Iranian Journal of Science & Technology, Transaction B, Engineering, Vol. 32, No. B3, pp 279-293
Printed in The Islamic Republic of Iran, 2008

VISUAL-BASED INTERFACE USING HAND GESTURE RECOGNITION AND OBJECT TRACKING*

A. CHALECHALE[1**] AND F. SAFAEI[2]

[1]Dept. of Computer, Faculty of Technology, Razi University, Kermanshah -67149, I. R. of Iran
Email: chalechale@razi.ac.ir
[2] School of Electrical Computer and Telecom. Eng., University of Wollongong NSW, Australia

Abstract– This paper presents a novel approach for human-machine interface using visual-based communication. Scene analysis, object recognition and tracking, together with gesture detection and classification, are employed to design and implement a system for real time interaction. A new paradigm is proposed for efficient selection of hand gestures that can be used in other gesture-based environments aiming at interactive multimedia. An inexpensive web camera and readily available tools (such as an ordinary pen) have been used in this approach to render the system cost effective and suitable for general and home-based multimedia applications. Focus is placed on real time applications with a fast decision function for gesture classification, and a model-based approach for orientation detection. Experimental results demonstrate the efficacy of the proposed approach when tested using different performance criteria.

Keywords– Interactive multimedia, visual-based interface, scene analysis, gesture recognition, object tracking

1. INTRODUCTION

Visual-based interface plays an important role in new technologies including virtual reality, robotics, medical engineering, and digital games. The Internet and World Wide Web dramatically extend the scope of these technologies and make them accessible to a rapidly growing audience. As a result, researchers are exploring web-based disciplines such as distributed virtual reality, telrobotics, telemedicine, and networked games. The application of a visual-based interface in a web-based environment needs special consideration, mostly in the areas of (a) rapid detection and recognition of the scene, where user's input (gesture) is created, and (b) fast communications through the network nodes, where the result is visualized [1, 2] . This paper focuses on the former task, that is, the design and analysis of a set of visual inputs for controlling a general man-machine interface.

Relying on a vision-based system and a new range of gesture transmitting devices, a computer can recognize users' gestures and perform appropriate actions. In this type of interface, gestures are used as commands in the virtual world, or in computer games. Here, instead of pressing keys on a keyboard, moving a mouse, or using a joystick or trackball, human's body gestures are used to create appropriate commands for a running program. In fact, the new proposed interface translates users' gestures into the users' intentions as the input to the system. Interaction is either through images, which are motion pictures taken of the user's gesture, or through a wearable device such as a marked glove, a wireless motion transmitter [3], a gesture pen [4], or an iGesture Pad [5]. It is interesting to note that a specific gesture may invoke different actions in different environments based on the interpretation defined in the driving engine. For example, turning down the thumb may cause a gun to shoot in a fighting game, while it can

*Received by the editors May 6, 2007; final revised form January 8, 2008.
**Corresponding author

cause a plane in a flight simulator to turn right.

Visual commands, the core concept of this paper, can be divided into two components: hand gesture and sign object. The former generates a specific command and the latter designates an orientation. They form a novel visual input device where the user can simultaneously issue visual commands and/or specify directions. These are similar to the main functions of a mouse, where users are able to press one of the buttons and/or move the mouse in one direction.

Several aspects of directing computers using human gestures have been studied in the literature; however gesture recognition is still an open problem. This is due to significant challenges in *response time, reliability, economical constrains*, and *natural intuitive gesticulation* restrictions [6]. The MPEG-4 standard has defined Facial Animation Parameters to analyze facial expressions and convert them to some predefined facial actions [7]. Jian *et al*. [8] have developed a lip tracking system using lip contour analysis and feature extraction. Similarly, human leg movement has been tracked using color marks placed on the shoes of the user to determine the type of leg movement using a first-order Markov model [9].

There are a number of works on sign language (hand alphabet) recognition. For example, Birk *et al*. [10] have proposed using principal component analysis (PCA) to extract features from images of hand gestures. The feature selection and generation of a classifier are performed off-line, while unknown gesture images are processed on-line. Since the PCA approach is a data driven method it is sensitive to changes in the pose of a gesture. Although normalization can almost solve this problem, the system is still sensitive to rotations orthogonal to the camera axis and also to a significant change in the way a gesture is made. Over the last decade a research team at the University of Western Australia has focused on a project to translate Australian sign language, called Auslan, to English, and vise versa [11]. Translation from English to Auslan requires: speech recognition, language mapping from English to Auslan, and the graphical animation of the Auslan signs. On the other hand, the reverse translation from Auslan to English requires: gesture recognition, language mapping from Auslan to English, and the synthesis of English speech or a text display. A color glove has been used to facilitate gesture recognition.

Continuous hand gesture recognition requires the detection of gestures in a video stream and their classification. Two continuous recognition solutions based on HMM are compared in [12]. The first approach uses a motion detection algorithm to isolate gesture candidates, followed by a HMM recognition stage. The second approach is a single-step HMM-based spotting method improved by a new implicit duration modeling.

Experimental results on continuous video data containing 41 different types of hand gestures show that the second approach provides better recognition results.

This paper presents a novel visual-based approach for multimedia interaction. In this approach hand gestures, together with the direction of a simple pointing arrow, are recognized by video analysis. Focus is placed on real-time applications where the user's input must be recognized in a limited time. A fast decision function has been derived for the gesture classification stage, while model-based reasoning is used for the direction detection. Both stages are conducted simultaneously and a pair of alpha-numeric codes is extracted from each commanding frame. The alphabetic portion indicates hand gesture while the numeric portion denotes the direction. The meaning of the hand gestures is application dependant and the direction of the pointing device is quantized into a set of predefined values based on user requirements. These make the approach more flexible and easy to incorporate into a wide range of interactive multimedia applications. Moreover, the presented approach has the ability of coarse-to-fine adjustment, which can be used in broad band communications considering dynamic network constrains. The proposed system has been developed using an ordinary and inexpensive web camera, which is more versatile and acceptable in general and home-based applications.

The rest of the paper is organized as follows: Section 2 explains our approach in detail. Section 3 presents experimental results and discussion. Finally, Section 4 concludes the paper and poses some new research directions.

2. SCENE ANALYSIS FOR HAND GESTURE RECOGNITION AND OBJECT DIRECTION DETECTION

This section explains a novel and fast video analysis approach for gesture recognition. To expand the scope of visual commands delivered at a time, hand gesture is augmented by another object, in this case an arrow or a pen to communicate additional information such as direction. Hence, we assume a sequence of frames containing a hand's paw, a pointing arrow (sign object), and some extra objects (see Fig.1).

Generally, the proposed approach has two stages, off-line and on-line. In the first stage a learning set is employed to extract scene features. In the on-line stage an unknown image is analyzed to recognize the involved gesture and the direction of a pointing device.

The scene is called *hand & arrow*, where the hand can pose any gesture similar to those depicted in Fig. 2 (known as international sign language hand alphabet).

Moreover, the pointing arrow can be placed in any direction between 0 and 360 degrees in a circular manner. Extra objects could be anything non-similar to the paw and the pointing device (a pen in this case). The pointing device needs to be distinguishable at least by different thicknesses (such as different ends in a regular pen or pencil) or by putting a marker on one end. This is to make a distinguishable arrow shape. For a system to recognize a sequence of these gestures, it must be able to determine (a) which gesture the user's hand is posing, and (b) which direction the arrow is pointing to.

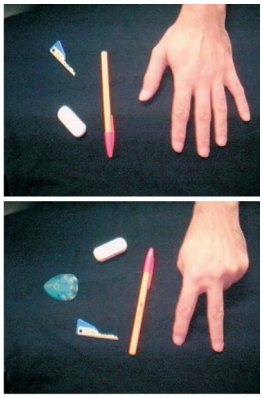

Fig. 1. Examples of objects in the scene

The arrow is carried by hand and when it reaches a desired direction, the hand poses a gesture. The combination of the hand gesture and the direction of the arrow constitute a command. The former is coded by an alphabet letter (A to Z), and the latter by a number (0 to 360). Fig. 3 shows a sequence of frames starting from a command, called the *commanding frame*, and ending with another commanding frame. All frames in between are *moving frames*, where there is no command involved. Upcoming frames are first analyzed to determine whether they are moving or commanding frames. Commanding frames are passed for further analysis of command extraction, whilst moving frames are discarded. The video sequence commences with a *starting* command and ends with a *stopping* command.

Since four phases of starting, moving, commanding, and stopping occur in a predictable order, a finite state machine can be used to guide the flow (see Fig. 4). Arrows in the state diagram represent primary frame analysis, which cause a state transition or remaining in the same state based on the result of the analysis. In the diagram S0 corresponds to *start* state, S1 and S2 to *moving* and *commanding* states,

respectively and S3 corresponds to *end* state. Command extraction requires further processing of the hand gesture and arrow object, which are discussed in the following sections.

a) Hand gesture selection and recognition

To explain the proposed scheme we utilize a collection of 5000 hand gestures and show how the approach works on this collection. The procedure can be adopted for other collections without any need to change its general structure. The collection is made by taking frames from videos recorded from a wide range of subjects including, adults, children, men, and women. Initially, the collection is grouped into a 25-hand alphabet. The images are color, in JPEG format, on a dark background. Fig. 2 shows representative gestures and Fig. 5 depicts some examples of the images.

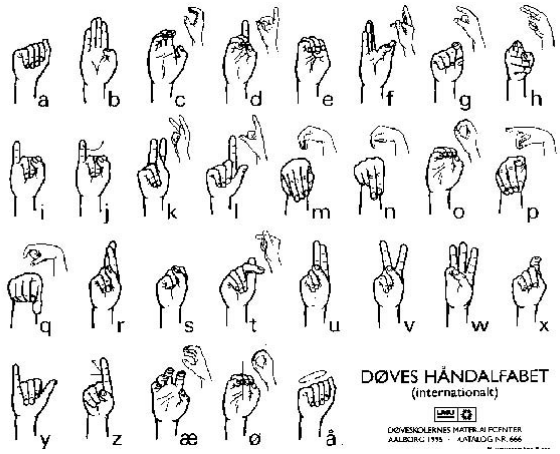

Fig. 2. International sign language hand alphabet [10]

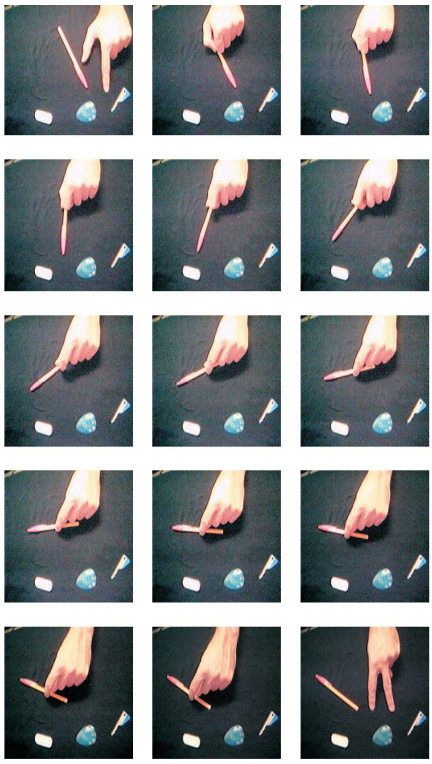

Fig. 3. A sequence of frames starting with a command (upper leftmost frame), some moving frames and ending with another command (lower rightmost frame)

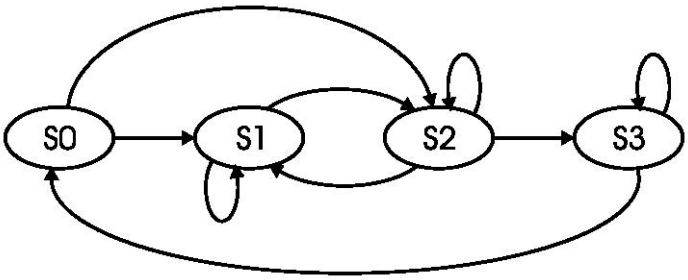

Fig. 4. State transition diagram. States are defined as: S0: Start S1: Move S2: Command S3: End. Transitions are determined by scene analysis algorithm

The full color frame is converted to a gray intensity image by eliminating the hue and saturation components while retaining the luminance. Due to varying lighting conditions of the images within the database, using a unique threshold to binarize images is inadequate. Fig. 6 shows instances where a unique threshold causes inappropriate segmentation of the hand shape. To avoid this, K-mean clustering is employed for binarization in the pre-processing stage [13]. This successfully segments hand gestures from the background (see Fig. 6).

Size normalization using nearest-neighbor interpolation is applied next. This is to achieve scale invariance property, which allows different size gestures to have similar features. The bounding box of the region of interest is found first and then normalized to w × h pixels (128 × 128 pixels in our experiments).

Next, for each segmented-normalized gesture g belonging to a gesture group G_i, i = 1 ...I, we extract J shape properties P_j, j =1 ...J . Currently, for the hand collection, I is 25 and J is chosen to be 14 corresponding to 25 gesture clusters and 14 predominant gesture properties, respectively. The properties include seven geometric and seven invariant moment-based functions. Geometric properties are: area (ar), perimeter (pr), major axis length (mj), minor axis length (mi), eccentricity (ec), and the ratio of ar/pr, and mj/mi.

The invariant moment-based functions have been widely used in a number of applications [14, 15, 16]. The first six functions ($\phi_1 - \phi_6$) are invariant under rotation and the last one (ϕ_7) is both skew and rotation invariant. They are based on the central i,j-th moments (μ_{ij}) of a 2D image f(x, y), which are defined as follows:

$$\mu_{ij} = \sum_x \sum_y (x-\bar{x})^i (y-\bar{y})^j f(x,y) \tag{1}$$

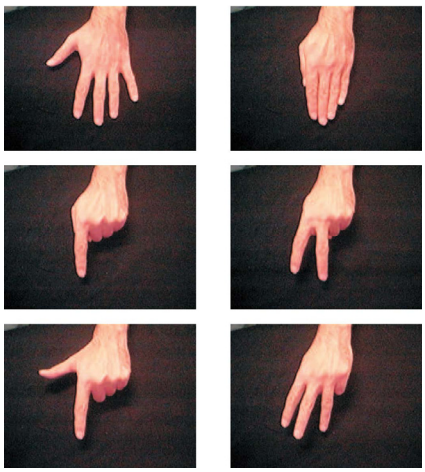

Fig. 5. Hand gesture examples

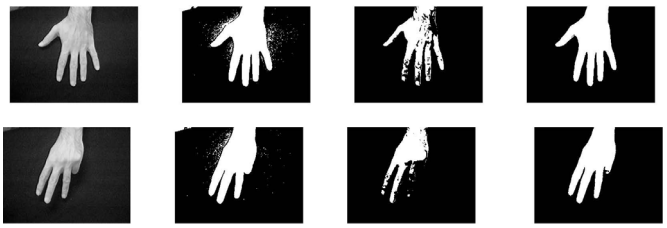

Fig. 6. Instances of gray scale images (first column) where lower thresholds (second column) make many unwanted noisy regions and higher thresholds (third column) destroy the hand region, while K-means clustering (last column) segments hand region properly

Then, the invariant moment-based functions are defined as

$$\begin{aligned}
\phi_1 &= \eta_{20} + \eta_{02} \\
\phi_2 &= (\eta_{20} + \eta_{02})^2 + 4\eta_{11}^2 \\
\phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
\phi_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \\
\phi_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12}) .[3(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03}) \\
\phi_6 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \\
\phi_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12}) - (\eta_{30} - 3\eta_{12})(\eta_{31} + \eta_{03})
\end{aligned} \tag{2}$$

Where $\eta_{ij} = \dfrac{\mu_{ij}}{\mu_{00}^{\gamma}}$ and $\gamma = \dfrac{i+j}{2} + 1$.

To determine the recognition power of each G_i cluster, we exploit a classification scheme using the properties P_j . Subsequently, we try to classify 500 gestures (20 of each group) into the associated groups.

Here, the average ratio of correctly classified gestures to all applied gestures multiplied by 100 is defined as the *Recognition Rate* (for each gesture G_i using property P_j). That is:

$$R_{ij} = \left(\frac{1}{I}\sum_{i=1}^{I} \frac{\text{number of correctly classified gesture } G_i \text{ using } P_j}{\text{total number of applied } G_i}\right) * 100$$

These recognition rates R_{ij} for i =1 ...I and j =1 ...J are obtained and saved in appropriate entries in a *cluster-property* matrix. The matrix is of size I*J whose rows correspond to gestures G_i and its columns correspond to properties P_j

The classification is based on the Bayesian rule assuming Gaussian distribution for the hand gesture patterns. This assumption is based on previous research reported in [17, 10]. To extract a decision function for our classifier, we consider J number of 1D probability density functions. Each function involves I pattern groups governed by Gaussian densities, with means m_{ij} and standard deviation σ_{ij}. Therefore, the Bayes decision function has the following form [18]:

$$d_{ij}(g) = p(g / G_i)P(G_i) \tag{3}$$

that is identical as

$$d_{ij}(g) = \frac{1}{\sqrt{2\pi}\sigma_{ij}} e^{[-\frac{(g-m_{ij})^2}{2\sigma_{ij}^2}]P(G_i)} \tag{4}$$

for i =1 ...I and j =1 ...J, where $p(g/G_i)$ is the probability density function of the gesture pattern g from cluster G_i and $P(G_i)$ is the probability of occurrence of the corresponding cluster.

Based on the hypothesis that there is no favorite gesture for the user, we can assume an equally likely

occurrence of all classes. Based on this assumption we can conclude that

$$P(G_1)= P(G_2) \cdots = P(G_i) \cdots = P(G_I)=1/I$$

Because of the exponential form of the Gaussian density, which persuade the use of a natural logarithm, and since the logarithm is a monotonically increasing function, the decision function in Eq. (4) can be modified to a more convenient form. In other words, based on the aforementioned assumption and facts, we can use the following decision function, which is less computationally expensive and much faster for the classification of hand gestures:

$$d_{ij}(b) = \ln[p(g/G_i)P(Gi)] = \ln p(g/G_i) + \ln P(Gi) \tag{5}$$

Considering Eq. (4), it can be written as

$$d_{ij}(g) = -\frac{1}{2}\ln 2\pi - \ln\sigma_{ij} - \frac{(g-m_{ij})^2}{2\sigma_{ij}^{\ 2}} + \ln P(G_i) \tag{6}$$

Dropping the constant values $-\frac{1}{2}\ln 2\pi$ and $\ln P(G_i)$, which have no effect on the numerical order of the decision function, a new expeditious decision function is obtained as

$$\hat{d}_{ij}(g) = -\ln\sigma_{ij} - \frac{(g-m_{ij})^2}{2\sigma_{ij}^{\ 2}} \tag{7}$$

for i =1 ...I and j =1 ...J , where m_{ij} and σ_{ij} are the mean and standard deviation of gesture group G_i using property P_j , and g is the corresponding scalar property of an unknown gesture.

Utilizing the above classification approach we calculate recognition rates R_{ij} for each single-valued property P_j and for each gesture group G_i and save them in the crossing cells of the corresponding rows and columns of the *cluster-property* matrix.

Moreover, to conduct multi-valued features and compare the effectiveness of combined features, different sets of features are generated and used as the feature vector for gesture classifications. Among many possible combinations, a set of 18 experimentally chosen features are selected.

In other words, to appraise a combinatory analysis and depict an efficient feature vector to be used for gesture recognition, a set of K = 18 different combinations of the geometric properties and invariant moment-based functions is generated and recognition rates are obtained. The set are extracted experimentally. Since the properties are multiple-valued, the decision function for the classification is obtained using feature vectors. In this case, the Gaussian density of the vectors in the i^{th} gesture class has the form

$$p(\xi / G_i) = \frac{1}{2\pi^{n/2}\left|C_{ik}\right|^{1/2}} e^{\left[-\frac{1}{2}(\xi - m_{ik})^T C_{ik}^{-1}(\xi - m_{ik})\right]} \tag{8}$$

for k =1, 2,...K, where ξ is the extracted feature vector of an unknown gesture and n is the dimensionality of the feature vectors, and |·| indicates matrix determinant. Note that each density is specified completely by its mean vector m_{ik} and covariance matrix C_{ik}, which are defined as

$$m_{ik} = E_{ik}\{\xi\} \tag{9}$$

and

$$C_{ik} = E_{ik}\{(\xi - m_{ik})(\xi - m_{ik})^T\} \tag{10}$$

where Eik{·} denotes the expected value of the argument over the gestures of class G_i using multiple-valued property P_k. Approximating the expected value E_{ik} by the average value of the quantities in question yields an estimate of the mean vector and covariance matrix as

$$m_{ik} = \frac{1}{N_i} \sum_{\xi \in G_i} \xi \tag{11}$$

and

$$C_{ik} = \frac{1}{N_i} \sum_{\xi \in G_i} (\xi \xi^T - m_{ik} m_{ik}^T) \tag{12}$$

where N_i is the number of gesture vectors from class G_i and summation is taken over those vectors for k =1, 2,...K.

Following a similar approach applied in single-valued features (Eqs. (5-7)), it is possible to obtain a simple and expeditious decision function as

$$\hat{d}_{ik}(\xi) = -\ln|C_{ik}| - (\xi - m_{ik})^T C_{ik}^{-1} (\xi - m_{ik}) \tag{13}$$

for i =1 ...I and k =1 ...K. Note that C_{ik} values are independent of the input ξ, which means they can be calculated off-line and saved in a look-up table. They are fetched (instead of computed) from the look-up table at the on-line stage to accelerate the decision making process.

The diagonal element c_{rr} is the variance of the r^{th} element of the gesture vector and the off-diagonal element c_{rs} is the covariance of x_r and x_s. When the elements x_r and x_s of the feature vector are statistically independent, c_{rs} =0. This property has been used to identify autonomous features and to pick them in a combination of features in multiple-valued properties. Interestingly, this fact renders the multivariate Gaussian density function to the product of the univariate density of each element of ξ vector when the off-diagonal elements of the covariance matrix C_{ik} are zero. This, in turn, expedites the generation of the look-up table.

The recognition rates R_{ik} for i =1 ...I and k =1 ...K are calculated utilizing Eq. (14) and saved in appropriate entries in another structure called *cluster-features* matrix. This represents not only the distinguishability of the isolated hand gestures, but also the recognition power of different sets of features to describe gestures.

The general paradigm explained above provides a straightforward method to select distinguishable gestures and has been shown to be effective in experimental results (next section). More importantly, column-wise summations in the *cluster-property* and *cluster-features* matrices indicate the recognition power of the simple properties and complex features respectively. On the other hand, row-wise summations exhibit the discrimination power of each gesture, an important clue to the selection of gestures for the *hand & arrow* application.

1. Object Extraction: Once a scene image has been segmented into a number of regions using K-mean clustering, thresholding or another method, it is necessary to identify each of the connected components in the image. The process of assigning each distinct region unique identifiers is called blob coloring. As there are several objects in the gesture scene, we need to do image segmentation first and then recognize the hand gesture and determine the direction of the arrow. Image segmentation is fulfilled by K-mean clustering (without any predefined threshold) and then blob coloring (sometimes called connected component labeling [19]) is used to identify objects.

To explain this procedure, let B be the binary image of the original full colored scene obtained by K-mean clustering of the gray scaled image. The connected components labeling algorithm that performs the

unit change from pixel to region is employed to label B as follows:

1. All pixels that have the value binary 1 and are connected to each other in an 8-connectivity neighborhood, are given the same identification label. The label is a unique index of the region to which the pixels belong.
2. For efficient implementation of the labeling algorithm we run-length encode the input image B.
3. Assign preliminary labels while scanning the runs and recording label equivalences in a local table.
4. The equivalence classes are resolved next and the runs are relabeled based on the resolved equivalence classes.

Geometric properties including area, circularity, eccentricity, position and size of different labeled regions can be exploited to extract region of interest. For the *hand & arrow* scene, experimental results (Section 3) show that area and circularity properties are adequate to successfully distinguish the hand and arrow in most cases. This prominently

Speeds up the scene analysis procedure. The next section explains the procedure of how to determine the direction of the arrow in the scene.

2. Detection of Arrow's Direction: When the region of the arrow is obtained using the connected component analysis explained above, a sequence of processes depicted in Fig. 7 is employed to determine its direction. First, the major axis of an ellipse that has the same second-moments as the region is obtained. Orientation O of the arrow is the angle between the x-axis and the major axis of the ellipse. Next, we need to determine the exact direction of the arrow since O is identical for two opposite directions in the same orientation, which is ambiguous and needs to be resolved. Consequently, to determine exact direction in a whole 360 degree circle, the region of interest is rotated so that the arrow is aligned horizontally. This implies a rotation angle of −O to be applied to the arrow's region. Based on the assumption that there is a difference in the thickness of the arrow at its two ends, we then divide the bounding box of the aligned region into two equal regions along the x-axis. In other words, we cut off the aligned region at the middle and calculate the area of each half by simply counting the number of foreground pixels. This explores the arrow's head, which is exploited to determine the direction of the arrow. For example, assuming the head is thicker than the tail of the arrow, if the orientation O of the arrow is calculated to be 30 degrees, and if the area of the first half of the aligned region is greater than the area of the second half, then the arrow is pointing to the direction of 210 degrees; or having O = −40 degrees, and if the area of the second half is greater than the first half this indicates an arrow direction of 320 degrees.

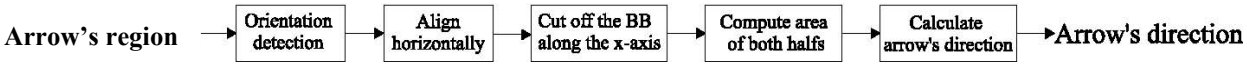

Fig. 7. Block diagram of the direction detection procedure

3. EXPERIMENTAL RESULTS AND DISCUSSION

To evaluate the efficacy of the proposed approach, we conducted several experiments including off-line gesture selection, feature extraction, real-time frame grabbing, scene analysis, gesture recognition, and direction detection. We used a cost effective USB web camera with a 30-Hz frame rate and a resolution of 320 × 240 pixels. This section presents results in three different areas: gesture selection and recognition, arrow's direction detection, and overall real-time analysis.

a) Gesture selection and recognition

As stated before, a collection of 5000 hand gestures is used for the training phase of the classification of hand gestures. The collection is created by extracting still frames from several video sequences. Frames in the collection include 25 sets of gestures having a number of members from 60 to 200. In the training stage, the statistical model parameters which include means and standard deviations (scalars) for individual properties, and means (vectors) and covariance matrices for combined features, were obtained. To evaluate models, another 500 frames were used as the test data and we tried to classify them in one of the 25 classes using the extracted models.

For each gesture in the test set, for both singular properties and feature vectors, recognition rates were obtained. They were recorded in the appropriate entries of the corresponding matrix as explained in Section 2.1. The rates are used to evaluate a specific gesture based on its geometric properties and the extracted feature sets, respectively. The recognition rate in each entry in the *cluster-property* matrix is the number of correctly classified gestures divided by the number of inputs. For example, if 12 out of 20 input gestures in the cluster G_{10} are correctly classified by the decision function given in Eq. 7 using perimeter property into the same cluster, then the recognition rate in row G10, column *pr* of the *cluster-property* matrix is calculated to be 12/20=60%. In this part, 14 individual properties (7 geometric and 7 invariant-based functions) were examined for the 25 gesture groups. To be able to compare the recognition power of different properties, an overall recognition rate was obtained for each column of the matrix by simply averaging the recognition rates recorded in the column. The overall results show that the top three best singular properties are *mj*, *mi*, and *ar/pr*. These properties, which have been determined at the off-line process, have been used for the real-time classification of the hand gestures in the analysis of the *hand & arrow* scenes.

The top ten best distinguishable gestures, which are explored using row-wise averaging of the recognition rates in the *cluster-property* matrix, are chosen for generating visual commands. In other words, the *commanding frames* contain one of these hand gestures, an arrow pointing to a direction between 0 and 360 degrees, and some extra objects non-similar to palm of the hand and the pointing device.

Next, we tried to classify test gestures using 18 combinatory feature sets. The recognition rates were obtained using the decision function in Eq. (14) and the results were saved in the cluster-feature*s* matrix, which currently, in our experiments, has 25 rows and 18 columns. The rows correspond to hand gesture clusters and the columns correspond to a variety of combination features (feature vectors). The number of entries in the feature vectors vary from two to seven. Noting that there are a massive number of different combinations, we chose only those properties which were shown to have better discriminating power using the cluster-propert*y* matrix. These properties have tentatively been selected based on their independent characteristics using covariance matrices.

Moment-invariant functions showed a lack of efficacy, while different combinations of geometric properties exhibited higher recognition rates. The overall recognition rate of 98.7% is obtained in this test using a five-entry feature vector {mj, mi, ec, ar, pr}.

b) Arrow's direction detection

The arrow's direction was detected using the procedure explained in Section 2.2 and quantized into 12 equal buckets (30 degree slices). The connected component analysis approach for object extraction and the arrow's direction detection method (Section 2.2) were tested off-line using 10000 frames. The frames included (a) a hand moving the arrow, and (b) a hand posing a gesture while the arrow is placed in a specific direction. There were other objects in the scene (two to five), while applying different lighting

conditions within a homogeneous background. Here, the results were very promising where in 98.6% of cases the system recognized between *moving* and *commanding* frames, and in 97.7% of the *commanding* frames the direction of the arrow was determined correctly. Lightening conditions adversely affect the recognition process and cause the system to fail in some cases. This allows segmentation and object extraction to be fooled and results in an inappropriate portion of the scene to be considered as the pointing arrow. Consequently, as the key feature to distinguish between *moving* and *commanding* frames is a hand carrying pointing device, the frames are mixed up and the system becomes confused. It also causes the direction detection procedure process to be in an incorrect region and wrong direction is calculated. This signifies the importance of lighting conditions which any visual-based interface should consider.

Command frame detection, which plays an important role in processing the frames, is based on the fact that in such frames the hand is not attached to the pointing device, while in moving frames the hand carries the device. Connected component analysis has been used to distinguish between commanding and moving frames.

It is worthy to note that the algorithm for the detection of arrow direction has been evaluated for different kinds of similar arrows (i.e.; pens and pencils), however it is an open research area to examine other pointing devices.

Obviously, in a non-homogeneous background the object extraction approach should be enriched using other information such as size, shape, illumination conditions, or meta-data pertaining to the scene and the objects.

c) Overall real-time analysis

In another test, we tried to do an overall evaluation of the real-time analysis of the emergence of 30 different visual commands by 7 different subjects using han*d* & arro*w* scenes. Here, we asked subjects to perform visual commands by their hand gestures and point the arrow to a desired direction as fast as they can. We tried to follow users' commands and to recognize them. Here, we used a feature vector of 3 elements: ar, pr, mj. This is because we intend to have a low computation for gesture detection and have time to process all incoming frames. Whilst the subjects are performing the requested task, our system is acquiring frames. Following the finite state machine, it tries to (a) distinguish moving and commanding frames, and (b) recognize user's command. This test includes more than 24 minutes of video as each subject was asked to perform at least 3 minutes of interaction with the system. Consequently, $24 \times 60 \times 30 = 43200$ frames were analyzed in a real-time environment.

To accurately evaluate the overall performance of the proposed approach, four different performance criteria have been defined. These are (1) distinguish between moving and commanding frames rate (DMCR), (2) hand gesture recognition rate (HGRR), object detection and classification rate (ODCR), and (3) arrow's direction detection rate (ADDR). They are calculated using the following formulas

$$\begin{aligned}
\text{DMCR} &= \frac{\text{total number of correctly distinguished frames}}{\text{total number of frames}} * 100 \\
\text{HGRR} &= \frac{\text{number of correctly recognized gestures}}{\text{total number of command frames}} * 100 \\
\text{ODCR} &= \frac{\text{number of correctly recognized objects}}{\text{total number of objects in all frames}} * 100 \\
\text{ADDR} &= \frac{\text{number of correctly detected arrow directions}}{\text{total number of commanding frames}} * 100
\end{aligned} \tag{14}$$

Figure 8 depicts the DMCR, HGRR, ODCR, and ADDR factors computed for our proposed approach. As can be seen, the accuracy percentages for DMCR and ADDR are reasonable (97.4% and 96.6% respectively). Although we have only an exploited area and circularity properties for object recognition, the object detection and classification rate is 99.3%, which is obviously high. This is due to the existence of dissimilar objects in the scene. In many real-world applications there are different objects in the scene which can easily be distinguished using a few characteristics. However, for more similar objects we can employ more properties including eccentricity, size, position, color, texture, and shape characteristics for recognition. As the definition of "recognition rate" implies, the percentage of correctness is reported here, so the false rates in all criteria (DMCR, HGRR, ODCR, ADDR) are obtained by 100 – (the criterion).

Hand gesture recognition rate is the lowest rate (95.3%). We also attempted to count the number of users' commands that the system lost. More importantly, the system ignored only 1.73% of the commands. Low hand gesture recognition rate (HGRR) is due to low computation applied for gesture detection, which speeds up scene analysis and causes a very low rate of missing frames.

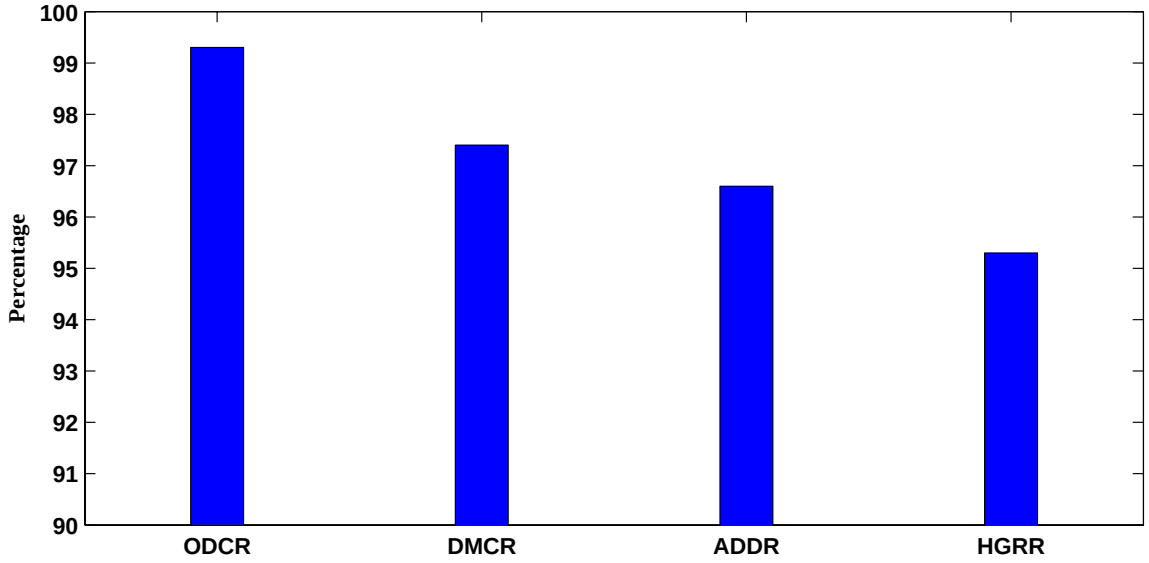

Fig. 8. Performance criteria percentages for the proposed approach. Abbreviations stand for: Object Detection and Classification Rate (ODCR), Distinguish between Moving and Commanding frames Rate (DMCR), Arrow's Direction Detection Rate (ADDR), and Hand Gesture Recognition Rate (HGRR)

To show the Gaussian property of the selected features, we obtained the area of 100 gestures and they are depicted in Fig. 9. As can be seen from the figure, the area of a hand gesture is changing around a center. Moreover, to more precisely show the Gaussianity of the area property, a histogram is also created and shown in Fig. 10.

As a final remark, in some cases the system counted one command more than one time. This is due to the fact that the user was slow or intentionally wanted to issue one command several times. This feature of the system is tolerable in many applications, e.g., in a shooting game it may cause uninterrupted gun fire, which is desirable. Otherwise, by setting an option, the system can simply be modified to ignore multiple identical commands, or a time-based constraint can be applied for state change.

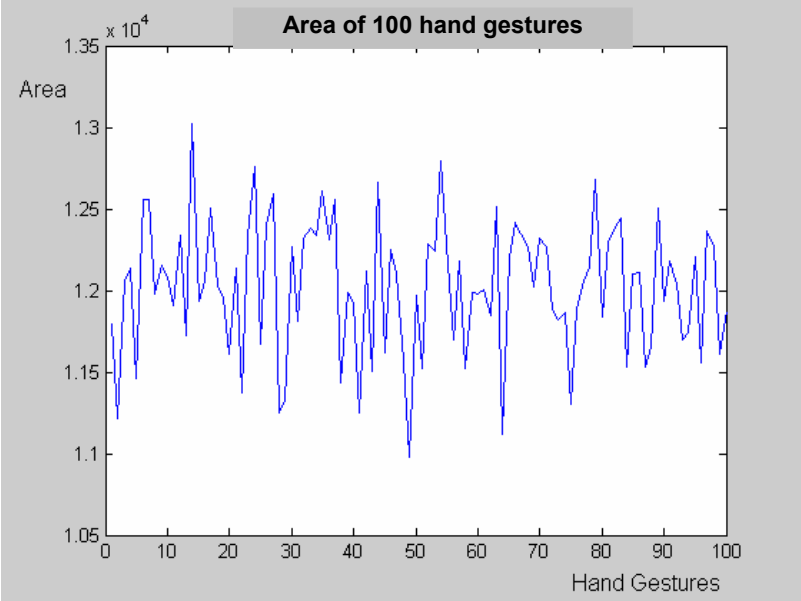

Fig. 9. Area of 100 hand gestures in number of pixels

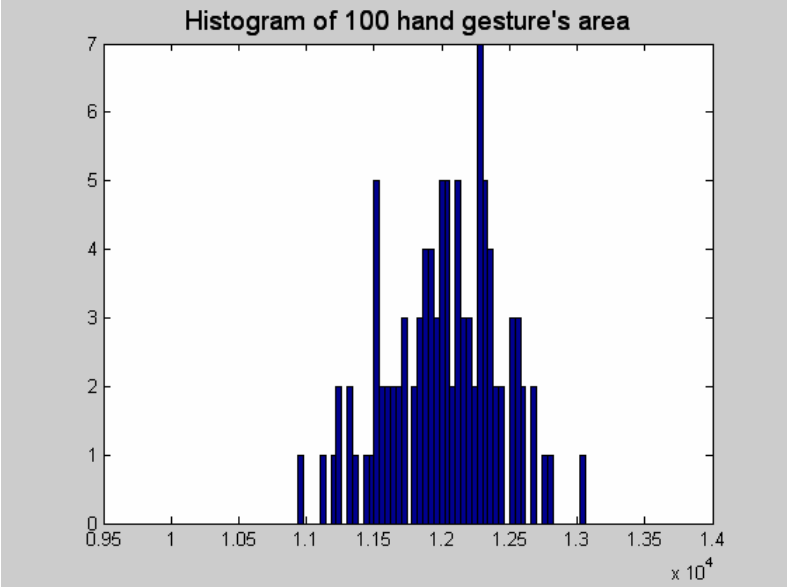

Fig. 10. Histogram of 100 hand gestures, to show the Gaussian property

4. CONCLUSIONS AND FURTHER WORK

We have proposed an interactive multimedia approach for interpreting user commands in a real-time environment. The approach utilizes hand gestures and a simple pointing arrow to generate visual commands. The focus was placed on fast recognition which is essential in distributed multimedia applications. The approach includes three main aspects: (1) image segmentation and object extraction, (2) hand gesture recognition and classification, and (3) arrow's direction detection.

We have also introduced a novel paradigm to select efficient hand gestures using *cluster-property* and *cluster-features* matrices. The former includes recognition rates for different gestures using singular properties and the latter deals with multiple-valued features. The recognition rates are obtained utilizing two simplified decision functions based on Bayesian analysis of statistical models. Moreover, we have examined several features to discriminate hand gestures in a simple, fast, and robust way. Direction

detection was conducted exploiting orientation of the major axis of the arrow's region and an assumption of different thickness at two ends. K-mean clustering and connected component analysis have been used for binarization and object extraction, respectively.

Experimental results are discussed in three different areas. The results explicitly show discrimination rank of individual hand gestures, which can be used to reasonably select appropriate gestures in other multimedia applications. Moreover, a set of combinatory features has been examined and a small feature vector is obtained. Results of object detection are promising while gesture recognition is fast enough and suitable for interactive applications. Experimental results in both off-line and real-time stages confirm the efficacy of the proposed method.

The proposed approach for gesture recognition/classification can be applied on other gestures including limb, head, and whole body gestures. Shape features extracted from the gesture image can be easily compared using the proposed approach. Moreover, we intend to employ the proposed scheme in immersive distributed environments, where several users using a distributed system communicate through their hand or body gestures plus a pointing arrow. For further improvements, objective criteria for user satisfaction can be defined and a non-homogenous background would be examined.

REFERENCES

1. Lee, S. W. (2006). Automatic gesture recognition for intelligent human-robot interaction. *Proc. IEEE 7th Intr. Conf. Automatic Face and Gesture Recognition*, pp. 645–650.
2. Chalechale, A. & Naghdy, G. (2007). Visual-based human-machine interface using hand gestures. *Proc. IEEE 9th Intr. Sypo. signal processing and its applications*, UAE.
3. Zhao, L. & Badler, N. I. (2005). Acquiring and validating motion qualities from live limb gestures. *Graphical Models*, Vol. 67, No. 1, pp. 1–16.
4. Barrientos, F. A. & Canny, J. F. (2002). Cursive: Controlling expressive avatar gesture using pen gesture. *Proc. Collaborative Virtual Environments*, Bonn, Germany, pp. 113–119.
5. iGesture web site, (2007). http://www.igesture.org/, the URL has been visited on 31/10/2007.
6. Kang, H., Lee, C. W. & Jung, K. (2004). Recognition-based gesture spotting in video games. *Pattern Recognition Letters*, Vol. 25, No. 15, pp. 1701–1714.
7. ISO/IEC JTC 1/SC 29/WG 11 N 2502, (1998). Information technology-generic coding of audio-visual objects-part 2: visual," Tech. Rep., ISO/IEC, Atlantic City, Oct.
8. Jian, Z., Kaynak, M. N., Cheok, A. D. & Chung, K. C. (2001). Real-time lip tracking for virtual lip implementation in virtual environments and computer games, *Proc. IEEE Int. Conf. Fuzzy Systems*, Vol. 3, pp. 1359–1362.
9. Chang, C. C. & Tsai, W. H. (2001). Vision-based tracking and interpretation of human leg movement for virtual reality applications. *IEEE Trans. Circuits and Systems for Video Technology*, Vol. 11, No. 1, pp. 9–24.
10. Birk, H., Moeslund, T. B. & Madsen, C. B. (1997). Real-time recognition of hand gestures using principal component analysis. *Proc. 10th Scandinavian Conf. on Image Analysis (SCIA'97)*.
11. E. J. Holden home page at the University of Western Australia, http://www.csse.uwa.edu.au/ eunjung/, the URL has been visited on 21/12/2004.
12. Morguet, P. & Lang, M. (1999). Comparison of approaches to continuous hand gesture recognition for a visual dialog system. *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, Vol. 6, pp. 3549–3552.

13. Thornton, J., Orengo, C. & Jones, D. (2003). *Bioinformatics: genes, proteins and computers*, BIOS.
14. Jain, A. J. & Vailaya, A. (1998). Shape-based retrieval: a case study with trademark image databases. *Patt. Recog.*, Vol. 31, No. 9, pp. 1369–1390.
15. Yoon, S. J., Park, D. K., Park, S. & Won, C. S. (2001). Image retrieval using a novel relevance feedback for edge histogram descriptor of MPEG-7. *Proc. IEEE Int. Conf. Consumer Electronics*, Piscataway, NJ, USA, pp. 354–355.
16. Mohamad, D., Sulong, G. & Ipson, S. S. (1995). Trademark matching using invariant moments. *Proc. second Asian Conf. Comput. Vision, [ACVV'95]*., Singapore, Vol. 1, pp. 439–444.
17. Birk, H. & Moeslund, T. B. (1996). Recognizing gestures from the hand alphabet using principal component analysis, M.S. thesis, Laboratory of Image Analysis, Aalborg University.
18. Gonzalez, R. C. & Woods, R. E. (1992). *Digital image processing*, Addison-Wesley.
19. Haralick, R. M. & Shapiro, L. G. (1992). *Computer and robot vision*, Addison-Wesley.