1. **INTRODUCTION**

Human Computer Interaction (HCI) is getting more and more important with the improvement of the technology. In the history, different perspectives like static keys and buttons, track path devices, touch and multi-touch screens have developed. The next innovation should let people use computers easily and effortlessly without touching. Thus, gesture recognition and motion tracking are the next improvement Human Computer Interaction.

The Gesturised computer operation makes a way for the user to operate the computer from a distance and also ends the era of keyboard and touch. The Gesture Recognition System will be the solution of getting out of an obligation of touching screen or pushing button that enforces the user to focus on the device when the user is doing another job. Gesture recognition is the process of recognizing the captured gesture and comparing with the assigned gesture and perform particular task assigned. Gesturised computer operation is assigning the task to a particular gesture recognised and enabling the operation of computer.

The objectives of this project are to detect and analyse the gestures and ease the interface between the user and the computer by gesture detection rather than click, press and touch. Gesture Recognition software will be designed as a whole simulation of a Computer. Since Computers are one of the most indispensable objects for people in their daily life, our software has to be designed with minimum faults. Implementation of the stereovision technology by using the integrated camera provides an inexpensive solution with respect to the current technologies. To achieve portability, integrated camera is used and the software will be implemented into the Computers which have Windows based operating systems.

In this report controlling the operations via gesture is explained. Detailed design concepts will also be shown. The required UML diagrams will be added in order that the detailed design description document will be more understandable with the necessity visuals.

**1.1 OBJECTIVE**

The main objective of our project is to detect and analyse the gestures and ease the interface between the user and the computer by gesture detection rather than click, press and touch and also help the handicap people to use the system efficiently. To achieve Portability, integrated camera is used and the software will be implemented into the Computers which have Windows based operating systems.

**1.2 PROBLEM STATEMENT**

To design and develop gesture detection software for operating computer using webcam which can even be used by handicapped people.

Operating the computer is not an easy task when the device is located away from user. Touching screen or pushing button enforces the user to focus on the device even when the user is doing another job. However, the Gesture Recognition System will be the solution of getting out of an obligation of touching the screen or pushing the buttons of the device in order to control the menu, open folders, play music, take a picture etc. By detecting the hand movements of a user, there will be no necessity of touching the computer devices. Since handicaps are unable to operate the computer as it requires the click, press and touch interface so far, by our software we bring forth the interface of gestures with the computers.

Some applications related with gesture recognition using sensors are already available for computers but they are quite expensive. To design and develop gesture detection software for operating computer using webcam which can even be used by handicapped people provides and inexpensive solution with respect to the current technologies.

**1.3 MOTIVATION**

With the increasing usage of computer by the people, the requirement of speed and efficiency plays a vital role. Through our software of gesture detection we make the operation of the computer faster and easier in day to day life. It also renders the most valuable interface of the computer and handicaps. As Apple has implemented mouse controlling and navigation using iNavigate software via gesture recognition and it is very much environmental dependent and cannot be used in any other platforms we are motivated to build a software which is more efficient and compatible when compared to iNavigate.

Microsoft has a product called Kinect which uses sensors and high cost camera to recognize the gestures, we are building a software which uses only integrated camera which makes it cost effective.

**1.4 SOME DEFINITIONS**

1.4.1 Finger detection:

Finger detection is a computer technology that detects fingers and palms assuming a white background.

1.4.2 Face detection:

Face detection is a computer technology that determines the locations and sizes of human faces in arbitrary (digital) images. It detects facial features and ignores anything else, such as buildings, trees and bodies.

1.4.3 Color detection:

Detecting the color by specifying the range of it.

1.4.4 Haar-Cascade:

Haar-cascade is an object detection algorithm used to locate faces, pedestrians,objects and facial expressions in an image. In Haar -cascade, the system is provided with several numbers of positive images ( like faces of different people at different background and negative images (images that are not faces but can be anything else like wall, chair, table, etc.

1. **LITERATURE SURVEY**

**2.1- Paper 1: IEEE paper**

Depth camera based hand gesture recognition and its applications in Human-Computer-Interaction

Authors : Zhou Ren, Jingjing Meng, Junsong Yuan

Name of conference:

Date of conference :13-16 Dec. 2011

Hand gesture is a technique used for Human-Computer-Interaction. It is applied in exploring large and complex data, computer games, virtual reality, health care, etc. Of various Human-Computer-Interactions (HCI), hand gesture based HCI might be the most natural and intuitive way to communicate between people and machines, since it closely mimics how human interact with each other. Its intuitiveness and naturalness have spawned many applications in exploring large and complex data, computer games, virtual reality, health care, etc. Although the market for hand gesture based HCI is huge, building a robust hand gesture recognition system remains a challenging problem for traditional vision-based approaches, which are greatly limited by the quality of the input from optical sensors. In this paper, the performance is compared in terms of speed and accuracy between FEMD (Finger-Earth Mover’s distance and traditional corresponding- based shape matching algorithm, Shape Context. And then several HCI applications built on top of an accurate and robust hand gesture recognition system based on FEMD has been introduced. FEMD metric is speciﬁcally designed for hand shapes. It considers each ﬁnger as a cluster and penalizes unmatched ﬁngers. In this paper, the FEMD based hand gesture recognition system is compared with traditional corresponding based matching algorithm, Shape Context. And several HCI applications on top of this novel hand gesture recognition system is built and its potential in other real-life HCI applications is demonstrated. This hand gesture recognition system performs robustly despite variations in hand orientation, scale or articulation. Moreover, it works well in uncontrolled environments with background clusters. The robust hand gesture recognition system can be a key enabler for numerous hand gesture based HCI systems is demonstrated. Currently, the most eﬀective tools for capturing hand gesture are electro-mechanical or magnetic sensing devices (data gloves). These methods employ sensors attached to a glove that transduces ﬁnger ﬂexions into electrical signals to determine the hand gesture. They deliver the most complete, application-independent set of real-time measurements of the hand in HCI. However, they have several drawbacks They are very expensive for casual use, they hinder the naturalness of hand gesture, and they require complex calibration and setup procedures to obtain precise measurements.

**2.2- Paper 2**: Hand gesture recognition using a real-time tracking method and Hidden Markov model

Author: Feng-Sheng Chen, Chih-Ming Fu, Chung-Lin Huang

Date of acceptance: 20 March 2003

Hand gesture recognition research has gained a lot of attentions because of its applications for interactive human-machine interface and virtual environments. Most of the recent works related to hand gesture interface techniques has been categorized as: glove-based method and vision-based method. Glove-based gesture interfaces require the user to wear a cumbersome device, and generally carry a load of cables that connect the device to a computer. There are many vision-based techniques, such as model-base and state-based which have been proposed for locating objects and recognizing gesturers. Recently, there have been an increasing number of gesture recognition researches using vision-based methods.

This paper introduces a hand gesture recognition system to recognize ‘dynamic gesture’ of which a gesture in performed singly in complex background. Different from previous HMM (Hidden Markov Model)-based gesture recognition systems, this system do not use instrumented glove, nor any markers, but use 2D video input. The system tracks the hand and analyzes the hand-shape variation and motion information as the input to the HMM-based recognition system. The system consists of three modules: a real-time hand tracking, feature extraction, HMM training, and HMM-based gesture recognition. First, a real time hand gesture tracking technique tracks the moving hand and then extracts the hand shape from complex background. It is a simple and reliable method developed as a real-time image processing subsystem which consists of five basic complementary image processes: motion detection, skin color extraction, edge detection, movement justification, and background subtraction. FD (Fourier descriptor) is applied to characterize the spatial information and the optical flow method for motion analysis to characterize the temporal information. FD and motion information of the input image sequence is combined as feature vector. With these extracted feature vectors, the system is trained using HMM approach which is used to recognize the input gesture. In training phase, HMM is applied to describe the gestures in term of model parameter for each different gesture. The gesture to be recognized is separately scored against different HMMs. The model with the highest score is selected as the recognized gesture. The system consists of 20 different HMMs which are used to test20 different hand gestures. The experimental results show that the average recognition rate is above 90%.

**2.3- White Paper By Apple Labs- “iNavigate”, 2011**

iNavigate takes advantage of face detection technology as a input and a well-designed control mode and allows users to take charge of the cursor on the screen. Users can move the cursor to the position they want to go and perform left click, right click, double click and drag operation.

The project is held by Hongyi Zhao and Liyi Zhao from Apple Lab of Tongji University.

**2.4- Paper 4: Robust Real-Time Face Detection**

Aurhors: Paul Viola, Microsoft Reasearch,One Microsoft Way, Redmond,WA 98052, USA

Michael J.Jones,Mitsubishi Electric research Laboratory,201 Broadway, Cambridge, MA 02139, USA

Date of acceptance: July 11, 2003

This paper describes a face detection framework that is capable of processing images extremely rapidly while achieving high detection rates. There are three key contributions. The first is the introduction of a new image representation called the “ Integral Image” which allows the features used by our detector to be computed very quickly. The second is a simple and efficient classifier which is built using the AdaBoost learning algorithm to select a small number of critical visual features from a very large set of potential features. The third contribution is a method for combining classifiers in a “Cascade” which allows background regions of the image to be quickly discarded while spending more computation on promising face-like regions. A set of experiments in the domain of the face detection is presented. The system yields face detection performance comparable to the best previous systems. Implemented on a conventional desktop, face detection proceeds at frames per second.

**2.5 – Paper 5: A model-based hand gesture recognition system**

Authors: Chung-Lin Huang, Sheng-Hung Jeng

Date of acceptance: 4 September 2000

This paper introduces a model-based hand gesture recognition system, which consists of three phases: feature extraction, training, and recognition. In the feature extraction phase, a hybrid technique combines the spatial (edge) and the temporal (motion) information of each frame to extract the feature images. Then, in the training phase, the principal component analysis (PCA) is used to characterize spatial shape variations and the hidden Markov models (HMM) to describe the temporal shape variations. A modified Hausdorff distance measurement is also applied to measure the similarity between the feature images and the pre-stored PCA models. The similarity measures are referred to as the possible observations for each frame. Finally, in recognition phase, with the pre-trained PCA models and HMM, the observation patterns is generated from the input sequences, and then the Viterbi algorithm is applied to identify the gesture. In the experiments, the method proved that it can recognize 18 different continuous gestures effectively.

**3. HARDWARE-SOFTWARE SPECIFICATION**

**3.1 Development Environment**

3.1.1 Hardware

Processor Pentium IV

RAM 2 GB

Mother Board 1.2 GHz

Graphic Card 512 MB

Peripherals Webcam

3.1.2 Software

Operating System Windows XP Professional

IDE Visual Studio 2010

Programming Language Open CV

Packages Open CV 2.4.3 for Windows, EmguCV

**3.2 Requirement Analysis**

Primary requirement for the project is to operate a computer by recognizing gestures like hand, head and color. Hand gesture recognition is achieved by capturing photos of the finger movements and count of fingers. Head gesture is recognized by adopting face recognition methods and eye blink. Computer operation is also achieved by certain color recognition. Requirement specifications are as given below:

3.2.1 Functional Requirements

1. Image acquisition using built in camera.
2. Gesture detection by analyzing the images obtained.
3. Gesture recognition for identification of the gestures and exact gesture for mouse movement. This function is done for hand, head and color.
4. Action based on recognized gesture for operating the computer.
5. Providing the GUI –webcam interface for face detection, mouse control color detection and output display.

3.2.2 Non-Functional Requirements

1. Ease of operation
2. Cost
3. Fast response

Assumption: A single user is operating the system.

1. **DESIGN DETAILS**

**4.1 Initial Design**

A gesture-based interface, is designed for gesture recognition of the user. The recognised gestures are used to generate motion control commands so that it can control the operation of the computer according to the user's intention. Face detection algorithm and object tracking algorithm are combined in our system to achieve accurate face detection, tracking and gesture recognition in real time. It is intended to be used as a human-friendly interface for normal and disabled people to operate our computer using their gestures rather than click, press and touch.

**4.2 Design assumptions and dependencies**

Firstly, this project comes up with an idea to be used in computers which have integrated camera located in front. Obviously, this kind of technology is not available today therefore we are going to implement this project in personal computers with integrated camera.

It is needed that the performance and time limits of the software will be applied considering that the software would be working on a computer.

It is assumed that this project will operate well on new generation computers without any problems if the computers have powerful processors and integrated camera located in front.

**4.3 Design Goals and Guidelines**

The highest priority of the software is to design the Gesture Recognition module. After the module is designed, Gesture Recognition module will be inherited. Thus, design of Gesture Recognition module is the first main step of the software.

The other important goal is the speed of the software. The process of the software, which is recognition of the gesture, processing it and providing the action, will be done in at most few seconds in order that the software will work in reasonable time.

Fig 4.1

In the above figure we show the block diagram for Gesture Detection and Recognition. Initially we capture the image using the Webcam and obtain the image sequences. Image sequences will be the input to the gesture detection algorithm and features will be extracted which are later classified into mouse click and mouse motion events based on the pre-assigned gestures. Depending on the recognized gesture particular action will be performed.

Fig 4.2

Fig 4.2 shows the use case diagram for gesture detection and recognition. Here, Gesture Recognition gets the frames from camera and creates proper gestures if the user makes valid hand movements. In other case, that is user makes meaningless hand movements which does not exist in hand gesture data set, no gesture is created.

Fig 4.3

Fig 4.3 shows the sequence diagram for gesture detection and recognition. Here, initially the camera captures the gesture which will be detected and recognized by the gesture recognition algorithm and a particular mouse event will be performed based on it.

**4.4 Work Flow**

4.4.1 Finger Detection

Image Capture

ca

Capture

Detection

Classification

Mouse Events

Fig No 4.4.1

4.4.2 Face Detection

Image Capture

ca

Capture

Detection(Haar Cascade)

Classification

Mouse Events

Fig No 4.4.2

4.4.3 Color Detection

Image Capture

Detection(Normal Detection)

Classification

Mouse Events

Fig No 4.4.3

**4.5 Software Architecture**

Software Architecture is describes as follows:

Backend Software

GUI

Camera Interface

Mouse Events

Fig 4.5 Software Architecture

Main control is through a GUI which inturn provides peripheral interfaces as well as connectivity to the backend software. It also includes output display frame. The peripherals include camera to grab the gestures and the mouse for computer operation. Backend software incorporates OpenCV and Csharp language i.e. image processing algorithm. More particulars of the backend software is given in section 5.2.

**5. IMPLEMENTATION**

**5.1 PROCEDURAL FLOW**

Procedural flow for implementation of the project are identified by the major steps given in table 5.1.

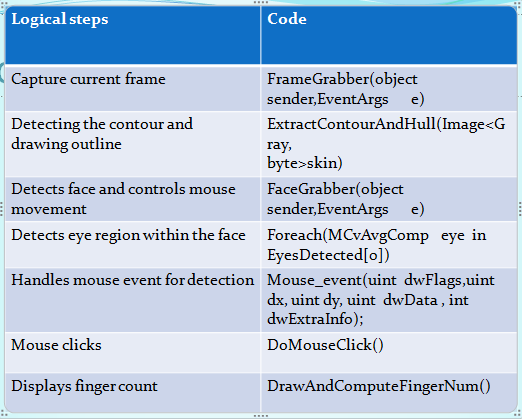


Table 5.1 Procedure for implementation

**5.2 PSEUDO CODE**

Pseudocode for implementing various parts are given in this section. This include camera frame grabbing, finger image processing, face detection, color detection. The major part of the code is implemented in Open CV while additional support code is written in C#.net

**5.2.1 Pseudo code for Finger detection**

void FingerGrabber(object sender, EventArgs e)

{

currentFrame = grabber.QueryFrame();

if (currentFrame != null)

{

currentFrameCopy = currentFrame.Copy();

skinDetector = new YCrCbSkinDetector();

Image<Gray, Byte> skin = skinDetector.DetectSkin(currentFrameCopy,YCrCb\_min,YCrCb\_max);

Image<Gray, Byte> cannyEdges = skin.Canny(cannyThreshold, cannyThresholdLinking);

ExtractContourAndHull(skin);

DrawAndComputeFingersNum();

imageBoxSkin.Image = skin;

imageBoxFrameGrabber.Image = currentFrame;

}

}

**5.2.2 Pseudo code for Face detection**

MCvAvgComp[][] facesDetected = grayFrame.DetectHaarCascade(\_face, 1.1, 1, Emgu.CV.CvEnum.HAAR\_DETECTION\_TYPE.FIND\_BIGGEST\_OBJECT, new Size(20, 20));

// MCvAvgComp[][] lefteyeDeteced = grayFrame.DetectHaarCascade(leye, 1.1, 1, Emgu.CV.CvEnum.HAAR\_DETECTION\_TYPE.FIND\_BIGGEST\_OBJECT, new Size(20, 20));

if (facesDetected[0].Length == 1)

{

MCvAvgComp face = facesDetected[0][0];

#region Luca Del Tongo Search Roi based on Face Metric Estimation --- based on empirical measuraments on a couple of photos --- a really trivial heuristic

// Our Region of interest where find eyes will start with a sample estimation using face metric

Int32 yCoordStartSearchEyes = face.rect.Top + (face.rect.Height \* 3 / 11);

Point startingPointSearchEyes = new Point(face.rect.X, yCoordStartSearchEyes);

yCoordStartSearchEyes);

Size searchEyesAreaSize = new Size(face.rect.Width, (face.rect.Height \* 2 / 9));

#endregion

}

**5.2.3 Pseudo code for Color detection**

void ColorGrabber(object sender, EventArgs e)

{

currentFrame = grabber.QueryFrame();

if (currentFrame1 != null)

{

currentFrameCopy = currentFrame1.Copy();

Image<Gray, Byte> huefilter = currentFrame.InRange(new Bgr(150,0,0),new Bgr(256,100,100));

imageBoxSkin.Image = huefilter;

imageBoxFrameGrabber.Image = currentFrame;

ExtractContourAndHull(huefilter);

DrawAndComputeFingersNum();

if(huefilter == currentFrame1.InRange(new Bgr(100,0,0),new Bgr(256,100,100)) )

{ blue = 6; }

}

}

**5.3 GUI Layout**

|  |
| --- |
| OUTPUT DISPLAY      00:00:00  Press to transfer control to mouse  Press to transfer control to webcam  Face Detection  Color Detection |

The button “Color Detection” transfers control to the color detection module and detects the color specified.

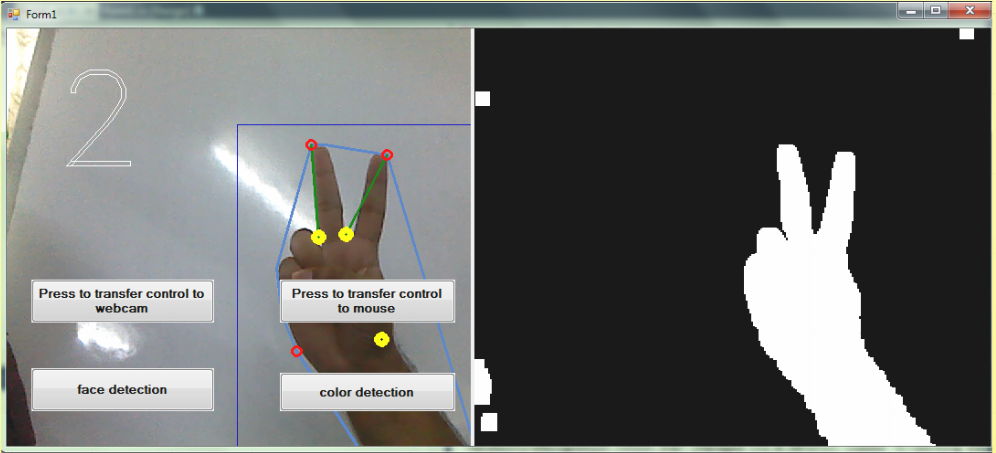
The button “Face Detection” transfers control to face detection module and detects the face and performs mouse movement. A timer is also used to perform the mouse click after 3 seconds when the left eye is closed.

The button “Press to transfer control to webcam” transfers control to finger detection module and performs the mouse events assigned.

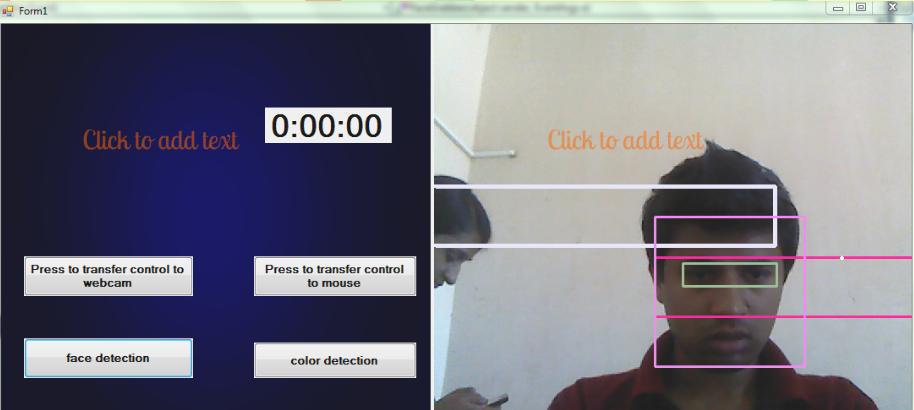
The button “Press to transfer control to mouse” transfers control to mouse.

1. **RESULTS AND ANALYSIS**

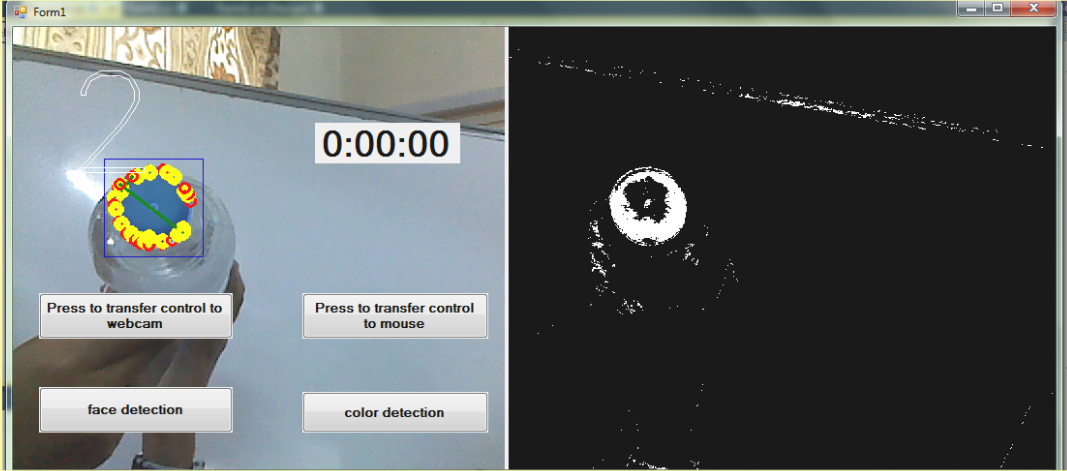
The snapshot of the captured output is as shown below:



The above snap shot shows the finger detection. When we show two finger it displays the number 2.



The above snap shot shows the face detection.



The above snap shot shows the color detection.

**7. TEST CASES**

**7.1 Software testing**

Software testing is the activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required result. Testing is an integral part in software development. It is broadly deployed in every phase in the software development cycle. Typically, more than 50% of the development time is spent in testing.

Testing objectives include:

* Testing is a process of executing a program with intent of finding an error.
* A good test case is one that has a high probability of finding a yet undiscovered error.
* A successful test is one that uncovers a yet undiscovered error.

Testing should systematically uncover different classes of errors in a minimum amount of time and with a minimum amount of effort. A secondary benefit of testing is that it demonstrates that the software appears to be working as stated in the specifications. The data collected through testing can also provide an indication of the software’s reliability and quality. But, testing cannot show the location of defect- it can only show that software defects are present. A successful test is one that uncovers a yet undiscovered error.

**7.2 Test Plan**

Testing is carried out in multiple levels. Activities at each level must be planned well in advance and it has to be formally documented. Based on the individual plan only, the individual test levels are carried out. Testing process starts with a test plan that identifies all the testing related activities that must be performed and specifies the schedule, allocates the resources and specifies guidelines for testing. The testing for this project is done in three phases, unit testing, integration testing and system testing. For the failure test cases regression testing is also done.

**7.3 Unit Testing**

A unit test is a method of testing the correctness of a particular module of source code. The idea is to write test cases for every no-trivial function or method in the module so that each test case is separate from the others if possible. The goal of unit testing is to isolate each part of the program and show that the individual parts are correct.

The unit test plan must clearly specify the scope of unit testing. In this project the unit test plan includes testing the basic input/output of the units along with their basic functionality. It also tests the boundary conditions. The input/output in the GUI mode pops up an error message in a separate window and resets the value to the defaults.

**7.4 Integration Testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**7.5 System Testing**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sl No | Test scenario | Input | Expected output | Actual Output | Status  (pass/fail) |
| 1 | void FingerGrabber(object sender, EventArgs e) | Video frames | Finger recognized  And MMCE | Finger recognized  And MMCE | pass |
| 2 | void FaceGrabber(object sender, EventArgs e) | Video frames | Face recognition  And MMCE | Face recognition  And MMCE | pass |
| 3 | void ColorGrabber(object sender, EventArgs e) | Video frames | Color recognition  And MM | Color recognition  And MM | pass |

**7.6 Test case**

**7.6.1 Unit Testing**

Table 7.1

MM – Mouse Movement

MMCE – Mouse Movement and mouse Click Events

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sl No | Test scenario | Input | Expected output | Actual Output | Status  (pass/fail) |
| 1 | private void Finger\_Click(object sender, EventArgs e) | FingerGrabber function triggered | Finger recognized frame | Finger recognized  Frame | pass |
| 2 | private void Face\_Click(object sender, EventArgs e) | FaceGrabber function triggered | Face recognized  Frame | Face recognized  Frame | pass |
| 3 | private void Color\_Click(object sender, EventArgs e) | ColorGrabber function triggered | Color recognized  frame | Color recognized  frame | pass |

**7.6.2 System Testing**

Table 7.2

**7.6.3 Integration Testing**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sl No | Test scenario | Input | Expected output | Actual Output | Status  (pass/fail) |
| 1 | if (fingerNum == 1 || fingerNum == 0)  if (fingerNum == 3)  if (fingerNum == 5) | Gesture recognized frames | Mouse movement  Mouse click  Opens specified folder | Mouse movement  Mouse click  Opens specified folder | pass |
| 2 | if (EyesDetected[0].Length != 0)  if (leftEyesDetected[0].Length == 0) | Gesture recognized frames | Mouse Movement  Mouse Click | Mouse Movement  Mouse Click | pass |
| 3 | if(huefilter == currentFrame.InRange(new Bgr(100,0,0),new Bgr(256,100,100)) ) | Gesture recognized frames | Mouse Movement | Mouse Movement | pass |

Table 7.3

**8. CONCLUSIONS AND FUTURE WORK**

**8.1 Conclusion**

In this project we have implemented gesturized operation of the computer. We are able to recognize six gestures namely single finger, double fingers, triple fingers, four fingers, five fingers and fist and perform mouse movements and clicks. We are also recognizing the face and performing mouse movement and click with right eye blink and one color detection to do mouse movement. The gesturized operation using low cost integrated camera has eliminated the need of using mouse and touch pad. Compared to current technologies available, we have achieved efficiency in terms of cost and operation speed. The various methods of face detection, finger detection and color detection will enable the handicaps as well as normal people to operate the system with ease and comfort.

**8.2 Future Scope**

Since the expected future is going to be holographic, gestures and hand movements can be the exact way to interface with the holographic technology.

**9. BIBLIOGRAPHY**

Books Referred

[1] Learning OpenCV Computer Vision with the OpenCV Library By Gary Bradski and Adrian Kaebler.

[2] OpenCV 2 Computer Vision Application Programming Cookbook By Robert Laganiere

Websites Visited

[1] Zhou Ren, Jingjing Meng and Junsong Yuan , “Depth camera based hand gesture recognition and its applications in Human-Computer-Interaction”, Information, Communications and Signal Processing (ICICS) 8th International Conference, 2011

[2] Feng-Sheng Chen, Chih-Ming Fu, Chung-Lin Huang, “Hand gesture recognition using a real-time tracking method and Hidden Markov models”, Institute of Electrical Engineering, National Tsing Hua University, Hsin Chu 300, Taiwan, ROC, 2003

[3]<http://code.google.com/p/inavigate> <http://www.appdonkey.com/apyearpDetail.php?app=iNavigate>

[4] Robust Real-Time Face Detection By Paul Viola and Michael J.Jones

[5] Chung-Lin Huang, Sheng-Hung Jeng, “A model-based hand gesture recognition system”, Electrical Engineering Department, National Tsing Hua University, Hsin Chu, Taiwan, 4 September 2000

[6] Suguna Devi S, Preethi K Mane and AjayKumar D, “ Face Detection System using OpenCV on Beagle Board ”, BMS College of Engineering, Bangalore-19, India

[7] Inseong Kim, Joon Hyung Shim, and Jinkyu Yang, “Face detection”

[8] Adesola Olaoluwa Anidu, Fasina Ebunoluwa Philip, “C# implementation of a face detection system using template matching and skin color information”, Anale. Seria Informatica Vol X fasc 1 – 2012, Annals. Computer Science Series. 10th Tome 1st Fasc. - 2012

[9] C˘at˘alin-Daniel C˘aleanu and Corina Botoca, “C# Solutions for a Face Detection and Recognition System”, FACTA UNIVERSITATIS (NIˇS) SER.: ELEC. ENERG. vol. 20, no. 1, 93-105 April 2007,

[10] Gary R. Bradski, “Computer Vision Face Tracking For Use in a Perceptual User

Interface ”, Microcomputer Research Lab, Santa Clara, CA, Intel Corporation

[11] Hart Lambur, Blake Shaw, “Gesture Recognition”, Department of Computer Science, Columbia University, December 21, 2004

[12] Beat Signer, Ueli Kurmann, Moira C. Norrie, “iGesture: A General Gesture Recognition Framework”, Institute for Information Systems, ETH Zurich

[13] M.Amutha Bharathi, “Color Detection”