# Human-Machine Interaction via Hand Gesture Recognition Using Web Camera

Ashok Kumar Yadav[1], Pooja Singh[2], Ramnaresh[3], Akshay Dixit[4], Mohd Maaz Ul Islam[5], Subhajeet Dey[6], Pawan Kumar[7]
[1,2,3,4,5,6,7]*Amity School of Engineering and Technology, New Delhi, India*

**Abstract- With increasing use of computer, there is a demand for friendly and easy interfaces to communicate with computer machine. Instead of uses mouse, joystick etc. which is uncomfortable for long time user, we need a better way of communicate to interact with machine. Using hand gesture recognition as a real time input is a better way of communication with computer and also is more natural and intuitive ways. Direct use of hands as an input device is an attractive technique which can communicate with more information and allowing a greater number of recognition system that can be used in a variety of human computer interaction applications. The human hand serves the dual purpose of a communication and a manipulation device. The present research work focuses on employing the hand as an interface device to a computer.**

**Keywords – Real time, gesture recognition, human computer interaction, tracking, hand gesture**

## I. INTRODUCTION

The use of computers in our everyday life is indispensable. Computers have become a key element of our society. From surfing the web, typing a document, playing a video game or storing and retrieving data are just a few of the instances involving the use of computers. With the price of computers depreciating constantly, they will even more influence our everyday life in the near future.

To improve the mannerisms with which we interact with the computers, Human- Computer Interaction (HCI) has been deemed as the most budding and lively field of research since past few years. To achieve natural and immersive human-computer interaction, the human hand could be used as an interface device [1].

Hand gestures are a powerful human to- human communication channel, which forms a major part of information transfer in our everyday life. Hand gestures are an easy to use and natural way of interaction. Using hands as a device can help people communicate with computers in a more intuitive and natural way. When we interact with other people, our hand movements play an important role and the information they convey is very rich in many ways. We use our hands for pointing at a person or at an object, conveying information about space, shape and temporal characteristics. We constantly use our hands to interact with objects: move them, modify them, and transform them. In the same unconscious way, we gesticulate while speaking to communicate ideas (’stop’, ’come closer’, ’no’, etc). Hand movements are thus a mean of non-verbal communication, ranging from simple actions (pointing at objects for example) to more complex ones (such as expressing feelings or communicating with others). In this sense, gestures are not only an ornament of spoken language, but are essential components of the language generation process itself [2].

This Direct use of the hand as an input device is an attractive method for providing natural human–machine interaction. To provide ease and naturalness to the user, we aim at recognizing hand gestures to interact with an application, thereby enhancing the user’s interaction with the computer. The sole purpose of this research is to provide the user with an option to handle everyday computing tasks by using simple hand gestures without the need of any hardware.

For taking the users’ input, we employ a standard web camera for monitoring the different gestures. The project allows the training of arbitrary gestures by developers which can then be recalled for interacting with the system like opening the Charms Utility, on any system running Microsoft Windows 8.

The application uses gesture recognition through webcam to use touch features of Windows 8. It uses webcam as primary input device. The application tracks the movement of the hand to trigger various event. Depending on the movement of user hand, various functions are triggered. On moving the hand from right to left, charms bar opens. Similarly on moving the hand from left to right app switch is opened. The application also takes the screenshot of the current screen on moving the hand from top to bottom. It also opens the desktop on moving the hand from bottom to top. The application is aimed to target all the windows 8 that does not have touch display. With this application, non-touch display users can use touch-enable features of Microsoft Windows 8.

## II. RELATED WORK

To improve the interaction in qualitative terms in dynamic environment, it is desirable to have a means of interaction as ordinary and natural as possible. Gestures, especially expressed by hands have become a popular means of human computer interface now days [4]. Human hand gestures may be defined as a set of permutation generated by actions of the hand and arm [5]. The movements may vary from simple action of pointing by finger to more composite ones that are used for communication among people. The use of fingers and palm for communication will help reduce the technological barricade of communication between informal computer using community and computer system as a whole. Thus the adoption of hand, particularly the palm and fingers as the means of input devices sufficiently lower the technological barrier in the interaction between the disinterested users and computer in the course of human computer interaction [3]. To accept hands as the input device, the applications need to rely on external devices that are able to capture the gestures and convert them into input. For this the usage of a video camera can be done that grabs user's gesture, along with that we require processing system that capture the useful features and partitions the behavior into appropriate classes.

Numerous applications related to gesture recognition have been designed for presenting, pointing, virtual workbenches, VR etc. Gesture input can be categorized into different categories depending on various characteristic [6]. One of the categories is deictic gesture that refers to reaching for something or pointing an object. Accepting or refusing an action for an event is termed as mimetic gestures. It is useful for language representation of gestures. An iconic gesture is way of defining an object or its features. Chai et al. [7] presents a hand gesture application in gallery browsing 3D depth data analysis method. It adds up the global structure information with the local texture variation in the gesture framework designed. Pavlovic et al. [5] have concluded in their paper that the gestures performed by users must be logically explainable for designing a good human computer interface. The current technologies for gesture recognition are not in a state of providing acceptable solutions to the problems stated above. One of the major challenges is evolution in the due course of time of the complexity and robustness associated with the analysis and evaluation for gestures recognition. Different researchers have proposed and implemented different pragmatic techniques for gesture as the input for human computer interfaces. Dias et al. [8] presents a free-hand gesture user interface which is based on finding the flight of fiduciary colour markers connected to the user's fingers.

The model used for the video presentation, is grounded in its disintegration in a sequence of frames or filmstrip. Liu and Lovell [9], proposed an interesting technique for real time tracking of hand capturing gestures through a web camera and Intel Pentium based personal computer. The proposed technique is implemented without any use of sophisticated image processing algorithms and hardware. Atia et al. [10] designs a tilting interface for remote and quick interactions for controlling the directions in an application in ubiquitous environment. It uses coin sized3D accelerometer sensor for manipulating the application. Controlling VLC media player using hand gesture recognition is done in real time environment using vision based techniques [11]. Xu et al. [12] used contact based devices like accelerometer and EMG sensors for controlling virtual games. Conci et al. [13] designs an interactive virtual blackboard by using video processing and gesture recognition engine for giving commands, writing and manipulating objects on a projected visual interface. Lee et al. [14] developed a Virtual Office Environment System (VOES), in which avatar is used navigate and interact with other participants. For controlling the avatar motion in the system a continuous hand gesture system is designed which uses state automata to segment continuous hand gesture and to remove meaningless motion. Xu et al. [15] presents a hand gesture recognition system for a virtual Rubik's Cube game control that is controlled by EMG and 3D accelerometer to provide a user-friendly interaction between human and computers. In this the signals segments the meaningful gestures from the stream of EMG signal inputs.

There are several studies on the hand movements especially gestures, by modelling the human body .On the basis of body of knowledge now it is possible to countenance the problem from a mathematical viewpoint [16]. The major drawbacks of such techniques are they are very complex and highly sophisticated for developing an actionable procedure to make the necessary jigs and tools for any typical application scenarios. This problem can be overcome by pattern recognition methods having lower hardware and computational overhead. These aspects have been considered in subsequent sections, by making the dynamic user interface for the validation of those concepts, where a user performs actions that generates an executable commands in an intelligent system to implement the user requirements in a natural way.

## III. PROPOSED FRAMEWORK

Hand Gesture Recognition Model uses motion detection as its first step, and then does some interesting routines with the detected object. Let's suppose we have a camera which monitors an area. When somebody gets into the area and makes some hands gestures in front of the camera, the application should detect the type of the gesture, and

raise an event, for example. When a hands gesture is detected, the application could perform different actions depending on the type of the gesture. For example, a gesture recognition application could control some sort of device, or another application sending different commands to it depending on the recognized gesture.

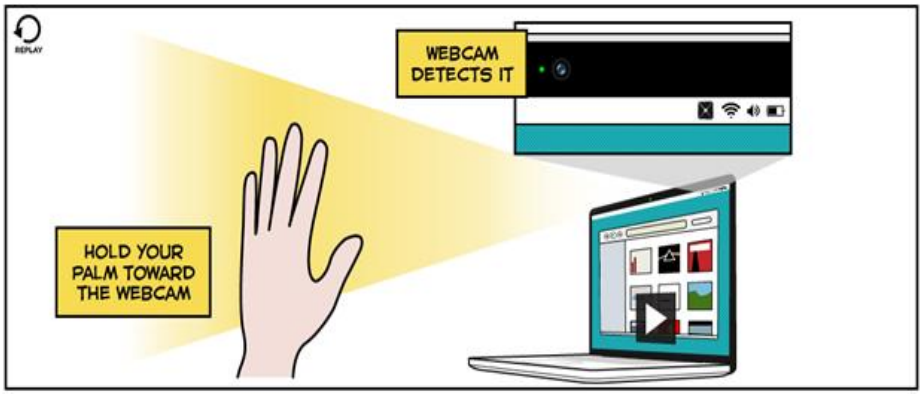


Fig. 1. Full System Diagram

We wish to recognize gestures on a complex background such as the living room which can be surrounded by moving persons and objects in the background. The webcam is either embedded in the screen of the computer or placed on top with the user facing it. Our method is experimented on 640x480 videos acquired with different webcams. Our goal is to, process at 25 fps and perform real time processing using a standard personal computer. The programming was done in C/C++ using the OpenCV library [17].

Technique for Hand Segmentation

We want to recognize specific gestures. To do that we process frames to locate the hand and its gesture.

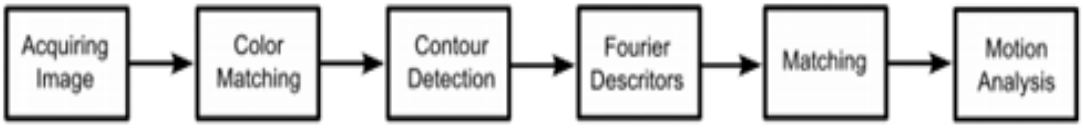


Fig. 2. Motion Analysis Algorithm

*3.1 Colour Based Algorithm*

Colour based segmentation is the most commonly used technique and is simple to use and computational efficient. Our method extracts pixels of the skin color and attempts to recognize a hand gesture from the contours.

The colour based segmentation relies on the identification of skin coloured pixels in a captured video frame. These pixels are grouped by regions and extracted as Blobs. These blobs are then processed to identify if they represent the hand.

The identification of skin coloured pixel relies on the sampling of one's skin colour. To extract the skin colour attributes, either loose assumptions can be made in RGB colour space or sampling can be done. Since loose assumptions may not adapt to different lighting, we chose a sampling method. This sampling can be done by different ways. The easiest way is to use another segmentation method to get a picture of the hand and sample pixel values from it.

Another efficient method is to extract colour from the user's skin. This will allow different colour skin types to be detected. As provided an efficient way of detecting a face, we can first locate the head and then retrieve the skin colour from the user's cheeks.

The RGB colour format does not provide good samples since it is well known that the RGB colour format does not match well to the human visual system. The YCrCb

Colour format is known for matching the perception of colour and is therefore used in video coding.

In the HSV (define) colour space, the hue represents the colour, the gray scale saturation, and the value the luminance component. Therefore, only one parameter is used to characterize a colour. This allows to direct manipulation of the hue component of a video frame. The drawback is that the hue component of wood colours is close to the hue component of skin colour, which affects the reliability of this method. To sample the skin colour, we

adopt an histogram representation of the colour. The height of a histogram column represents the proportion of pixel having this hue in the skin colour sample.

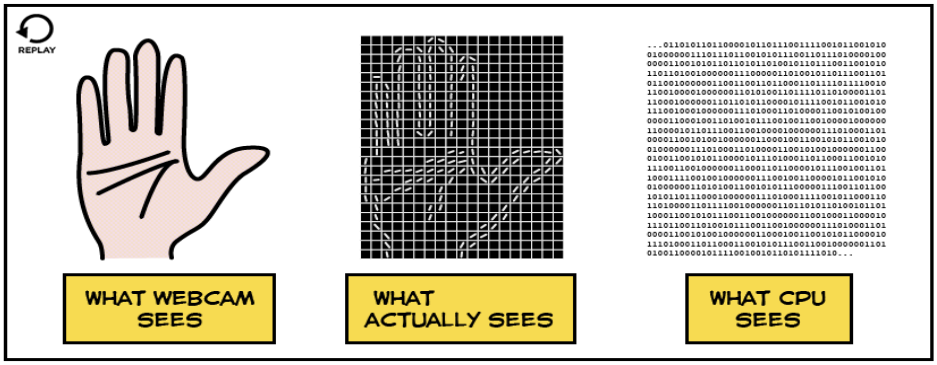


Fig. 1. Conversion of Hand Image into Binary data

The head detection method to sample the skin colour where the pink square is the result of the head detection and the blue squares represent the regions from where the skin colour was extracted, we want to detect the parts of the video frame which belong to the user skin. To do that, we use a method called back projection. This operation affects the histogram value of each pixel. Therefore the brighter a pixel becomes after back projection, the higher chances are that it belongs to a skin region.

*3.2 Hand Recognition*

We retrieve contours from the binary mask obtained and discard very small contours to avoid false positives. We only take the upper part of the binary object so that exposed arm will not interfere. We set a maximum ratio for the height of the object on the width of the object to 1.1, which corresponds to the normal ratio for the palm of a hand. We first distribute the contour's points at regular intervals along the contour, and then we transform the resulting vector of points in the frequency domain using Fourier's Descriptors. Fourier Descriptors provide a simple way to obtain a translation invariant and scale invariant contour. Eq. (1) shows the computation of the Fourier's descriptors.

$$FD(\omega) = \sum_{k=0}^{N-1} (Pt_k.x + j \times Pt_k.y)\, e^{j2\pi\left(\frac{\omega k}{N}\right)}$$

Where Ptk is the kth point of the contour, N is the number of points in the contour and is the frequency. We set the first coefficient of Fourier's Descriptor to zero to have a translation invariant pattern and we divide each coefficient by the second coefficient so that our pattern becomes scale invariant. We apply a low pass filter and calculate the distance between the contour and a reference and then perform the inverse transform. We use Eq. (2) to compute the distance between the reference and a contour retrieved from the image.

$$Dist = \sum_{k=0}^{N-1} \sqrt{(Pt_k.x - Ref_k.x)^2 + (Pt_k.y - Ref_k.y)^2}$$

where Refk is the kth point of the reference contour. Fig.15(a) presents an example of reference contour and Fig.15(b) presents an actual detected contour matched with the reference. The OpenCV distribution comes with a trained frontal face detector which we used to sample the skin color from the cheeks. The algorithm is designed to work well for rigid objects and characteristic views, which is our case. The user is standing in front of the TV. If we consider one gesture at a time, for a specific gesture, the fingers can be static and the hand considered as a rigid object.

*3.3 Supervised Learning*

The training requires a large database of positive and negative frames. Positive frames are frames containing a hand to be recognized while negative frames are frames without any hand reference. The steps to train the Haar cascade of features include manually segmenting the positive samples, converting samples to a uniform set and starting the learning process. To create our positive data set, we used 4 different webcams. Part of the image segmentation was done automatically using the previous methods and the restwas done manually to avoid a bias due to the use of one of our method. The negative samples were taken from the Haar training google code tutorial . The method requires a number of sample to be in the order of thousands. We used 952 positive samples and 1602 negative samples to train a Haar cascade to recognize a palm gesture.

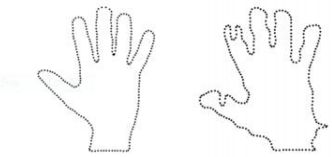

Fig. 4. Hand Contour Using Motion Analysis

Fig.16(a) presents an example of a positive sample and Fig.16(b) presents an example of a negative sample. This method computes the difference between two successive video frames, which is usually stable enough to enable hand recognition using to a reference. It has the advantage of being computationally efficient for a fewer references we first compute the difference between the image and the previous one, and convert it to grayscale. The grayscale map is obtained by adding the contribution of each channel of the RGB original frame to the difference image. Then, we established a distance map between a reference and the motion observed with the webcam. The map is obtained by differencing the grayscale image and the reference throughout the grayscale image. The Sum of Absolute Difference (SAD) described in Eq. (3) is used to create the correlation.

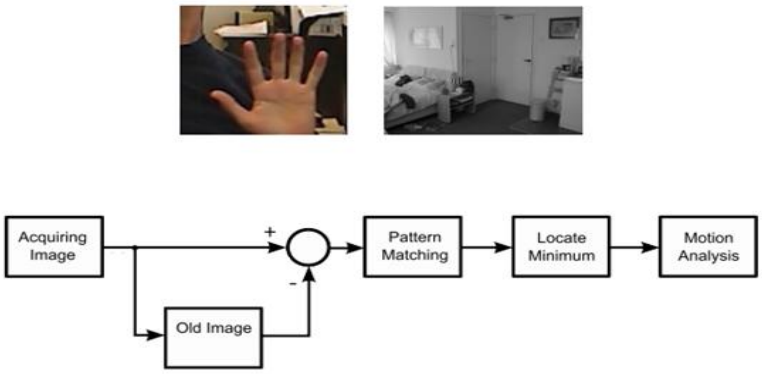


Fig. 5.Motion Analysis OpenCV

*3.4 Down Sampling*

The major drawback of this method is inherent in all pattern matching methods: the processing time is very long for one pattern, but increases proportionally to the number of patterns to match, including scaled references. This means that we can only process either a few references with a constant scale or use a wider scale with only one reference. Both reference and motion maps obtained from the webcam are down sampled to reduce the processing time.

Fig. 6.Downsampled Image

| | Downsampling range | | | | | |
|---|---|---|---|---|---|---|
| Amount of scaling | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 47.1ms | 11.6ms | 5.41ms | 2.89ms | 2.02ms | 1.31ms |
| 2 | 95.3ms | 22.9ms | 11.5ms | 6.56ms | 3.53ms | 2.61ms |
| 4 | 192ms | 46.6ms | 22.2ms | 13.1ms | 8.30ms | 5.83ms |
| 6 | 289ms | 72.6ms | 32.2ms | 20.0ms | 12.6ms | 8.80ms |
| 8 | 394ms | 99.0ms | 44.1ms | 23.5ms | 16.8ms | 12.6ms |

Fig. 7. Down sampling Range table

The template presented was sampled manually from a difference frame of a user demonstrating a hand gesture. This approach gives correct results; however it can be greatly improved by modifying the templates. The method used to set the threshold is sensitive to the number of white pixels in a reference. If a reference counts many fingers shown, it will have more edges and therefore the DistRef will be higher, making it very sensitive to noise compared to the other since its threshold will be significantly higher. Therefore we need to level all threshold values. A simple way to do that is to add grey pixels to references. Besides, the inside of the hand contains unstable white zones due to the lack of strong edges in this moving part. A very simple way to discard this information is to fill the inside of the hand references with grey pixels. Since we use a difference between the reference and the image, any value of the image bitmap will give the same value for the inside of the hand, which suppress the problem of unstable white zones. In addition the thresholds will be closer for different references making them less vulnerable to noise. SAD Calculation Eq. (3) is used to compute the distance between the frame and the reference. It can be seen as a window of the same size as the reference used to extract a region of interest (ROI) shifted by one pixel until the whole frame is covered. For each position the difference between the ROI and the reference is computed. Computing this directly would incur a long processing time.

To optimize the processing time, we need to do the least operation per frame and per template in the processing loop. Therefore we compute FFT and RRef for each frame final difference map is computed for each reference.

## IV. PROPOSED METHOD

The following research follows the following method to attain the desired result.

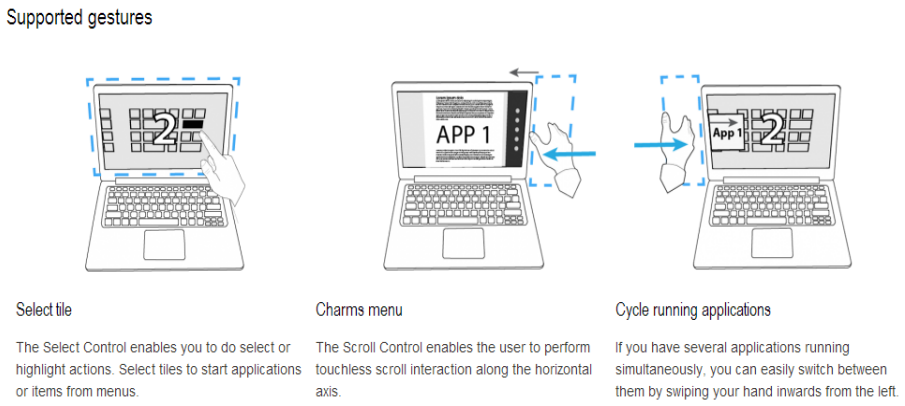


Fig. 8. Proposed gestures

*4.1 General Skin and Chrominance Model*

As in many natural processes, random variations of skin chrominance tend to cluster around a mean in the chrominance space. This is the most commonly observed probability distribution case and is called Gaussian (normal) distribution. H-S , g-r, and Cr-Cb components construct chrominance in each color space and it is expected that each distribution of their skin color values would yield a Gaussian distribution. To extract mathematically useful information, the histograms of the chrominance components should be like in the following form

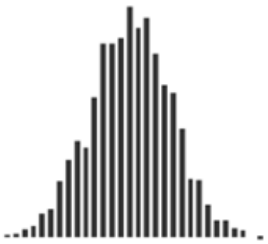

Fig. 9. histograms

If such a distribution is observed in chrominance data, the mathematical representation would yield the following well known probability density function.

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}$$

If the two of the chrominance components (e.g. Hue and Saturation) demonstrate such a distribution shown in Fig.19, skin chrominance distribution of the skin pixels can be modelled by a Gaussian joint probability distribution given by

$$p[\mathbf{x}(i, j/W_s) = (2\pi)^{-1}|\mathbf{C}_s|^{-1/2} \exp[-\lambda_s{}^2(i, j)/2]$$

where the vector x(i; j) = [ x(i; j) y(i; j)] T corresponds to values of the chrominance (x; y) (can be thought as H and S or g and r components) of a pixel with coordinates (i; j), Ws is the distribution representing skin color, Cs is the covariance matrix for skin chrominance, and s (i; j) is the Mahalanobis distance from the vector x(i; j) to the mean vector ms = [mxs mys]T obtained for skin chrominance.

$$[\lambda_s(i, j]^2 = [\ \underline{\mathbf{x}}(i, j) - m_s\ ]^T\ \mathbf{C}_s^{-1}\ [\ \underline{\mathbf{x}}(i, j) - m_s\ ]$$

The equation defines elliptical surfaces in chrominance space of s (i; j) centered about ms and where Cs determines the principal axes. Equation 6.1 presents the probability of a pixel with coordinates (i; j) belongs to the class Ws . Correspondingly, if s (i; j) (Mahalanobis distance) of a pixel increases, probability of that pixel belonging to class Ws decreases.

By the above theoretical background information, one can model skin locus in a color space by estimating ms and Cs. In colour space calculations ms is constructed by the mean values of the chrominance components (e.g. Cb andCr) which were extracted from the recorded skin sample pixels. Then the mean vector of a color space is defined as,

$$m_s = [\mu_x\ \ \mu_y]$$

and mean values of x and y components can be estimated directly by the values of recorded skin sample pixels using the following equations:

$$\mu_x = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\mu_y = \frac{1}{n}\sum_{i=1}^{n} y_i$$

where n is the total number of pixels used in the experiment and xi and yi values are the chrominance components' values of the ith pixel respectively in defined colour space. For the estimation of the covariance matrix Cs, let us start with the definition of variance in discrete one dimensional case,

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n} (x_i - \mu_x)^2$$

Finally, cross covariance of the chrominance components and variance of each chrominance component need to be estimated. The estimations in the discrete case are done by the following formulas. And cross covariance of the chrominance components Cxy can be estimated by the following formulas:

$$C_{xy} = \sum_{i=1}^{n} [x_i - \frac{1}{n}\sum_{j=1}^{n} x_j][y_i - \frac{1}{n}\sum_{j=1}^{n} y_j]$$

$$\sigma_x^2 = \frac{1}{n}\sum_{i=1}^{n} (x_i - \mu_x)^2$$

$$\sigma_y^2 = \frac{1}{n}\sum_{i=1}^{n} (y_i - \mu_y)^2$$

By considering the above equations, one can model distribution of skin colour in a defined colour space. Here the aim is calculating the mean and variance of CrossCovariance values of the skin chrominance components and define an ellipse that modelling (circulating) the skin locus. An instance of a distribution with its covariance matrix CS and its locus is illustrated in Fig.20.

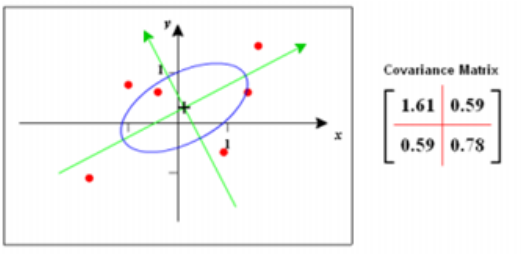


Fig. 10. Hidden Markov Model

By using chrominance model we can determine the hand color and apply a trajectory detection algorithm to trace the path of chrominance matrix, Trajectory analysis can be performed by many ways, either by simple matching such as features points or direction. More complex modeling theories have emerged to recognize complex gesture trajectories using Hidden Markov Models (HMM). HMM have the advantage that the states can be mapped directly with part of the hand trajectory. However these methods require a lot of trajectory samples for training and therefore are difficult to adapt to user-specific gestures and create extra steps for simple applications such as below:

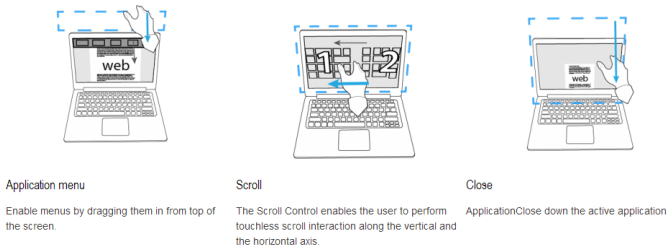


Fig. 11.

*4.2 Depth Map*

The OpenCV library provides a framework to calibrate single and stereo camera. However, we used the Minoru 3D webcam which is available commercially at low cost and does not require the extra calibration step.

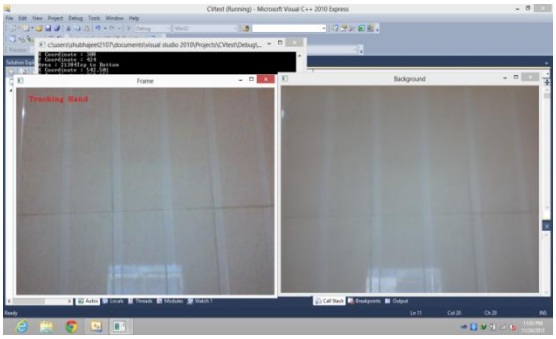


Fig. 12. Bakcground Sampling

Nevertheless, we tested the accuracy of the calibration. By calibrating each webcam view using a chessboard, the high quality of the lenses do not create any radial or tangential distortion. The stereo calibration using several pairs of images of the chessboard also lead to the conclusion that the webcam does not require these extra steps. The Hartley's algorithm allows us to conclude that the webcam are correctly aligned and therefore does not require calibration.

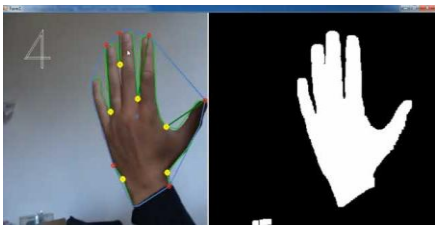

Fig. 12. Contour Defects

A stereo webcam does not provide a 3D model of the scene but rather a 2D view with a depth information. This depth information is computed by observing the correlation between the two frames extracted from each webcam. Similarities are found in the two frames as part of a similar object. From the difference of viewing angle of the same object between the left and the right frame, the depth of the object can be estimated. From this depth map, we can estimate the relative distance of different objects from the camera. A matching algorithm to use may include key points matching and macro block matching. We choose to use the macroblock matching method because we wish to

extract the area of the object. Hirschmuller has developed a very robust algorithm for stereo block matching which results in a high-precision depth map. However, this algorithm cannot run in real time on a computer. Hence, we chose the simpler block matching algorithm. The stereo matching algorithm represent the bottleneck of the stereo processing, using the block matching method, we obtain a frame rate of 16.2 fps which is equivalent to a processing time of 61.7ms per frame.

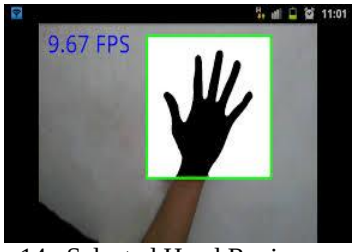


Fig. 14. Selected Hand Region

*4.3 Simple Sampling Method*

We propose a simple application for the stereo webcam, extracting a hand reference to calibrate other methods requiring some training. This method aims at extracting a hand sample from the user prior to any knowledge concerning the hand gesture, the hand posture or the hand color. From the two images generated by the webcam, a stereo matching algorithm is used to extract a depth map of the viewed scene. As we want to extract a hand reference, a simple way to segment it is to have the user reach out his hand in front of the camera. In this configuration, the hand is located in the first plane of the webcam image, the user in the second plane, and the background in the next plane. To segment the hand from this configuration, we employ the Otsu's thresholding method, which allows the depth map to be divided in two regions: the first plane where the hand is located and the other planes.

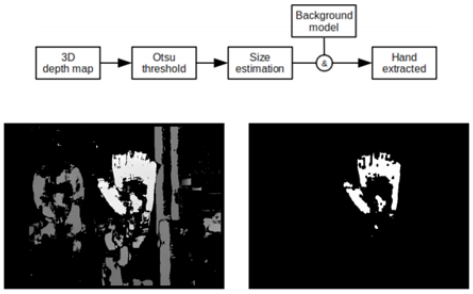


Fig. 15.Background elimination algorithm

It can be noted that the image includes some noise at the bottom of the hand but overall the hand is correctly segmented. Fig.25(c) illustrates the superposition of the two previous images (i.e., the extracted and segmented hand). From the segment hand, several parameters can be extracted such as the colour distribution, the hand shape, the hand gesture or the ambient lighting.

The background modelling allows segmenting correctly the hand most of the time. How-ever, if the user appears in the segmented area, it will affect the segmentation accuracy. Indeed the user is not part of the background, therefore if a part of his body, for example his head, is aligned with his hand and the webcam, it will appear as part of the area selected. After segmentation, both the head of the user and the hand will appear in the extracted image and needs to be separated.

Our method offers an original way to automatically detect and segment a hand. It does not rely on any assumption about the hand characteristics (e.g., size, skin color) and can be applied in any configuration. It can be used as a calibration step for other hand gesture recognition methods or to create a hand database for machine learning purposes.

| $H \leftrightarrow U$ \ $U \leftrightarrow W$ | 0.5m | 1.0m | 1.5m | 2.0m | 2.5m | 3.0m |
|---|---|---|---|---|---|---|
| 10cm | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| 20cm | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| 30cm | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| 40cm | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ |
| 50cm | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ |

Fig. 16. Camera Focus Range Table

## V. CONCLUSION AND FUTURE SCOPE

This paper enables us to implement the touch features of Windows 8 without having a touch display for a Windows 8 system. This is done by the means of a webcam, which has become a significant accessory in every computer system. The application will help overcome a technological barrier between informal computer using community and the computer system.

We aim to extend the above functionality to different Operating Systems available

## VI. REFERENCES


[1] Conci, Nicola, Paolo Ceresato, and Francesco GB De Natale. "Natural human-machine interface using an interactive virtual blackboard." In 2007 IEEE International Conference on Image Processing, vol. 5, pp. V-181. IEEE, 2007.

[2] Rautaray, Siddharth S., and Anupam Agrawal. "Real time multiple hand gesture recognition system for human computer interaction." International Journal of Intelligent Systems and Applications 4, no. 5 (2012): 56-64.

[3] Vardy, Andrew, John Robinson, and Li-Te Cheng. "The wristcam as input device." In Digest of Papers. Third International Symposium on Wearable Computers, pp. 199-202. IEEE, 1999.

[4] Man, Wong Tai, Sun Man Qiu, and Wong Kin Hong. "Thumbstick: a novel virtual hand gesture interface." In ROMAN 2005. IEEE International Workshop on Robot and Human Interactive Communication, 2005., pp. 300-305. IEEE, 2005.

[5] Freeman, William T., David B. Anderson, P. Beardsley, Chris N. Dodge, Michal Roth, Craig D. Weissman, William S. Yerazunis et al. "Computer vision for interactive computer graphics." IEEE Computer Graphics and Applications 18, no. 3 (1998): 42-53.

[6] Soontranon, N., Supavadee Aramvith, and Thanarat H. Chalidabhongse. "Improved face and hand tracking for sign language recognition." In International Conference on Information Technology: Coding and Computing (ITCC'05)-Volume II, vol. 2, pp. 141-146. IEEE, 2005.

[7] Pavlovic, Vladimir I., Rajeev Sharma, and Thomas S. Huang. "Visual interpretation of hand gestures for human-computer interaction: A review." IEEE Transactions on Pattern Analysis & Machine Intelligence 7 (1997): 677-695.

[8] Chai, Xiujuan, Yikai Fang, and Kongqiao Wang. "Robust hand gesture analysis and application in gallery browsing." In 2009 IEEE International Conference on Multimedia and Expo, pp. 938-941. IEEE, 2009.

[9] Dias, José Miguel Salles, Pedro Nande, Pedro Santos, Nuno Barata, and André Correia. "Image Manipulation through Gestures." Proceedings of AICG 4 (2004): 1-8.

[10] Atia, Ayman. "Interaction with tilting gestures in ubiquitous environments." arXiv preprint arXiv:1007.5158 (2010).

[11] Rautaray, Siddharth Swarup, and Anupam Agrawal. "A novel human computer interface based on hand gesture recognition using computer vision techniques." In Proceedings of the First International Conference on Intelligent Interactive Technologies and Multimedia, pp. 292-296. ACM, 2010.

[12] Zhang, Xu, Xiang Chen, Wen-hui Wang, Ji-hai Yang, Vuokko Lantz, and Kong-qiao Wang. "Hand gesture recognition and virtual game control based on 3D accelerometer and EMG sensors." In Proceedings of the 14th international conference on Intelligent user interfaces, pp. 401-406. ACM, 2009.

[13] Lee, ChanSu, SangWon Ghyme, and ChanJong Park. "The control of avatar motion using hand gesture." In In Proceedings of ACM VRST'98. 1998.

[14] Zhang, Xu, Xiang Chen, Yun Li, Vuokko Lantz, Kongqiao Wang, and Jihai Yang. "A framework for hand gesture recognition based on accelerometer and EMG sensors." IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans 41, no. 6 (2011): 1064-1076.

[15] Conci, Nicola, Paolo Ceresato, and Francesco GB De Natale. "Natural human-machine interface using an interactive virtual blackboard." In 2007 IEEE International Conference on Image Processing, vol. 5, pp. V-181. IEEE, 2007.

[16] Yi, Beifang, Frederick C. Harris Jr, Ling Wang, and Yusong Yan. "Real-time natural hand gestures." Computing in Science & Engineering 7, no. 3 (2005): 92.

[17] Bernard, Arnaud, and Benny Bing. "Hand gesture video browsing for broadband-enabled HDTVs." In 2010 IEEE Sarnoff Symposium, pp. 1-5. IEEE, 2010.