tested the system using two different approaches to model the temporal gestures, forward HMMs and multilayer neural networks. By combining the appearance and motion cues, both HMM models and the neural network achieved an recognition accuracy of over 96%. The proposed scheme is a flexible and efficient way to capture 3-D visual cues in a local neighborhood around an object. The experiment results show that these local appearance and motion features capture the necessary visual cues to recognize different manipulative gestures.

In our current experiment setup, the manipulated objects are 2-D icons that lie on a 2-D plane. The geometry between the cameras and each object is similar and relatively fixed. The proposed appearance feature is not invariant to geometric transforms, such as rotation and translation. For general VEs, the interaction subjects can occupy a relatively large space, such that the geometry of each object with respect to the cameras can vary greatly. To overcome this variance, improvement to the feature extraction scheme is necessary.

The gesture vocabulary in our current experiment only consists of six dynamic gestures. In the future, we intend to address more complex gestures and a larger gesture vocabulary. We also plan to investigate other ways to model the gesture dynamics, such as HMMs that achieve minimal classification errors.

Acknowledgments

We are grateful to Darius Burschka for help with the Visual Interaction Cues project. This material is based upon work supported by the National Science Foundation under Grant No. 0112882. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

1. Vassilis Athitsos and Stan Sclaroff. Estimating 3D Hand Pose from a Cluttered Image. In *Computer Vision and Pattern Recognition*, volume 2, pages 432–439, 2003.
2. Aaron Bobick and Andrew Wilson. A State-based Approach to the Representation and Recognition of Gesture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(12):1325–1337, 1997.
3. Jason J. Corso, Darius Burschka, and Gregory D. Hager. The 4DT: Unencumbered HCI With VICs. In *Proceedings of CVPRHCI*, 2003.
4. James Davis and Aaron Bobick. The Representation and Recognition of Action Using Temporal Templates. In *Computer Vision and Pattern Recognition*, pages 928–934, 1997.
5. Jonathan Deutscher, Andrew Blake, and Ian Reid. Articulated Body Motion Capture by Annealed Particle Filtering. *Computer Vision and Pattern Recognition*, 2, 2000.
6. Richard Duda, Peter Hart, and David Stork. *Pattern Classification*. John Wiley and Sons, Inc, 2001.
7. Graham D. Finlayson, James Crowley, and Bernt Schiele. Comprehensive Colour Image Normalization. In *Proceedings of the European Conference on Computer Vision*, number 1, pages 475–490, 1998.
8. D. Gavrila. The visual analysis of human movement: a survey. *Computer Vision and Image Understanding*, 73:82–98, 1999.
9. Theo Gevers. Color based object recognition. *Pattern Recognition*, 32(3):453–464, 1999.
10. Frederick Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, 1999.
11. S. Malassiotis, N. Aifanti, and M. Strintzis. A Gesture Recognition System Using 3D Data. In *Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission*, pages 190–193, 2002.
12. Kenji Oka, Yoichi Sato, and Hideki Koike. Real-Time Fingertip Tracking and Gesture Recognition. *IEEE Computer Graphics and Applications*, 22(6):64–71, 2002.

13. Vasu Parameswaran and Rama Chellappa. View Invariants for Human Action Recognition. In *Computer Vision and Pattern Recognition*, volume 2, pages 613–619, 2003.
14. F. Quek. Unencumbered Gesture Interaction. *IEEE Multimedia*, 3(3):36–47, 1996.
15. Lawrence Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
16. Aditya Ramamoorthy, Namrata Vaswani, Santanu Chaudhury, and Subhashi Banerjee. Recognition of Dynamic Hand Gestures. *Pattern Recognition*, 36:2069–2081, 2003.
17. J.M. Rehg and T. Kanade. Visual tracking of high DOF articulated structures: An application to human hand tracking. In *Computer Vision – ECCV '94*, volume B, pages 35–46, 1994.
18. Christopher Wren, Ali Azarbayejani, Trevor Darrell, and Alex Paul Pentland. Pfinder: Real-time tracking of the Human Body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):780–784, 1997.
19. D. Scharstein and R. Szeliski A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *International Journal of Computer Vision*, 47:7-42, 2002.
20. T. Starner and A. Pentland. Real-time american sign language recognition from video using hidden markov models. Technical Report TR-375, M.I.T. Media Laboratory, 1996.
21. Andries van Dam. Post-wimp user interfaces. *Communications of the ACM*, 40(2):63–67, 1997.
22. Christian von Hardenberg and Francois Berard. Bare-Hand Human-Computer Interaction. In *Workshop on Perceptive User Interfaces*, 2001.
23. Ying Wu and Thomas S. Huang. Hand Modeling, Analysis, and Recognition. *IEEE Signal Processing Magazine*, 18(3):51–60, 2001.
24. Junji Yamato, Jun Ohya, and Kenichiro Ishii. Recognizing Human Actions in Time-sequential Images Using Hidden Markov Model. In *Computer Vision and Pattern Recognition*, pages 379–385, 1992.
25. Guangqi Ye, Jason J. Corso, Darius Burschka, and Gregory D. Hager. VICs: A modular vision-based hci framework. In *Proceedings of 3rd International Conference on Computer Vision Systems(ICVS 2003)*, pages 257–267, 2003.
26. Guangqi Ye and Gregory D. Hager. Appearance-based visual interaction. Technical report, 2002. CIRL Lab Technical Report, Department of Computer Science, The Johns Hopkins University.
27. Guangqi Ye, Jason J. Corso, Darius Burschka, and Gregory D. Hager. VICs: A Modular HCI Framework Using Spatio-Temporal Dynamics. *Machine Vision and Applications*, 16(1), pages 13-20, 2004.
28. Zhengyou Zhang, Ying Wu, Ying Shan, and Steven Shafer. Visual Panel: Virtual Mouse Keyboard and 3D Controller with an Ordinary Piece of Paper. In *Workshop on Perceptive User Interfaces*, 2001.