A Real-Time Hand Gesture Interface for Medical Visualization Applications

Juan Wachs[1], Helman Stern[1], Yael Edan[1], Michael Gillam[2], Craig Feied[2], Mark Smith[2], and Jon Handler[2]

[1]Department of Industrial Engineering and Management, Ben-Gurion University of the Negev, Be'er-Sheva, Israel, 84105,
{helman, yael, juan}@bgu.ac.il.
[2]Institute for Medical Informatics, Washington Hospital Center, 110 Irving Street, NW, Washington, DC, 20010,{feied, smith, handler, gillam}@medstar.net

Abstract. In this paper, we consider a vision-based system that can interpret a user's gestures in real time to manipulate objects within a medical data visualization environment. Dynamic navigation gestures are translated to commands based on their relative positions on the screen. Static gesture poses are identified to execute non-directional commands. This is accomplished by using Haar-like features to represent the shape of the hand. These features are then input to a Fuzzy C-Means Clustering algorithm for pose classification. A probabilistic neighborhood search algorithm is employed to automatically select a small number of Haar features, and to tune the fuzzy c-means classification algorithm. The gesture recognition system was implemented in a sterile medical data-browser environment. Test results on four interface tasks showed that the use of a few Haar features with the supervised FCM yielded successful performance rates of 95 to 100%. In addition a small exploratory test of the Adaboost Haar system was made to detect a single hand gesture, and assess its suitability for hand gesture recognition.

Keywords: haar features, fuzzy c-means, hand gesture recognition, neighborhood search, computerized databases.

1 Introduction

Computer information technology is increasingly penetrating into the hospital domain. It is important that such technology be used in a safe manner in order to avoid serious mistakes leading to possible fatal incidents. Unfortunately, It has been found that a common method of spreading infection includes computer keyboards and mice in intensive care units (ICUs) used by doctors and nurses [7]. Many of these deficiencies may be overcome by introducing a more natural human computer interaction (HCI), especially speech and gesture. Face gestures are used in FAce MOUSe [6] whereby a surgeon can control the motion of the laparoscope. Gaze, is used as one of the diagnostic imaging techniques for selecting CT images by eye movements [12]. Here a vision-based gesture capture system to manipulate windows and objects within a graphical user interface (GUI) is proffered. Research using a hand gesture computer vision system appeared in [4]. In [13] the tracking position of fingers is used to collect quantitative data about the breast palpation process. In our work we consider hand motion and posture simultaneously. Our system is user independent without the need of a large multi-user training set. We use a fuzzy c-mean discriminator along with Haar type features. In order to obtain a more optimal system design we employ a neighborhood search method for efficient feature selection and classifier parameter tuning. The real time operation of the gesture interface was tested in a hospital environment. In this domain the non-contact aspect of the gesture interface avoids the problem of possible transfer of contagious diseases through traditional keyboard/mice user interfaces.

A system overview is presented in Section 2. Section 3 describes the segmentation of the hand from the background. Section 4 deals with feature extraction and pose recognition. The results of performance tests for the FCM hand gesture recognition system appear in Section 5. Section 6 concludes the paper.

2 System Overview

A web-camera placed above the screen (Figure. 1(a)) captures a sequence of images like those shown in Figure 1(b). The hand is segmented using color cues, a B/W threshold, and various morphological image processing operations. The location of the hand is represented by the 2D coordinates of its centroid, and mapped into one of eight possible navigation directions of the screen (see Figure 1(c)) to position the cursor of a virtual mouse. The motion of the hand is interpreted by a tracking module. At certain points in the interaction it becomes necessary to classify the pose of the hand. Then the image is cropped tightly around the blob of the hand and a more accurate segmentation is performed. The postures are recognized by extracting symbolic features (of the Haar type) from the sequence of images. The sequence of features is interpreted by a supervised FCM that has been trained to discriminate various hand poses. The classification is used

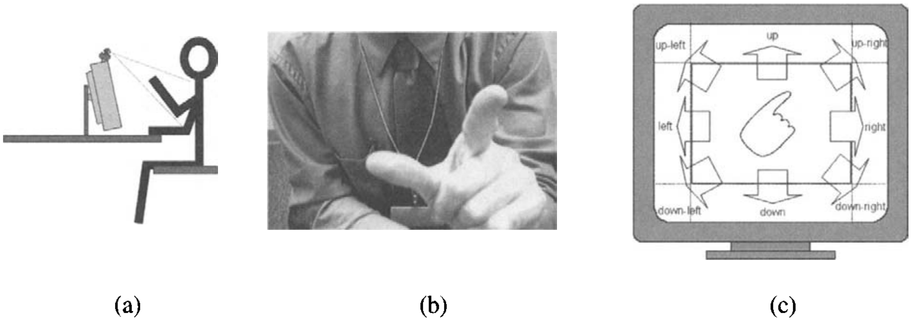


Fig. 1. (a)(b) Gesture Capture. (c) Screen navigation map

to bring up X-rays images, select a patient record from the database or move objects and windows in the screen. A two-layer architecture is used. The lower level provides tracking and recognition functions, while the higher level manages the user interface.

3 Segmentation

In order to track and recognize gestures, the CAMSHIFT [2] algorithm is used together with an FCM algorithm [10]. For CAMSHIFT, a hand color probability distribution image is created using a 2D hue-saturation color histogram [3]. This histogram is used as a look-up-table to convert a camera image into a corresponding skin color probability image through a process known as back propagation. A backprojected image assigns to each pixel a number between 0 and 1 as the likelihood of it being classified as a hand pixel. Thresholding to black and white, followed by morphological operations, is used to obtain a single component for further processing to classify the gestures.

The initial 2D histogram is generated in real-time by the user in the 'calibration' stage of the system. The interface preview window shows an outline of the palm of the hand gesture drawn on the screen. The user places his/her hand within the template while the color model histogram is built (Fig. 2), after which the tracking module (Camshift) is triggered to follow the hand. The calibration process is initiated by the detection of motion of the hand within the region of the template. In order to avoid false motion clues originated by non hand motion a background maintenance operation is maintained. A first image of the background is stored immediately after the application is launched, and then background differencing is used to isolate the moving object (hand) from the background.

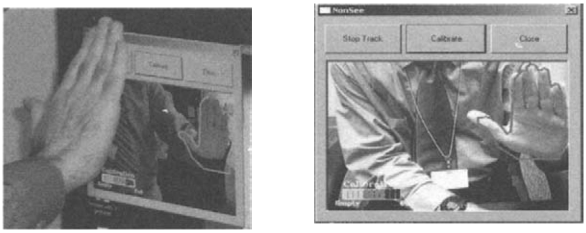


Fig. 2. User hand skin color calibration

Since background pixels have small variations due to changes in illumination over an extended period of time, the background image must be dynamically changed. Background variations are identified by a threshold applied to the absolute difference between every two consecutive frames. If the difference is under some threshold t_1, then the current images contain only a background, otherwise, an upper threshold level t_2 is checked to test whether the present object is a hand. In case that the current image is a background, the background stored image is updated using a running smoothed average.

$$Bcc_k(i,j) = (1-\alpha) * Bcc_{k-1}(i,j) + \alpha * f(i,j) \tag{1}$$

In (1) Bcc_k is the updated stored background image at frame k, Bcc_{k-1} is the stored background image at frame k-1, α is the smoothing coefficient (regulating update speed), f(i,j) is the current background image at frame k. Small changes in illumination will only update the background while huge changes in intensity will trigger the tracking module. It is assumed that the hand is the only skin colored object moving on the area of the template. The process of calibration requires only a few seconds, and is necessary as every user has a slightly different skin color distribution, and changes in artificial/daylight illumination affect the color model. A low threshold and open and close morphology operations followed by largest component selection are applied to obtain a single connected blob (Fig. 3).

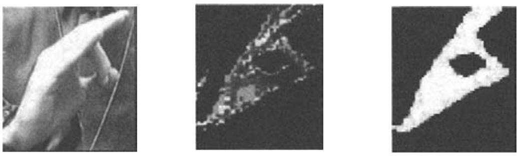


Fig. 3. Image processing of the pose

4 Feature Extraction and Pose Recognition

4.1 Hand Tracking and Pose Recognition

Hand gestures are classified using a finite state machine (Fig. 4).When a doctor wishes to move the cursor over the screen, the hand moves out of the 'neutral area' to any of 8 directional regions (Fig. 5). This interaction allows the doctor to rest the hand in the 'neutral area'.

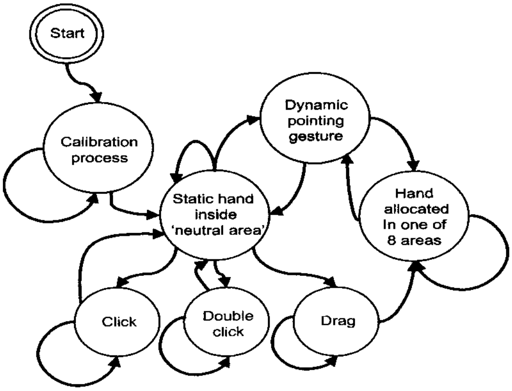


Fig. 4. State machine for the gesture-based medical browser

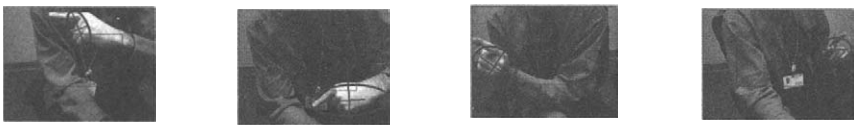
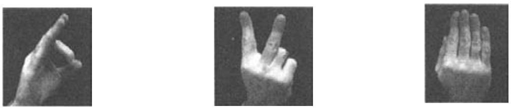


Fig. 5. Four quadrants mapped to cursor movement

To facilitate positioning, hand motion is mapped to cursor movement. Small, slow hand (large fast) motion cause small (large) pointer position changes. In this manner the user can precisely control pointer alignment.When a doctor decides to perform a click, double-click, or drag with the virtual mouse, he/she places the hand in the 'neutral area' momentarily to trigger the transition. These three poses are shown in Figure 6.



(a). Click (b). Double-click (c). Drag

Fig. 6. The gesture vocabulary

4.2 Haar Features

Basically, the features of this detector are weighted differences of integrals over rectangular sub regions. Figure 7(a)-(d) visualizes the set of available feature types, where black and white rectangles correspond to positive and negative weights, respectively. The feature types consist of four different edge-line features. The learning algorithm automatically selects the most discriminate features considering all possible feature types, sizes and locations. These features are reminiscent of Haar wavelets, which can be computed in constant time at any scale, and use the original image without preprocessing. Each rectangular feature is computed by summing up pixel values within smaller rectangles, see Eq. (2).

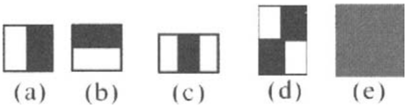


Fig. 7. Extended integral rectangle feature set

$$f_i = \sum_{i \in I = \{1,\ldots,N\}} \omega_i * \mathrm{Re}\, cSum(r_i) \qquad (2)$$

In (2) $\omega_i \in \Re$ are weights, r_i is the *ith* rectangle, and N is the number of rectangles. The weights have opposite signs (indicated as black and white in Figure. 7), and are used to compensate between differences in area. Efficient computation is achieved by using summed area tables. We have added a block average feature (see Fig. 7(e)) to f_1 , f_2 , f_3 , and f_4 (see Fig. 7(a)-(d)) selected from the original feature set of Viola-Jones. A rectangle, r, in the image can be defined by the (x,y) position of its upper left corner, and by its width w and height h. We constrain the total set of rectangles in an image, by using the relation: x=w*n, and y=h*m. where n and m are integer numbers. Hence, the total number of rectangles is less than 13,334 in lieu of (>750,000) for a 100x100 resolution classifier using the full Viola-Jones set 9].

4.3 Pose Recognition

In our system we reduce the Haar rectangular positions severely to a set of 'selected' rectangles v. These rectangles are limited to lie within a bounding box of the hand tracking window, and are obtained by dividing the window into m rows and n columns. For each cell a binary variable is used to decide whether it is selected or not. A more elaborate strategy enables one to define the type of feature for selected rectangles. Therefore, a set of rectangles in a window is defined by {n,m,t}, where n, m are columns and rows; and t={t_1,...,t_i,...,t_v} represent the type of feature of rectangle i (indexed row wise from left to right). The feature type t can take integer values from 0 to 5, where 0 indicates that the rectangle is not selected, and 1,2,3,4,5 represent features of type f_1 , f_2 , f_3 ,f_4 and f_5, respectively. The hypothesis expressed in Viola and Jones is that a very small number of these features can be combined to form an effective classifier. As opposed to Viola and Jones method, our learning algorithm is not designed to select a single rectangle feature which best separates the positive and negative for each stage of a cascade of classifiers. Instead, we evaluate a set of rectangle features simultaneously, which accelerates the process of feature selection. The Haar features selected are input into the hand gesture FCM recognition system. Note, that feature sizes are automatically adjusted to fit into a dynamically changing bounding box created by the tracking system.

4.4 Optimal Feature and Recognition Parameter Selection

Feature selection and finding the parameters of the FCM algorithm for classifying hand gestures is done by a probabilistic neighborhood search (PNS) method [8]. The PNS selects samples in a small neighborhood around the current solution based on a special mixture point distribution:

$$\mathrm{PS}(x \mid h) = \begin{cases} h, & x = 0 \\ h((1-h)^{|x|})/2, & x = \pm 1, \pm 2,,,,\pm(S-1) \\ ((1-h)^{|x|})/2, & x = \pm S \end{cases} \tag{3}$$

Where, S = maximum number of step increments, h = probability of no change, x_j = a random variable representing the signed (positive or negative coordinate direction) number of step size changes for parameter p_j, and $P_S(x|h) = P_r(x = s)$ the probability of step size s, given h.

Figure 8 shows an example of the convergence behavior of the PNS algorithm for 5 randomly generated starting solutions. Figure 9 shows the optimal set of features selected by this run. The features f_4 and f_5 capture characteristics of a palm based gesture using diagonal line features and average grayscale. Inner-hand regions (such inside the palms) and normal size fingers are detected through f_1, while f_3 captures the ring finger based on edge properties. Hence, this is quite different from traditional gesture classifiers which rely on parametric models or statistical properties of the gesture. Note, that the result is a set of common features for all three of our pose gestures. The optimal partition of the bounding box was 2x3 giving 6 feature rectangles. The parameter search routine found both the number of sub blocks and the type of Haar feature to assign to each.

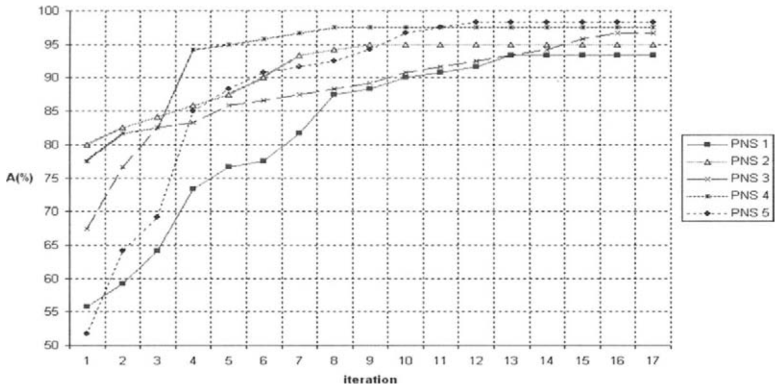


Fig. 8. Convergence curve for five solutions of the PNS algorithm.

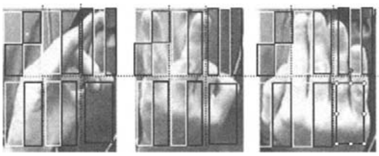


Fig. 9. Automatically selected features (f_4, f_1, f_3, f_1 ,f_1 , f_5) for the 2x3 partition

5 Test of the Hand Gesture FCM Classifier

To evaluate the overall performance of the hand gesture tracking and FCM recognition system, we used the Azyxxi Real-time Repository ™ [1], which was designed to accommodate multi-data types. The data-set consists of 20 trials of each of 4 tasks: Select Record of Patient, Browse X-ray collection, Select specific X-ray and Zoom in Damaged Area. The user was asked to perform the tasks sequentially. The total results for one experienced user are shown in Table 1. The success task rate shows the times an action (part of the task) was performed correctly without catastrophic errors. Minor errors are related to inaccurate position of the cursor due to fast movements or changes in direction, while catastrophic errors occurred as a result of misclassification of the supervised FCM. In general, the results below indicate both the ability of the system to successfully track dynamic postures; and classify them with a high level of accuracy.

Table 1. Results of medical tasks using hand gestures

Task	Steps	Trials	Success Task
Select Record of Patient	1	19	94.74%
Browse X-ray collection	2	20	100%
Select specific X-ray	1	20	100%
Zoom in Damaged Area	2	19	94.74%

6 Conclusions

In this paper, we consider a vision-based system that can interpret a user's gestures in real time to manipulate objects within a medical data visualization environment. A hand segmentation procedure first extracts binary hand blobs from each frame of an acquired image sequence. Dynamic navigation gestures are translated to commands based on their relative positions on the screen. Static gesture poses are identified to execute nondirectional commands. This is accomplished by using Haar-like features to represent the shape of the hand. These features are then input to the FCM algorithm for pose classification. The PNS algorithm is employed to automatically select a small number of visual features, and to tune the FCM algorithm. Intelligent handling of features allows non discriminating regions of the image to be quickly discarded while spending more computation on promising discriminating regions. The gesture recognition system was implemented in a sterile medical data-browser environment [11] . Test results on four interface tasks showed that the use of these simple features with the supervised FCM yielded successful performance rates of 95 to 100%, which is considered accurate enough for medical browsing and navigation tasks in hospital environments. The explanation for the 5% drop in accuracy is due to the confusion between the 'double click' and 'drag' gestures, as a result of lack of samples in the training and testing sets. In a future study gestures that include shadows, occlusion and change in geometry will be used to enrich the datasets. Another issue to be addressed is false triggers as a result of a fast moving objects others than the hand. An approach to this problem is to store a generic 2D skin color distribution trained off line, and to compare it to the candidate object color histogram. Catastrophic errors due to confusion between gestures can be reduced by using the probabilities of gesture occurrences in a transition matrix based on the state machine presented in Fig. 5. An appealing alternative method for fast recognition of a large vocabulary of human gestures suggests using Haar features to reduce dimensionality in hand attention images. Future work includes recognition of dynamic two handed gestures for zooming, rotating images, and testing with larger gesture vocabularies.

Acknowledgments

This project was partially supported by the Paul Ivanier Center for Robotics Research & Production Management, Ben Gurion University.

References

1. Azyxxi Online Source (2003) Available: http://www.imedi.org/dataman.pl?c=lib&dir=docs/Azyxxi
2. Bradski GR (1998) Computer vision face tracking for use in a perceptual user interface. In Intel Technical Journal, pp 1-15.
3. Foley JD, van Dam A, Feiner SK and Hughes JF (1987) Computer graphics: principles and practice, 2 Ed, Addison Wesley
4. Graetzel C, Fong TW, Grange S, and Baur C (2004) A non-contact mouse for surgeon-computer interaction. J Tech and Health Care 12:3:245-257
5. Lienhart R and Maydt J (2002) An extended set of haar-like features for rapid object detection. In IEEE ICIP 2002 vol:1, pp 900-903
6. Nishikawa A, Hosoi T, Koara K, Negoro D, Hikita A, Asano S, Kakutani H, Miyazaki F, Sekimoto M, Yasui M, Miyake Y, Takiguchi S, and Monden M (2003) FAce MOUSe: A novel human-machine interface for controlling the position of a laparoscope. IEEE Trans on Robotics and Automation 19:5:825-841.
7. Schultz M, Gill J, Zubairi S, Huber R, Gordin F (2003) Bacterial contamination of computer keyboards in a teaching hospital. Infect Control Hosp Epidemiol 24:302-303
8. Stern H, Wachs JP, Edan Y (2004) Parameter calibration for reconfiguration of a hand gesture tele-robotic control system. In Proc of USA Symp on Flexible Automat, Denver, Colorado, July 19-21
9. Viola P and Jones M (2001) Rapid object detection using a boosted cascade of simple features. In IEEE Conf on Computer Vision and Pattern Recog, Kauai, Hawaii
10. Wachs JP, Stern H, and Edan Y (2005) Cluster labeling and parameter estimation for automated set up of a hand gesture recognition system. In IEEE Trans in SMC Part A 2005. vol. 35, no. 6, pp: 932- 944.
11. Wachs JP, Stern H (2005) Hand gesture interface for med visual app web site. Available: http://www.imedi.org/docs/references/gesture.htm
12. Yanagihara Y, Hiromitsu H (2000) System for selecting and generating images controlled by eye movements applicable to CT image display, Medical Imaging Technology, September, vol.18, no.5, pp 725-733
13. Zeng TJ, Wang Y, Freedman MT and Mun SK (1997) Finger Tracking for Breast Palpation Quantification using Color Image Features. SPIE Optical Eng 36:12, pp 3455-3461