IJMI
International Journal of Machine Intelligence
ISSN: 0975–2927 & E-ISSN: 0975–9166, Volume 3, Issue 4, 2011, pp-268-271
Available online at http://www.bioinfo.in/contents.php?id=31

GESTURE BASED DESKTOP INTERACTION

SREEKANTH N.S.[1]*, GOPINATH P.[2] , SUPRIYA N PAL[3], NARAYANAN N.K.[4]
[1]Center for Development of Advanced Computing ,#68 Electronic city, Bangalore, 560100, Email- nss@cdac.in
[2]Center for Development of Advanced Computing ,#68 Electronic city, Bangalore, 560100, Email- gopinath@cdac.in
[3]Center for Development of Advanced Computing ,#68 Electronic city, Bangalore, 560100, Email- supriya@cdac.in
[4]Department of Information Technology, Kannur Univeristy, Kerala, Email- nknarayanan@gmail.com
*Corresponding Author: Email- nss@cdac.in

Received: November 06, 2011; Accepted: December 09, 2011

Abstract- This paper discusses the implementation details of computer vision based gesture processing. Finger gestures are processed and recognized for interacting with the desktop and desktop applications. Gestures are captured via webcam; and conventional image processing techniques and algorithms are used for video frame analysis. For identifying the object of interest in each video frame BLOB colour-segmentation algorithm is used. 8-directional code features are extracted from the motion vector for recognizing of gestures.
Key words - Multimodal Interface, Vision based interaction, gesture based interaction, motion vector.

Introduction

The gesture recognition opened up new mode of interaction in Human Computer Interface. Gestures have long been considered an interaction technique that can potentially deliver more natural, creative and intuitive methods for communicating with computers[1].The gestures are commonly used in human-human interaction and plays a major role in communication when the participants are unable to speak, or the situation does not allow the participant to speak etc. The gestures also play as an offset-input to the other mode of communication, for example gesture and speech are co-expressive and they form a part of rich human conversational features [2,3].

If we look at the evolution of gesture based interaction initially there were glove based devices, but they lacked the naturalness factor as they had introduced an additional hardware constraints on the user. The models employed for gesture processing are either 3D models or image based processing.

The former lacks the computational efficiency and the simplicity compared to other. In the image based processing method there are several techniques based on color, contour and correlation for identifying gestures [4]. The present work focuses on the image based processing and follows the algorithms and procedures of Machine vision a.k.a computer vision. Here gestures are the predefined orientation and movement of fingers / palm in space. These movements and orientation are tracked and identified as appropriate gestures.

Machine Vision based Gesture processing

In this study we have explored the possibility of using gestures for communicating with desktops and desktop applications. Computer vision based soft solution is proposed; only web cam is required for interacting with the system. The movement of fingers/ palm is tracked and mapped to the screen coordinates. The gestures are the pre-defined pattern of movement of object (palm/finger). As we are following the machine vision based algorithms the individual frames of the video are processed to analyze the gestures. For easy processing and object tracking a predefined colour sticker will be placed on the fingers. The movements of the object of predefined colour are the gestures. The system has to be trained for individual users for better performance. The training module will collect few samples w.r.t various gestures and extracts the required parameters. The correlation between the training sample and the input will be calculated and appropriate decision will be made by the system during recognition (classification) phase.

Selection of Gestures for communication

There are many guidelines and strategies for selecting appropriate gestures for communication[5]. Depending on the type of operation user want to perform, we have to choose appropriate gestures. Some important operations we commonly found across any application are pointing, selection, page-up and page-down (a.k.a scrolling), zooming, selecting or moving to next or previous object or event etc. It is quite natural that we use fingers/palm as an

auxiliary form for communication along with speech to point an object or to express certain ideas or event or emphasis certain words in speech during a discussion. For example, we may move palm or fingers towards right for expressing move forward or towards next and similarly towards left for moving backward or accessing the previous object. In this study, the uses of such gestures for interacting with desktop and desktop applications are experimented.

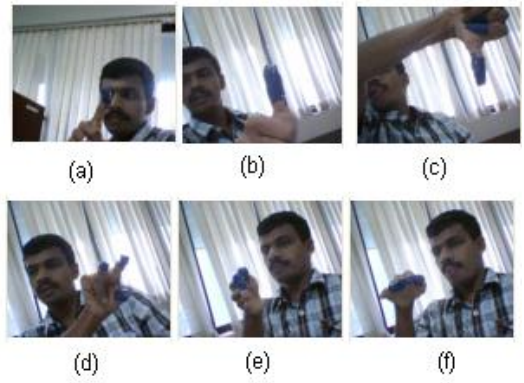


Fig.(1) Different gestures for communication; (a) pointing (b) ok (c) cancel (d) zoom-in/out depends on the movement of fingers (e) move next or previous depends on the movement of palm (f) move up or down depends on the movement of palm

Few gestures are identified which can be used for interacting with the systems for performing common operation across any applications. Here finger/palm gestures are captured with a web cam and the video frames are processed for analyzing and recognizing the gestures. For easy identification of the object (palm/finger) we use certain colored sticker in to the fingers. Some of commonly used gestures are shown in Fig.(**1**)

Architecture of the system

The system comprised of two modules ; training module and the gesture recognition module. Like any machine learning based classification system, training module will collect the data and extract the feature vectors prior to testing or using the system. This vector is used as a reference value for classification. The gesture recognition system/module have following components. Image acquisition, segmentation, motion vector generation, gesture analysis and recognition, vocabulary mapping and feedback. Actually the training module is a subsystem of recognition system. First three components are common for both training and recognition system. After motion vector generation training module will store those feature vectors in database and will be used for the classification purpose. Training module is not discussed here in detail; as its components are similar to recognition module. Image acquisition is a process of capturing image frames from the camera and the frames have to be given for next level of processing, i.e., segmentation. After segmentation, the object of interest will be isolated from the image and control will pass to motion vector generation module. This is a component which will keep track or record the locus of points of object of interest. Once locus of points are generated (motion vector) the 8-directional codes will be generated. The directional code with the orientation of the object will be given as input to the gesture analysis and recognition module. This module will analyze the pattern and it also identifies the appropriate gestures which are apt to the situation and corresponding gesture string will be generated. The string will be given to the vocabulary mapping module where transformation of user gesture in to system understandable form takes place. It also prompts the user to give appropriate and unambiguous input. The following diagram gives the module wise structure of the system.

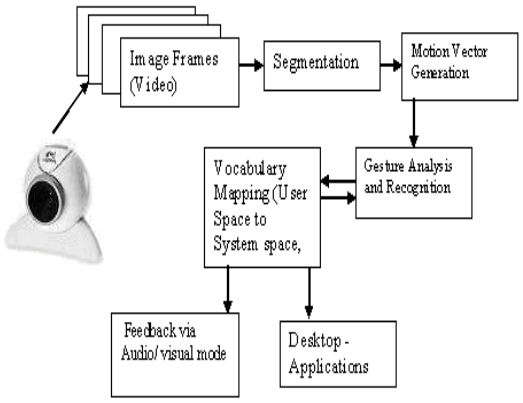


Fig.(2) Architecture of the system

Segmentation

As we have discussed in previous section the blue colour stickers are placed on the fingers. The occurrence of blue colour has to be identified and isolated from each image frame. The movement and orientation of the fingers (colour) also have to be recorded for gesture recognition. To identify the region of pixels (blue colour region), blob coloring algorithm is used. Blob coloring is a technique used to find regions of similar color in an image. The algorithm works by passing a "backwards L" shaped template over the image from left to right and top to bottom Fig.(**3**).

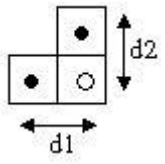


Fig.(3) "Backwards L" shaped template

The template is used to calculate the distance between the current pixel represented as circle, and the one to the left, and the one above represented as dots [6]. For an RGB image this

distance is equal to the Euclidean distance in the RGB color space. Let m be the current pixel and z the one of the neighbor pixel under consideration, then the Euclidean distance between z and m is calculated as

$$D(z,m) = \| z-m\|$$

$$= [(z-m)^T (z-m)]^{1/2}$$

$$= [(z_R-m_R)^2+ (z_G-m_G)^2 +(z_B-m_B)^2]^{1/2}$$

Where || . || represent the norms of the argument and subscripts R,G,B denotes the RGB components of vectors m and z. The locus of points such that D (z,m) < Threshold value it can be counted as a part of the pixel[7].

Segmentation – Isolating Object of Interest

For different gestures, user has to express with different pattern of palm and finger positions with predefined movement. Some gestures are computationally simple to recognize, with single blob unit in each frame. This category we only requires the orientation of object in certain direction. The computationally simple gestures are pointing gestures (finger point to screen), cancel, ok etc. Fig.(**4**) a, Fig.(**4**) b. which only requires the object orientation in space. The gestures like move left, move right, move up, move down gestures are computationally moderate complexity. All these gestures required object tracking and motion vector generation. Fig.(**4**) b, Fig.(**4**) c.

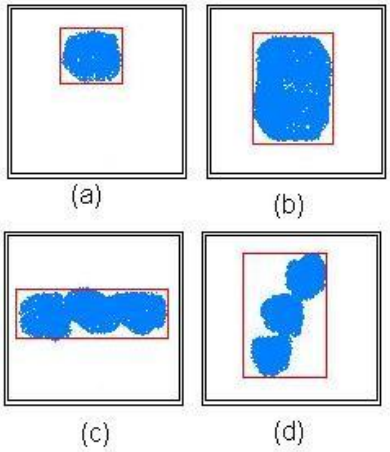
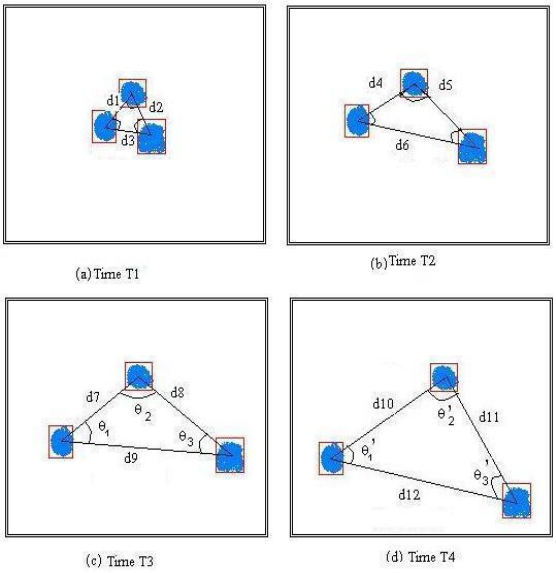


Fig.(4) Simple gestures with single blob unit, The blue region is finger/s identified in individual frames which is called blob (a) point (b) cancel (c) move up/down (depends on motion vector)(d) move to next/previous(depends on motion vector).

So far we have discussed the gestures with single blob unit. The gestures like zoom in and zoom out are computationally complex with more than one blob unit. Fig.(**5**), shows the frames at different time intervals T1, T2, T3, T4 for zoom-in. While zoom in action the three fingers will move apart as time progress. The distances between the fingers or its corresponding angles in consecutive frames are measured for identifying the gestures zoom-in and zoom-out. For zoom-out the same pattern in opposite direction, i.e fingers will come closer. Here the movement of more than one blob unit has to be tracked.



Fig(5) Gestures with more than one blob unit. Zoom-in gesture frames in increasing order of time. The blue colour regions are fingers which move apart for zoom-in and come closer for zoom-out

Motion Vector Generation

After the object of interest is isolated, whose movements have to be tracked and corresponding motion vector will be generated. As we have discussed in previous section certain gestures only require the orientation information; which does not carry the motion vector. This category we call it as scalar gestures, eg. cancel, ok , open close and pointing gestures. Another class which requires the information about the orientation of the object along with the direction of movement. We call this category as vector gestures. The gestures like move next, move previous, move-up move-down etc are vector gestures. The vector gesture can also have complex structure with more than one blob unit. For example zoom-in, zoom out gestures have more than one blob unit and their movement pattern decide the type of gesture. The motion vector generation module tracks the object movement and it will keep track of all locus of points ie. position coordinates.

Motion vector will be a 2 dimensional vector which can store the x and y coordinate associated with object at each moment. This vector will convert in to equivalent 8-directional codes. The directional codes are widely used feature vector for online handwriting recognition. Same method used here also for recognition of gestures [8]. For scalar gestures the motion vector will be assigned as 0. For pointing and navigation of cursor through screen, directional codes will not be generated.

Gesture Analysis and Recognition

The expressed gesture is to be recognized and which will be mapped to equivalent system command. The motion vector and orientation information corresponds to the expressed gestures will be generated. The generated orientation information (whose histogram) along with the directional code features will be compared with the pre-stored template. The Levenshtein Minimum edit distance [9] algorithm is used for the directional code string comparison. If minimum edit distance values for two strings found to be equal, the probability of occurrence of the string on that particular context will be analyzed i.e Max P(S / C) will be selected, i.e probability of occurrence of a string 'S' at the particular context C, where the value for C will be assigned based on the currently active application, and relevance of the operation at that context.

Vocabulary Mapping and feedback

Once gesture is recognized the meaning of it has be converted in to system understandable form[10] and if any error, that has to be notified. The recognized gestures will be in the form of <action><parameters> format this has to be converted in to equivalent system understandable format. For example for the move up gesture the equivalent code generated will be <move-up> so a interrupt will be generated for moving the scroll bar down. This operation is valid only if there is a vertical scroll bar seen in the display area. In absence of scroll bar the system will not respond or can give a feed back to user by stating that "nothing to move-up". The feedback can be audio or visual format.

Implementation Details

The system implemented in java with media frame work support. The objects of interest (fingers) are marked with blue color for easy tracking. To reduce the search region in frames, once the object is located in one frame, search for object only in a small region around the previously identified location (here 50 * 50 region). If the object is lost then search the entire frame to locate the object and once located follow the previous steps. In this way we can considerably reduce the searching time. The probability of context dependent gestures has to be stored in a separate file. The values can be generated manually or dynamically while using the system. In this study this file is manually generated.

Conclusion and Future work

Ten gestures are implemented for this study. The system is tested in two types of environment. The first one with trained user who knows about the pros and cons of the system and the other category is the first time user who has very minimal knowledge about the system. For a trained user it is giving more than 80 percent accuracy where as the first time user it is found to be 40 to 50 percent. More gestures have to be identified, which can be implemented for better prediction of this method.

References

[1] Maria Karam (2006) *PhD thesis, University Of Southampton.*

[2] Francis Quek, David McNeilly, Robert Bryll, Susan Duncan, Xin-Feng Ma, Cemil Kirbas, Karl E. McCulloughy, and Rashid Ansari (2002) *ACM Transactions on Computer-Human Interaction*, Vol. 9, No. 3, Pages 171–193

[3] Francis Quek (2003) *Proceedings of the Ninth IEEE International Conference on Computer Vision (ICCV 2003) 2-Volume Set*.

[4] Qing Chen (2008) *PhD Thesis, Canada .*

[5] Schapira E. and Shrma R. (2001) *Proceedings of the 2001 workshop on Perceptive user interfaces ACM New York, NY, USA 2001* . http://www.cs.ucsb.edu/PUI/PUIWorkshop/PUI-2001/a3.pdf

[6] Wöllert, Thomas (Dipl.-Inf. (FH)) (2006) *Computer Graphics and Image Processing, Munich University of Applied Sciences*, Germany.

[7] Rafael. C Gonzalez, Richard. E Woods, Steven .L Eddin (2005) *Pearson Education –Third Indian Reprint -2005* , pp.251

[8] Manikandan B. J., Gowri Shankar, Anoop V., Datta A., Chakravarthy V. S. (2002) *Proceeding of the International Conference on Natural Language Processing (ICON-2002) Vikas Publishing House Pvt. Ltd*. pp.285-291.

[9] Algorithm Implementation /String/Levenshtein distance wikibooks *http://en.wikibooks.org/wiki/Algorithm_Implementation/Strings/Levenshtein_distance*

[10] Sreekanth N.S., Supriya N. Pal, MeeraThomas, Ahamad Hassan, Narayanan N. K. (2009) International *Journal of Information Studies*, Vol 1. No .2, pp 131-137 .