Real Time Hand Gesture Recognition Including Hand Segmentation and Tracking

Thomas Coogan[1], George Awad[2], Junwei Han[3], Alistair Sutherland[4]

Dublin City University, Dublin 9, Ireland
{[1]tcoogan, [2]gawad, [3]jhan, [4]alistair} @computing.dcu.ie

Abstract. In this paper we present a system that performs automatic gesture recognition. The system consists of two main components: (i) A unified technique for segmentation and tracking of face and hands using a skin detection algorithm along with handling occlusion between skin objects to keep track of the status of the occluded parts. This is realized by combining 3 useful features, namely, color, motion and position. (ii) A static and dynamic gesture recognition system. Static gesture recognition is achieved using a robust hand shape classification, based on PCA subspaces, that is invariant to scale along with small translation and rotation transformations. Combining hand shape classification with position information and using DHMMs allows us to accomplish dynamic gesture recognition.

1 Introduction

The primary goal of any automated gesture recognition system is to create an interface that is natural for humans to operate or communicate with a computerized device. Furthermore we aim to develop our system without using data gloves or colored gloves. Such a system could be used in, virtual reality, robot manipulation or gaming. In fact gesture recognition could be used to improve the intuitiveness of any Human-Computer Interaction (HCI). We routinely use hand gestures when communicating, describing and directing during our everyday activities. Incorporating gestures with HCI could be an extremely beneficial development.

In recent years various approaches to gesture recognition have been proposed, Gupta et al [1] presented a method of performing gesture recognition by tracking the sequence of contours of the hand using localized contour sequences. Chen et al [2] developed a dynamic gesture recognition system using Hidden Markov Models (HMMs). Patwardhan et al [3] recently introduced a system based on a predictive eigentracker to track the changing appearance of a moving hand. Kadir et al [4] describe a technique to recognize sign language gestures using a set of discrete features to describe position of the hands relative to each other, position of the hands relative to other body locations, movement of the hand, shape of the hand. While some of these approaches display impressive results, many exploit controlled environments and a compromise between vocabulary size and recognition rate.

To achieve accurate gesture recognition over a large vocabulary we need to extract information about the hand shape. Accomplishing this entails detecting the hands, segmenting them, differentiating them and classifying them. This requires using skin detection techniques and handling occlusion between skin objects to keep track of the status of the occluded parts. We present a unified system for segmentation and tracking of the face and hands in a gesture recognition using a single camera. Unlike much related work that uses color gloves [5], we detect skin by combining 3 useful features: color, motion and position. These features together, represent the skin color pixels that are more likely to be foreground pixels and are within a predicted position range. Also, unlike other work that avoid occlusions entirely by choice of camera angle, sign vocabulary, or by performing unnatural signs [6,7], we handle occlusion between any of the skin objects using a Kalman filter based algorithm.

Once the hand is segmented classification is required. In hand shape recognition, transformation invariance is key for successful recognition. We propose a system that is invariant to small scale, translation and shape variations. This is achieved by using a-priori knowledge to create a transformation subspace for each hand shape. Transformation subspaces are created by performing Principal Component Analysis (PCA) on images produced using computer animation. We introduce our method that enables us to train this appearance based method using computer animation images. Also presented is the incorporation of this hand shape classifier into a dynamic gesture recognition system. Position information is combined with hand shape information to construct a feature vector that is passed to a DHMM for dynamic gesture recognition.

The remainder of this paper is organized as follows: The method used to segment the face and hands is reported in section 2. The gesture recognition technique including hand shape recognition is described in section 3. In section 4 we detail some experiments and finally we offer some conclusions in section 5.

2 Segmentation and Tracking

In gesture recognition we need to segment and track three objects of interest: the face and the two hands. The skin segmentation module is responsible for segmentation of skin objects, similarly the object tracking module is responsible for matching the resulting skin blobs of the segmentation component to the previous frame blobs while keeping track of the occlusion status of the three objects. In the next sections we will explain the details of these two components.

2.1 Skin Segmentation

In order to robustly detect the skin objects, we combine three useful features: color, motion and position. Color cue is useful because the skin has a distinct color that helps to differentiate it from other colors. The motion cue is useful in discriminating foreground from background pixels. Finally, the predicted position of objects using Kalman filter helps to reduce the search space.

2.1.1 Color Information

In order to collect candidate skin pixels, we use two simple classifiers. First, a general skin model (color range) is applied on small search windows around the predicted positions of skin objects. As the fixed color range can miss some skin pixels, we propose another color distance metric dist(C_{skin}, X_{ij}) to take advantage of the prior knowledge of the last segmented object. This metric is the Euclidean distance between the average skin color C_{skin} in the previously segmented skin object and the current pixel X_{ij} in the search window at positions i and j. Finally, we normalize the values of the prior knowledge color metric P_{col}

2.1.2 Motion Information

Finding the movement information takes two steps. Firstly, motion detection, then next step, finding candidate foreground pixels. The first step examines the local gray-level changes between successive frames by frame differencing:

$$D_i(x, y) = \left| W_i(x, y) - W_{i-1}(x, y) \right| \quad (1)$$

Where W_i is the ith search window and D_i is the absolute difference image. We then normalize D_i to convert it to probability values. The second step assigns a probability value $P_m(x, y)$ for each pixel in the search window to represent how likely this pixel belongs to a skin object. This is done by looking backward to the last segmented skin object binary image in the previous frame search window OBJ_{i-1} and applying the following model on the pixels in D_i:

$$P_m(x, y) = \begin{cases} 1 - D_i(x, y) & \text{if } OBJ_{i-1}(x, y) \equiv 1 \\ D_i(x, y) & \text{otherwise} \end{cases} \quad (2)$$

In this way, small values (stationary pixels) in D_i that were previously segmented as object pixels will be assigned high probability values as they represent skin pixels that were not moved, and new background pixels with high D_i will be assigned small probability values. So simply, this model gives high probability values to candidate skin pixels and low values to candidate background values.

2.1.3 Position Information

To capture the dynamics of the skin objects, we assume that the movement is sufficiently small between successive frames. Accordingly, a Kalman filter model can be used to describe the x and y coordinate of the center of the skin objects with a state vector S_k that indicates the position and velocity. The model can be described as:

$$S_{k+1} = A_k S_k + G_k \quad (3)$$

$$Z_k = S_k + V_k \quad (4)$$

Where A_k is a constant velocity model, G_k , V_k represents the state and measurement noise respectively and Z_k is the observation. This model is used to keep track of the position of the skin objects and predict the new position in the next frame.

Given that the search window surrounds the predicted center, we translate a binary mask of the object from the previous frame to be centered on the new predicted center. Then the distance transform is computed between all pixels in the search window and pixels of the mask. The inverse of this distance values assigns high values to pixels that are belonging to or near the mask and low values to far pixels. The distance values are then converted to probabilities P_{pos} by normalization

2.1.4 Information Combination.

After collecting the color, motion and position features, we combine them logically using an abstract fusion formula to obtain a binary decision image $F_i(x,y)$:

$$F_i(x,y) = \begin{cases} 1 & \text{if } (P_{col}(x,y) > \tau) \; OR \quad ((P_g(x,y) == 1) \\ & \quad AND \quad (P_m(x,y) > \upsilon) \; AND \quad (P_{pos}(x,y) > \sigma)) \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

Where P_{col}, P_m, and P_{pos} is the decision probability values of the color metric, motion, and position respectively. P_g is the output of the general skin model, and τ, υ, and σ are thresholds where σ is determined adaptively by the following formula:

$$\sigma = \frac{size\ ((P_m(x,y) > \upsilon)\ AND\quad (P_{pos}(x,y) \equiv 1))}{size\ (P_m(x,y) > \upsilon)} \tag{6}$$

The threshold σ determines the margin that we are searching into around the predicted object position. In Eq. (6) this is formulated by finding the overlapping between the predicted object position and the foreground pixels above certain threshold value. The other thresholds values are determined empirically.

2.2 Tracking and Occlusion Detection

2.2.1 Occlusion Detection.

In this paper, we propose a Kalman filter based algorithm to detect occlusion between any of the face and the two hands. In general, the algorithm uses the Kalman filter model to track the four corner points of the bounding box around the face and two hands. This model can predict in the next frame the positions of these four corner points. Accordingly, we check to see if there is any overlap between any of the bounding boxes in the next frame. If there is an overlap, we raise an occlusion alarm corresponding to the two bounding boxes that will overlap.

If in the next frame, the number of detected skin objects is less than the current frame objects *and* an occlusion alarm was raised previously, we conclude that occlusion happened. On the other hand, if the number of detected skin objects decreases and no occlusion alarms were raised, then one or more skin objects have left the frame.

2.2.2 Tracking.

The tracking process starts by first constructing search windows around each of the predicted positions of the tracked objects. When two or more objects are occluded,

they are treated as one object and one search window is constructed around their position. Next, connected regions are labeled after removing noisy small regions. Using the number of detected skin objects and the occlusion alarms as discussed in section 2.2.1, we maintain a high-level understanding of the status of the current frame with respect to the occlusion status. For example, if we detected 1 object and occlusion alarm between the face and left hand is raised, then we conclude that the face and left hand are occluded and the right hand is hiding. This technique can be extended to handle all 7 situations of occlusion status: separate face and two hands, face and 2 hands occluded, separate hand with face and hand occluded, face and hiding hands, face and hands all occluded.

The final step in the tracking part is the blob matching where the previous frame blobs are matched against the new frame blobs using the knowledge of the high-level occlusions status we described above. The matching is done using the distance between the previous objects centers and the new objects centers.

3 Gesture Recognition

3.1 Hand Shape Recognition

To date many approaches have been proposed for hand shape recognition. These include: (i) Shamaie [8] introduced a PCA based approach. (ii) Just et al [9] introduced a hand shape system based on the Modified Census Transform MCT. (iii) A method to classify hand postures against complex cluttered background was proposed by Triesch & von der Malsburg [10] using elastic graph matching. (iv) Yuan et al [11] developed their system by determining a new Active Shape Model (ASM) kernel based on the shape contours. (v) Chen et al [2] present a method of classifying the static hand poses by using the Fourier Descriptor to characterize the spatial features of the hands boundary.

Many of these approaches for hand shape recognition display sub-optimal results due to the highly deformable nature of the hand. Once again a compromise is determined between small vocabulary and accurate recognition. Due to the complex temperament of the hand, any hand shape classifier needs to be able to cope with small rotations and translation transformations.

We propose a transformation subspace technique to combat these issues. Our proposed method creates the invariant subspaces from a sampled subset of all possible transformation images. These images are produced systematically using the commercially available POSER computer animation (CA) software and includes 3D hand transformation. Performing PCA on these images will generate a subspace that accurately represents the complex transformation hyper-plane.

Using this method of a-priori knowledge to construct the subspaces means we can eliminate the process of automatic subspace segmentation as proposed by [12]. It also allows us to dismiss the need for managing outliers or missing data in our subspaces [12]. This means we can create more accurate transformations subspaces to what was previously possible using the simple PCA method.

PCA provides M orthogonal eigenvectors {u1, … ,uM} of the covariance matrix, that correspond to the first M largest eigenvalues, in order to maintain a minimum

energy of the dataset. In order to classify test images, a distance metric needs to be introduced. We project the test image into the subspace and find the perpendicular distance of the projected point to the eigenvectors representing the subspace.

We now have the backbone for hand shape recognition system. ISL contains 28 static fingerspelling gestures. The system is constructed as follows:

Training - Generating a transformation subspace for each hand shape.

Testing – Project the test image into each of the subspaces to find the subspace with the nearest perpendicular distance. This subspace will be representative of one particular hand shape.

3.2 Hand Image Pre-Processing

If this technique is to be worthwhile we need to be able to accurately classify images of real hands when we train with CA images. In order to do these we need to neutralize the differences between CA hand images and images of a human hand. Such a system would inherently be multi-user as it would be trained and tested by different users. We have developed a detailed pre-processing step to counteract issues such as: skin color, illumination variation, hand size and distance from the camera.

3.2.1 Hand Scaling and Alignment.

Once the hand has been identified and segmented it should be scaled and aligned to ensure the system can deal with users with different sized hands and users at differing distances from the camera. We exploit a simple but effective custom of scaling the hand objects so that they occupy a predetermined area in a 32*32 resized image. Alignment is accomplished by repositioning the hand object so that the center of the bounding box lies in the center of the image.

3.2.2 Skin Color and Illumination Variation

Removing skin color and color variance due to illumination is essential in an appearance based multi user hand shape recognition system. First the hand image is converted to grayscale, this reduces the space at which color can be represented. In order to color normalize each hand image in grayscale space we have incorporated a color histogram equalization approach into our system. Color histograms are graphs that depict the color distribution of pixels in an image. Histogram Equalization is the process of redistributing the color values in the image so that the image histogram takes a predetermined form.

We know from the hand-scaling step that all hand objects are resized to occupy the same area within an image. With this in mind, a common histogram can be defined that can represent all hand images. It contains a large spike that represents the background of the image; this is located at the beginning of the color scale because the background pixels are set to 0. Then a gaussian-shaped pulse exists towards the end of the color scale represents object pixels. This positioning is important to maximize the contrast of the normalized hand image.

3.2.3 Image Filtering

We have found that it is useful to apply a simple gaussian filter to an image before classification. This filter can help smooth out noise in an image. The hand image is convolved with a 9*9 gaussian kernel with a small standard deviation to ensure the filtering doesn't blur important information in the image.

3.3 Dynamic Gesture Recognition

Dynamic gesture recognition requires both temporal and spatial recognition of the hand movement and hand shape. We have devised a simple system using Hidden Markov Models (HMMs) to recognize dynamic gestures. A HMM is a tool for representing probability distributions over a sequence of observations. For our dynamic gesture recognition system the sequence of observations are feature vectors. These feature vectors consist of two elements, both of which are positive integers. The first denotes the group to which the static hand shape has been classified. It should be noted that 40 static hand shape groups are currently used to distinguish the gestures in this system. The second defines the position of the hand in the image and will be in the range 1-9. The position of the hand is classified by determining the section of the image that the center of the hand lies in. This center is calculated by finding the center of the hands bounding box.

The image is divided into 9 sections. These sections are created by diving the image vertically by drawing two lines either side of the head. The 1st of the horizontal lines, H1, is located directly under the head. The 2nd, H2, is placed M pixels below H1, where M is the length of the head object. Using this technique position can be calculated invariantly to the distance of the user from the camera. A gesture is then represented as a sequence of tuples, containing both shape and position information.

A DHMM is trained for each possible gesture using many different examples. A gesture is classified online, by manually identifying its start and stop point, then finding the DHMM with the highest probability for the feature vector of the test sequence.

4 Experiments

4.1 Static Hand Shape Classification

In order to classify real hand images we create a subspace for each hand shape as in section 3.1. A subspace for each hand shape is now created by performing PCA on 3969 images as depicted in equation 7.

$$\begin{aligned} &1\ \text{Origin image}^{[1]} \times 49\ \text{translations}^{[2]} \times 9\ \text{rotations in yaw direction}^{[3]} \\ &\times 3\ \text{rotations in pitch direction}^{[4]} \times 3\ \text{rotations in roll direction}^{[5]} \\ &= 3969\ \text{images}. \end{aligned} \tag{7}$$

[1] Origin hand image that can be defined as being the perfect orientation of the hand shape.
[2] Origin hand shapes translated in all directions using combinations of 2, 4 and 6 pixels.
[3] 9 rotations are used in the yaw direction as this is the direction that contains most significant deviation. These rotations are 3 degrees apart covering a total pitch of 24 degrees.
[4] 3 rotations in the roll direction, each at 10 degrees covering a total pitch of 20 degrees.
[5] 3 rotations in the roll direction, each at 10 degrees covering a total pitch of 20 degrees.

All test and training images are pre-processed using the techniques described in section 3.2. We developed a test set in order to test the amount of energy we need to retain in each of these subspaces. This test set contained 560 images, 20 occurrences of each of the 28 hand shapes been used. All these images were acquired from one trained user of the system over 4 separate sittings on 2 different days. In theses images the assumption has been made that the user is wearing long sleeves covering the arms. The first objective of our experiments was to identify the energy retention value that gives superior recognition. We have found that when 80% of the energy is retained the lowest error rate is achieved. One explanation for this is once we go over 80% the subspaces attempt to retain information that is local to that of the individual user, i.e. local characteristics of the computer animation images. It is important to find this balance between retaining as much information as possible without introducing noise in our subspaces. 80% energy retention entails keeping 12-16 of the most significant eigenvectors, depending on the hand shape. Having a low number of eigenvectors is also important to maintain efficiency.

Table 1. Confusion matrix for the 28 static handshapes

	A	B	C	D	E	F	G	H	I	K	L	M	N	O	P	Q	R	S	T	U	V	W	Y	1	2	3	4	5
A	18																	1	1									
B		20																										
C			20																									
D				16				2											2									
E					20																							
F						19																		1				
G							18							2														
H								18						1	1													
I									20																			
K										18																2		
L											20																	
M												20																
N													20															
O														19				1										
P															19					1								
Q										1						17					2							
R																	17			3								
S	1				1													18										
T					1												1		18									
U																				20								
V																					20							
W																					1	19						
Y																							20					
1																								20				
2																									20			
3										1																19		
4																					1						19	
5																										3		17

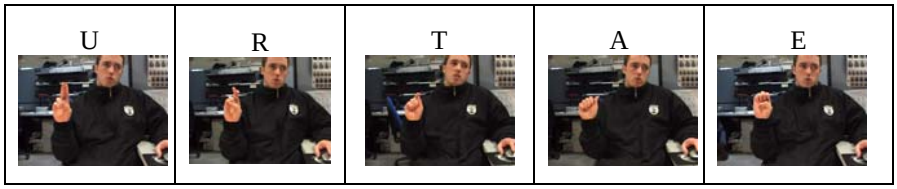

Fig. 1. Sample Static gestures

Preserving 80% energy in the subspace we achieve 94.5% recognition accuracy for our test set. The performance accuracy of each individual static gesture can be observed in **Table 1**. This table exhibits the confusion matrix for the static gesture

recognition vocabulary. Most confusion is caused where gestures are very similar. The two gestures that give highest confusion are U and R. These gestures only differ slightly as can be seen in **Fig 1**. When performing U, the index and middle finger lay parallel, while performing R the index and middle finger are crossed. These differences become particularly minute once the images are scaled to 32*32.

4.2 Dynamic Gesture Recognition

In order to test the accuracy of our dynamic gesture recognition system we have generated a vocabulary of 17 dynamic single-handed gestures. This lexicon was created to ensure gestures exist that differ only in either hand shape or hand position. 20 samples of each isolated gesture were recorded employing 2 different users, of different racial origins, over 4 different days. The videos are captured in an office environment with additional lighting to the front of the user. In our experiments the samples are divided into test and training sets by random sampling. The proportion of test and training data was then varied over the recognition experiments. A DHMM was trained for each gesture using the selected training data. We then tested the recognition accuracy using the remaining unseen data. Recognition accuracy was calculated by computing the average performance for different sampling of the training data. **Table 2** displays this performance for each of the different number of data samples used in training. As expected the performance increases as the number of training samples increases. It is also interesting to note that reasonably high classification rates can be achieved using only one training sample for the DHMM.

This classification has been achieved on a standard PC using the matlab interpreter with non-optimized code in real time at 10 frames per second.

Table 2. illustrates the performance for each of the different number of data samples used for training

No. Training samples	1	2	3	4	5	6	7	8	9	10
Average Performance	83.0	88.9	91.9	94.2	94.6	95.3	95.9	97.1	98.5	98.6

5 Conclusions

In this paper a detailed framework is presented for accurate real time gesture recognition. A unified approach for segmenting and tracking skin color objects has been described. Tracking helps to reduce the search space for segmentation while accurate segmentation helps to accurately enhance the tracking performance. Also accurate segmentation assists improving hand shape recognition using the subspace classifier described. A novel approach of training a subspace classifier using images generated from computer animation is also illustrated. Using image-processing techniques we

have shown that accurate recognition is possible for human hands. Combining this hand shape information with the position information a gesture recognition system was generated. Successful classification was achieved for isolated gestures even with limited training. To improve performance over a larger lexicon we intend to introduce a more detailed position gauge, increase the bank of allowable hand shapes along with introducing new features such as hand motion.

Acknowledgements

This work is partly funded by the Irish Research Council for Science Engineering and Technology, (IRCSET).

References

1. Gupta, L., Ma S.: Gesture-Based Interaction and Communication: Automated Classification of Gesture Contours", IEEE Transactions on Systems, Man and Cybernetics – Part C: Applications and reviews, Vol 31, No 1, 2001
2. Chen, F.-S., Fu, C.-m., Huang, C.-L.: Hand Gesture Recognition Using a Real-time Tracking Method and Hidden Markov Models, Image and Vision Computing, Vol 21, pages 745-758, 2003.
3. Patwardhan, K. S. Dutta Roy, S.: Hand gesture modeling and recognition involving changing shapes and trajectories, using a Predictive EigenTracker, Pattern Recognition, (Article in Press), 2006.
4. Kadir,T., Bowden,R., Ong,E. J., Zisserman, A.:Minimal Training, Large Lexicon, Unconstrained Sign Language Recognition, In Proc BMVC'04, Vol 2, pp849-858, 2004.
5. Shamaie, A., Sutherland, A.: Hand Tracking in Bimanual Movements, Image and Vision Computing, 23, pp. 1131–1149, 2005.
6. Huang, C.-L., Jeng, S.-H.: A Model-Based Hand Gesture Recognition System, Machine Vision and Application, 12(5), pp. 243-258, 2001.
7. Terrillon, J.-C, Piplr, A., Niwa, Y., Yamamoto, K.: Robust Face Detection and Japanese Sign Language Hand Posture Recognition for Human-Computer Interaction in an Intelligent Room, In Proc. Int'l Conf. Vision Interface, pp. 369-376, 2002.
8. Shamaie, A.: Hand Tracking and Bimanual Movement Understanding, PHD Thesis, Dublin City University 2003.
9. Just, A., Rodriguez, Y., Marcel, S.: Hand Posture Classification and Recognition using the Modified Census Transform, in Proc. IEEE International Conference on Face and Gesture, pp 351-356, 2006.
10. Triesch, J., Von der Malsburg, C.: Classification of hand postures against complex backgrounds using elastic graph matching, Image and Vision Computing Vol 20, pp 937-943, 2002.
11. Yuan, Y., Barner, K.: An Active Shape Model Based Tactile Hand Shape Recognition with Support Vector Machine, in Proc 40th Annual Conf Information Sciences and systems, 2006.
12. Bishoff, H., Leonardis, A., Maver, J.: Multiple Eigenspaces, Pattern Recognition, Vol 35, pp 2613-2627, 2002.