ashutosh.saxena@ieee.org

CSIDC 2003 Interim Report

Country :	India
University :	Indian Institute of Technology Kanpur
District :	Kanpur – 208016

Team number: 2

Mentor: Prof. A. K. Chaturvedi
akc@iitk.ac.in
+91-512-2597613

SANKET: Interprets your Hand Gestures

SANKET

Interprets your Hand Gestures

Team Members:

Ashutosh Saxena	ashutosh.saxena@iitk.ac.in	Electrical	20	Male
Aditya Awasthi	awasthi@iitk.ac.in	Electrical	21	Male
Vaibhav Vaish	vaibhav.vaish@iitk.ac.in	Electrical	20	Male

I. INTRODUCTION

The use of hand gestures provides an attractive alternative [2-3] to cumbersome interface devices for human-computer interaction (HCI). In particular, visual interpretation of hand gestures can help in achieving the ease and naturalness desired for HCI.

The goal of our project is to develop a real-time system capable of understanding commands given by Hand gestures. The user must be able to communicate to computer all basic commands required by a human-computer interface. Our system will provide following functions:

- Dual Mouse pointer motion
- About ten motion-based gestures—sufficient to operate a browser.
- Static gestures—to input digits and other common inputs. Dialogue windows having "OK" and "Cancel" buttons can be answered.
- Provision for interfacing external hardware (e.g. a robot) via a port to our system.

To operate computer with hand gestures, no accessories like gloves are needed in our system.

II. BENEFITS OF PROJECT

Working with computers has become an integral feature of our society. Although computer work is not directly harmful to our health, there is a link between working with computers and the development of injuries. Various muscle and tendon disorders are caused due to continuous and improper use of keyboard and mouse [1].

The project offers a new human-computer interface based on hand gestures, which allows users to do basic operations on a computer. The system will greatly benefit physically challenged persons.

Further, our system provides an option to connect external machinery like robotic arm, which require precise and complicated (human type) manipulations.

Multimodal inputs to computer have recently become popular. Our project offers a new input channel to a computer. The application developed can be easily integrated into an operating system, enhancing its user-friendliness tremendously.

III. INNOVATION

Although a lot of research has been done recently on Hand Gesture Recognition and there exist algorithms for this purpose, but there is no product currently in the market to the best of our knowledge which offers hand gesture recognition. Our project develops an application that recognizes hand gestures and then generates events that can be passed on to other applications/ operating system.

Our system follows a hybrid approach. It recognizes both motion-based and static hand gestures. To implement the algorithm given real-time constraints was one of the most difficult tasks. Our system is a novel application that allows communication of most necessary commands to computer. Further, our system can work with any camera that supports streaming video input to the computer.

IV. SYSTEM ORGANIZATION

The only hardware external to the system is a camera. The camera can be connected to the computer via an appropriate port. Any camera that supports streaming video input can be used. We have tested our system with following cameras: Sony DSC P51[1], Intel Easy PC Camera CS110 (online), Lego Vision Command CCD camera.

We have used OpenCV[2] library which is supported on common operating systems like Windows and Linux. The C++ code can be used on any platform. Results in this report are given for the following system—AMD Athlon 2000, 266 MHz FSB, 256 MB DDR RAM. The compiler used was Microsoft Visual C++.

---

[1] Does not support streaming video, analysis done on stored video clips.

[2] Intel Open Source Computer Vision Library. [Online] http://www.intel.com/research/mrl/research/opencv/

V. PRINCIPLES OF OPERATION

We follow a hybrid approach to Hand Gesture Recognition. We intend to recognize both static and motion-based gestures. The images are captured from Camera and then passed to the algorithm for recognition. The various steps of the algorithm are described below:

1. **Hand Segmentation:** From the image acquired from camera, the region of interest, i.e., hands have to be localized. The feature we use for this purpose is skin color. The image is transformed in YCbCr [5] color space. As shown in fig 1, Skin color is modeled as a two dimensional Gaussian in Cb-Cr space—neglecting Y component takes care of lighting variations. *Mahalnobis* [5] distance is used to estimate pixels which lie in hand region. The distribution of Cb-Cr components for skin was made by manually selecting skin regions in hundreds of images. During actual execution, the distribution is modified to match the distribution of the particular person and environmental conditions. The binary Image is subjected to *Morphological*, *Connected Component Operators* and further heuristics to separate hand regions from other objects like face, noise, etc. Fig. 2 shows the original image, and fig. 3 shows the binary Image obtained by applying mahalnobis distance in YCbCr color space. Morphological operators and other heuristics remove noise and select only hand regions.

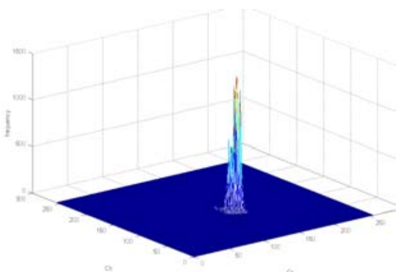

Fig. 1. Skin color distribution in Cb-Cr color space.

Fig 2. Original RGB Image

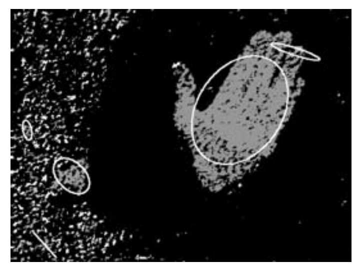

Fig 3. Binary Image without morphological operators. (Obtained connected components shown by ellipses).

2. **Affine transforms** for hand regions between successive image frames are estimated using local descent criterion [3]. Hands are approximated as ellipses. The affine transform parameters (translation, rotation, scaling, and shearing) and ellipse parameters represent the Feature vector (centroid, orientation, major/minor axes, shear) for each hand (for subsequent frames).
3. **Determining intended hand gesture** from the feature vector. The feature vector is transformed to a most expressive representation in a 6-dimensional space. The space is partitioned into regions—each representing a gesture.
4. **Static hand Gestures:** Apart from the above discussed motion-based gestures, we intend to recognize certain static gestures like OK, Cancel, and numbers from 1 to 5. The static gestures for these are shown in fig 4.

Fig. 4. Static Hand Gestures. From left: “OK”, “One”, “Two”, “Three”, “Zero”. The image is output of the hand localization algorithm.

Till now, we have developed and implemented (1)-(3). Part (4) is under developmental stage.

VI. DESIGN STRATEGY

We follow a modular approach to coding. Each module corresponds to a specific operation of the algorithm. Code can be roughly classified into three broad divisions:

1. Extraction of hand(s) in each frame of acquired video stream (from camera or movie clip). After first frame, processing is done only on a localized portion of the image (enclosing hand regions).
2. Extraction of "feature vector"—based on algorithm discussed above, and recognition of gestures from the feature vector.
3. Generating events (signals/interrupts) based on gestures and passing them on to a) operating system, b) a software application, c) external hardware via a parallel/serial port.

VII. COST

The only hardware external to computer system is Camera. We have worked with the following cameras: Sony DSC P51 (does not supports streaming video), Intel Easy PC Camera CS110 (online), Lego Vision Command CCD camera. Cost of none of these cameras exceeds 400$. Further, our system is flexible and can work with any camera with certain constraints.

VIII. TEAM ORGANIZATION

Although we worked together for the project, the team can be divided the team into two parts:

- Simulators: Mr. Ashutosh Saxena was in this subgroup. This group was responsible for identifying which algorithms to use and then simulating them on Matlab. Judging and selection of appropriate algorithm and tuning the parameters of the algorithm was the major job of this group.
- Coders: Mr. Aditya Awasthi and Mr. Vaibhav Vaish were in this subgroup. This group was responsible for coding the given algorithm in C++, and optimizing the algorithm for real-time operation. Development of appropriate GUI and interfacing of Camera constituted other jobs of this group.

Major decisions were taken collectively keeping in mind both the final objective, feasibility and real-time constraints. Some of the complicated algorithms [4] (simulated on Matlab) had to be simplified to satisfy real-time constraints.

IX. RESULTS

A. Results Achieved

Till now, we have recognized some of the motion based gestures. Results for the following gestures are reported:

- **Dual Mouse pointer motion**: Table 1 lists the resolution achieved for mouse pointers representing the two hands for various cameras and various subjects. Subjects were asked to move the mouse pointer and click on a position in MxN grid. Position, velocity and acceleration of the centroid are used to determine pointer position and motion.
- **Dynamic Gesture Recognition**: The gestures are discussed below. Please note that the action performed after identifying gestures is only indicative and can be used by an application as desired:
    - *No operation (NOP):* It is essential that the computer performs no actions unless specifically intended by the user. We have taken a large rejection rate so that no action is performed. A specific region in F-space represents *NOP*.
    - *Click*: This action consists of two sequences of events: Press and Release for each mouse pointer. Sudden opening of palm represents event "Press" and sudden closing of palm represents "Release". "Click" consists of "Press" followed by "Release" within 10 frames.
    - *Rotation*: Rotating each hand independently is detected, as well joining both hands and rotating for an emphasized action.

- *Window Resize*: For this action, two hands need to be moved with equal speed in opposite direction. The direction, speed, and distance moved, determine the change in window size (length and width).
- *Back/forward*: First both hands closed and then opening them in particular sequences initiates events:
    - *Open left followed by right:* Forward
    - *Open right followed by left:* Back
    - *Close left followed by right:* Stop/Reload
    - *Close right followed by left:* Close current window

Fig 5. Hand as a switch.

- *Switch:* One hand is used as a switch, as shown in fig. 5.
- *Dynamic "Opera-type" gestures:* By observing the centroid motion of hands for various successive frames, following gestures can be identified. Please note that speed of the hand movement for these gestures must be above a certain threshold. At this stage, only simplified version of this has been implemented.
    - *Duplicate page*: Hold right button, move down then up
    - *Restore or maximize page*: Hold right button, move up then right
    - *Minimize page*: Hold right button, move down then left
    - *Close page*: Hold right button, move down then right

Table 1. Mouse Sensitivity achieved with various cameras and users.

Camera	Resolution	Motion blur	Resolution (grid size, %area)		
			User 1	User 2	User 3
Sony DSC P51*	320 x 240	Very low	32x24, 0.26%	32x24, 0.26%	32x24, 0.26%
Intel Easy PC Camera CS110	160 x 120	Medium	16x12, 0.52%	16x12, 0.52%	12x9, 0.93%
Lego Vision Command	640 x 480	High	32x24, 0.26%	32x24, 0.26%	32x24, 0.26%

* Experiments performed on stored movie clips.

B. Outcome at the end of the project

At the completion of our project, we intend to realize a seamless hand-gesture based human-computer interface. Specifically, we intend to identify the following gestures:

- **Motion-based Gestures,** simple versions of which have been discussed in section IX (A).
- **Static gestures** representing certain events like: "OK", "Cancel", "One", "Two", "Three", "Four", "Five".

The recognized gestures will act as input to a) software applications or operating system and b) for manipulating external machinery via parallel/serial port.

REFERENCES

[1] *Computer Ergonomics and health*, [Online] http://www2.umdnj.edu/eohssweb/ergo/msd.htm

[2] Yuntao Cui, et al., *A Learning-Based Prediction-and-Verification Segmentation Scheme for Hand Sign Image Sequence,* IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 21, no. 8, August 1999.

[3] Ming-Hsuan Yang, Ahuja, N., Tabb, M, "Extraction of 2D motion trajectories and its application to hand gesture recognition", *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, Volume: 24 Issue: 8, Aug 2002, Page(s): 1061 -1074N.

[4] Mark Tabb and Narendra Ahuja, "Multiscale Image Segmentation by Integrated Edge and Region Detection", *IEEE Transactions on Image Processing,* vol. 6, no. 5, May 1997.

[5] Mayank Bomb, IT-BHU, "Color Based Image Segmentation using Mahalnobis Distance in the YCbCr Color Space for Gesture Recognition*", IEEE India Council MV Chauhan Student Paper Contest 2002*.