Touchless Feed: A Sixth Sense based Innovative

Abhishek Sambyal
Department of Computer Science Engineering
SMVDU, Katra
J&K, India-182320

Sparsh Sharma
Department of Computer Science Engineering
SMVDU, Katra
J&K, India-182320

Naveen Kumar Gondhi, Ph.D
Department of Computer Science Engineering
SMVDU, Katra
J&K, India-182320

ABSTRACT
Touchless Feed is a gestural interface for future computing which aims at using computer from far. It allows us to interact with our PC easily with the help of hand gestures. Here basically the mouse pointer is controlled by movement of fingers which are covered by some color markers. The webcam captures the position of our fingers and as the position changes; the pointer of the mouse is moved accordingly. Various hand gestures are also recognized by the camera to perform specific tasks. The objective is to interact with the computer through gestures. The Aforge.net library in C# contains the various libraries which help in motion detection, pattern recognition, & hence taking suitable actions to interact with the computer.

General Terms
Pattern Recognition, Algorithm

Keywords
Touchless feed, Gesture, AForge, Blobs.

1. INTRODUCTION
People are restricted to sit and be in touch with the computer even for minor jobs like opening a document, turning off the pc, controlling the volume of speakers and some other small works. It is a tedious task to reach the system again and again even for some clicks. So the use of computer is not as flexible and easy as it could be with the use of this technology. Also the gestural interfaces that are used today are not as interactive and flexible as they can be. Therefore a technology is required which enables interaction in a healthy way with the computer thereby making people more synergistic. In the nutshell it is built in order to make the use of computer more user friendly. It is not something completely new but an extension to an existing technology.

An open source framework of C# namely AForge.net is used which is specifically designed for developers and researchers who are working in the field of Artificial Intelligence and Computer Vision robotics, image processing, neural networks, machine learning, genetic algorithms, fuzzy logic etc [10]. This framework includes the set of libraries and sample applications, thereby demonstrating their features: AForge.Imaging- library having filters & image processing routines; AForge.Video- set of libraries for video processing; AForge.Fuzzy- fuzzy computations library; AForge.Robotics- library providing support of few robotics kits; AForge.MachineLearning- machine learning library [10]. The following classes of **AForge.NET** are used:

A. HSL Filtering
This image processing filter operates in HSL color space and its work is to filter the pixels; which color is inside/outside of the chosen HSL range - keeping pixels with colors inside/outside of the chosen range and filling the rest with specified color as shown in fig 1[10].

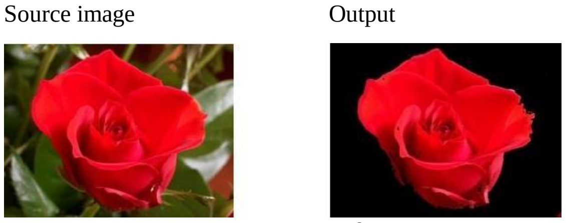

Fig 1. Applying HSL filtering[10]

B. Grayscale
It converts color image into grayscale image. This class is the base class for image gray scaling. Other classes also inherit from this class and specify **RGB** coefficients used for color image conversion to grayscale as shown in fig 2[10].

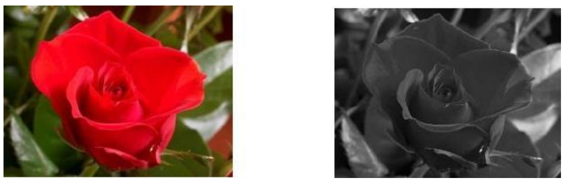

Fig 2. Applying Grayscale[10]

C. Threshold
This filter does image binarization using specified threshold value. All the pixels with intensities equal or higher than threshold value are converted to white pixels. All other pixels with intensities below threshold value are converted to black pixels as shown in fig 3[10].

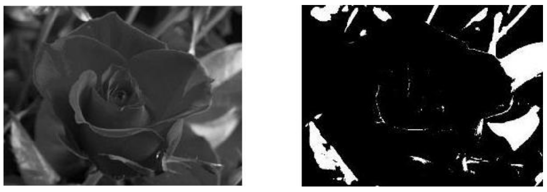

Fig 3. Applying Threshold[10]

D. Extract Biggest Blob
The filter locates the biggest blob in the source image and extracts it. The filter also can use the source image for the biggest blob's location only, but extract it from another image, which is set using Original Image property as shown in fig 4. The original image usually is the source of the processed image[10].

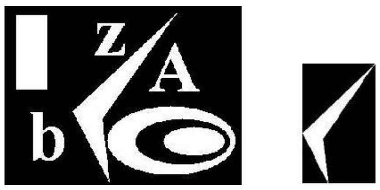

Fig 4. Extracting Biggest Blob[10]

E. VideoCaptureDevice Class

This video source class captures the video data from local video capture device, like frame grabber, USB web camera (or internal), capture board - anything which supports **DirectShow** interface. For devices having a shutter button or support external software triggering, the class also allows to do snapshots. Both video size and snapshot size can be configured[10].

The DFD of the proposed system will be as shown below:-

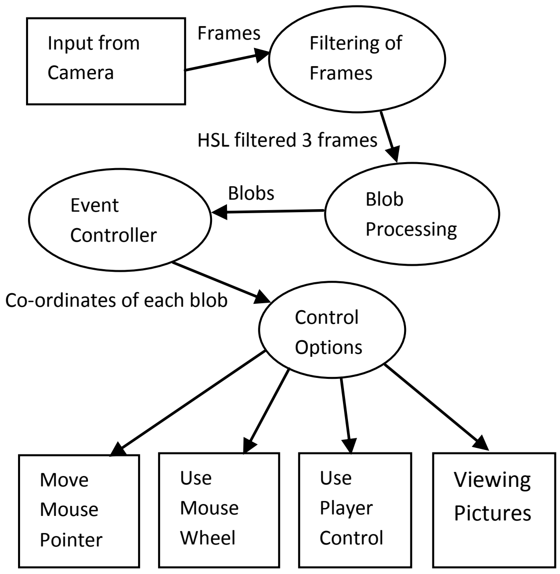

Fig 5. Level 1 DFD

2. EXPERIMENTAL METHOD

A. Preliminary

First select a video source. For this a VideoCaptureDevice class is used. There is already a file named ColorFilterValues.txt saved which has the basic values of HSL for the three primary colors- Red, Green, and Blue which will be loaded and the process will start. The main screen will be displayed which will have the options of color settings, camera options and marker controls. Also there will be a small screen called CView which will be displayed at the bottom right corner of the screen. This screen will display the image as captured by the camera.

B. Adjustments/Settings

The basic values of HSL for all the three colors will be loaded onto the screen. But since the light intensity varies at different places for different colors so there might be a possibility that the main screen will not detect the three colors correctly. So the adjustments have to be made to make the screen detect the three colors accurately. For this there is a Color Settings option on the main screen. On selecting this option an access will be available to the three characteristics of a color- Hue, Saturation and, luminance

Each of these characteristic will be provided with a minimum and maximum option. Then select the desired values by varying the values between minimum and maximum range. Also a Histogram is also available below these characteristics. The function of the histogram is that only the desired selected color will be visible in the white shade while the whole screen will appear to be black. So the adjustments can be made for all the three colors by changing their hue, saturation, and luminance and looking for the desired result in the histogram.

C. Control Options

Now after the adjustments are made for the three colors and the screen is able to detect the three colors accurately, chose the control option which you want to perform. As displayed on the main screen, there are three control options: 1. Move the mouse pointer (blue) 2. Use of mouse wheel (green + blue) 3. Use of player control (red)

For moving the mouse pointer, the color designated is Blue. As the blue marker is moved in front of the camera the mouse pointer will move on the screen corresponding to it. Thus all the functions of mouse can be performed.

Now to use mouse wheel, enable the second option. This will cause the screen to detect the green and blue color only, ignoring all others. Many functions have been assigned to the use of these 2 colors like zooming in and zooming out the image, copy-paste, and control running programs or active windows etc.

For some special functions the red marker is used. Like for the use of media player which involves controlling the level of volume, selecting the previous and next songs, and viewing of the pictures etc. this control option will be enabled.

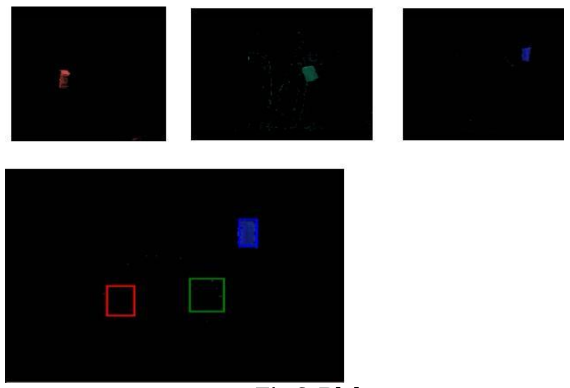

Fig 6. Blobs

D. Processing the image

After the image is obtained on the screen, next step is to get the coordinates of blobs.

First of all the frame obtained is rotated to 180 degree and flipped. This is done so as to match the direction sense i.e. to match the images which are initially mirror images of each other. The image as obtained is shown in fig 7 as shown below.

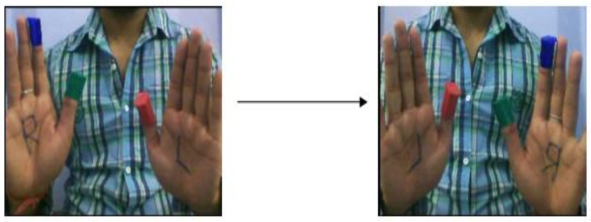

Fig 7. Image As Processed By Computer

Three copies of the resulting frame are taken and each frame is applied with red filter, green filter and blue filter respectively.

The various bits obtained after applying grayscale are then locked and threshold is applied to them. This filter does image Binarization using specified threshold value. All pixels with intensities equal or higher than threshold value are converted to white pixels. All other pixels with intensities below threshold value are converted to black pixels.

The bits are unlocked and blob formation takes place. Various blobs are formed and these blobs are stored in an array. Next step is to extract the biggest blob from all the blobs obtained. For this ExtractBiggestBlob class is used. The filter locates the biggest blob in the source image and extracts it. The filter also can use the source image for the biggest blob's location only, but extract it from another image, which is set using OriginalImage property. The original image usually is the source of the processed image.

Now using pen and rectangle classes the rectangle is drawn around each blob. Then finally the dimensions of this blob are sent to the event controller for further processing.

E. Event Controller

1. Now for each blob the following dimensions are received by event controller: x-co-ordinate, y-co-ordinate, height, and width as shown in fig 8.

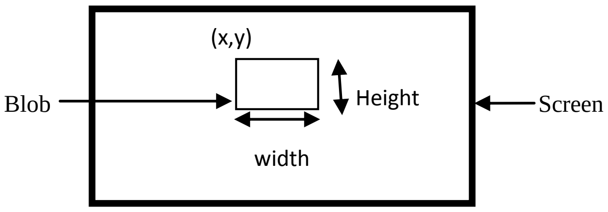

Fig 8. Dimensions of Blob

2. Center point calculation

Now the center of each blob is calculated. Suppose, Center= Cen, then for each blob, CenColorD will be calculated. The color can be blue, red or green, and D will be x and y. Example,

For blue blob, the center will be as shown in fig 9.

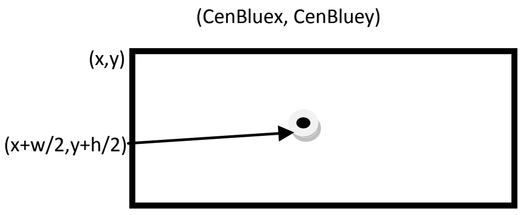

Fig 9. Center Point Calculation

3. Calculate the change in center.

Now when the marker is moved in front of the camera its corresponding blob will also move on the screen. So new co-ordinates of blob will be obtained and hence a new center will be there. There is a need to calculate this change in center as shown in fig 10. Now this new center will be made current center as well as global center.

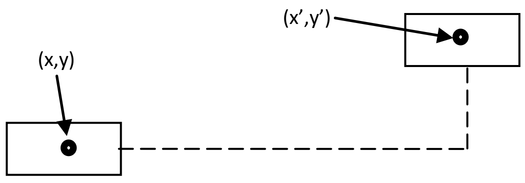

Fig 10. New Center Point Calculation

Some of the sample images are shown below:-

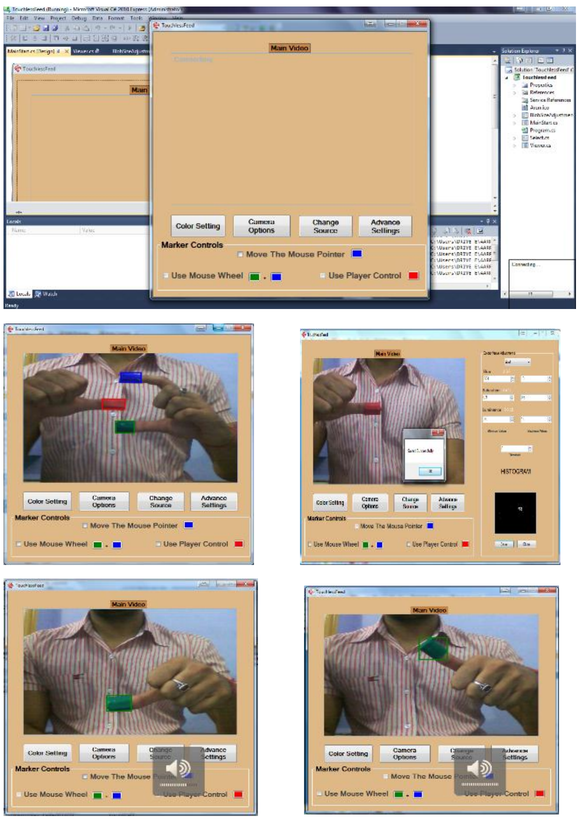

Fig 11. Sample Images

4. Find out how close green and blue blobs are.

The difference between the two blobs will serve the way for further calculations. The new center obtained and the change in center will be saved for the next frame.

```
5. Events
Example: Moving a mouse pointer
1) Is the position of center changed? If no then do nothing.
2) If changed, then check whether the mouse wheel is ON. If mouse
wheel is OFF then do nothing.
3) If mouse wheel is ON, and co-ordinates of blue greater than 1

      If change in center of blue between -1 and +1, then sensitivity
      factor is 0. So x'=x+Δx and y'=y+Δy.
      If change in center of blue between -3 and +3, then sensitivity
      factor is 2. So x'=x+2Δx and y'=y+2Δy.
      If change in center of blue between -8 and +8, then sensitivity
      factor is 3. So x'=x+3Δx and y'=y+3Δy.
      If change in center of blue between -∞ and +∞, then sensitivity
      factor is 8. So x'=x+8Δx and y'=y+8Δy.
4) If mouse wheel is ON,
     If red position is equal to zero.
          Calculate center difference between blue & green.
          If the range is between -10 to -2
               Zoom is not happening.
          If center change of green or blue is 0, do nothing
          else
               If center change of green is less than -8 & center
               change of blue is greater than +8 in y-direction
               then copy;
               else if center change of green is greater than +8 &
               center change of blue is less than -8 in y-direction
               then paste;
               else if center change of green is less than -1 &
               center change of blue is more than +1 in x-direction
               then zoom in;
               else if center change of green is greater than -1 &
               center change of blue is less than +1 in x-direction
               then zoom out;
```

.

3. CONCLUSION
In the ever shrinking world of Information Technology, this research is only a humble joint venture to satisfy a small part of the Motion Detection & Processing. The system is highly flexible and can be modified to detect motion & all of us thereby making the system user friendly. Security is one main consideration in the project. The system is protected from any unauthorized access. We hope the entire objection to the system is rectified and the users will accept the system. There is no claim of this product being perfect, or anything near that. This is only a humble attempt made under trying circumstances. This system has been designed in an attractive manner so that, even a user with minimum knowledge can operates the system easily.

4. FUTURE SCOPE
This paper focuses on the use of a personal computer from far without even touching it. The usability of the system is very high, and finds tremendous use if we were to manage several computers just by sitting at one place. The addition of new features like enlarging, shrinking, windows opening & closing, playing songs, watching images, zooming in & zooming out and many more functionalities enhances it. In the future we would like to enhance the speed and accuracy so as to further improve the user experience.

5. ACKNOWLEDGEMENT
I am thanking to my management for their continuing support and encouragement for completing this work and we are thanking to our faculty members of SMVDU for their valuable suggestions.

6. REFERENCES
[1] Lingchen Chen ; Feng Wang ; Hui Deng ; Kaifan Ji "A Survey on Hand Gesture Recognition", International Conference on Computer Sciences and Applications (CSA), IEEE, 2013

[2] Ul Haq, E. ; Pirzada, S.J.H. ; Baig, M.W. ; Hyunchul Shin "New hand gesture recognition method for mouse operations", 54th International Midwest Symposium on Circuits and Systems (MWSCAS), IEEE, 2011

[3] S.P. Kumar ; O. Pandithurai, "Sixth sense technology", International Conference on Information Communication and Embedded Systems (ICICES), IEEE, 2013

[4] K.Arai & R. Mardiyanto "Camera as Mouse and Keyboard for Handicap Person with Troubleshooting Ability, Recovery, and Complete Mouse Events" International Journal of Human Computer Interaction (IJHCI), Volume 1 Issue 3.Ding, W. and Marchionini, G. 1997 A Study on Video Browsing Strategies. Technical Report. University of Maryland at College Park.

[5] V. Pradeep "A Simple Algorithm for using Face as a Pointing Device using OpenCV " International Journal of Advanced Research in Computer Science and Software Engineering Volume 2, Issue 2, February 2012

[6] Robertson P, Laddaga R., Van Kleek M "Virtual mouse vision based interface".

[7] R.S. Pressman "Software Engineering, A practitioner's approach"

[8] O'Reilly Media , "Programming C#"

[9] Paul Beaker ,"Computer Graphics"

[10] AForge.NET open source framework - AForge.NET :: Framework- www.aforgenet.com/aforge/framework

[11] Various articles from www.wikipedia.org