A PROJECT REPORT

ON

HEAD TRACKING VIRTUAL MOUSE

SUBMITTED BY

GAURI KULKARNI

BABASAHEB KHEDKAR

ANIMESH KOTWAL

MOHIT MODI

UNDER THE GUIDANCE OF

PROF. K.S.SURESH BABU

DEPARTMENT OF COMPUTER ENGINEERING

PILLAI'S INSTITUTE OF INFORMATION TECHNOLOGY,

ENGINEERING, MEDIA STUDIES & RESEARCH

NEW PANVEL – 410 206

UNIVERSITY OF MUMBAI

Academic Year 2012 – 13

Department of Computer Engineering

Pillai's Institute of Information Technology,

Engineering, Media Studies & Research

New Panvel – 410 206

CERTIFICATE

This is to certify that the requirements for the project entitled 'HEAD TRACKING VIRTUAL MOUSE' have been successfully completed by the following students:

Name	**Roll No.**
Gauri Kulkarni	B-15
Babasaheb Khedkar	B-12
Animesh Kotwal	B-04
Mohit Modi	B-47

in partial fulfillment of Bachelor of Engineering of Mumbai University in the Department of Computer Engineering, Pillai's Institute of Information Technology, Engineering, Media Studies & Research, New Panvel during the academic year 2012 – 2013.

Internal guide _____

Prof. K.S. Suresh Babu

Internal Examiner _____ **External Examiner _____**

Head of Department _____ **Principal _____**

Prof. Sharvari Govilkar **Dr. R.I.K. Moorthy**

ACKNOWLEDGEMENT

No project is ever complete without the guidance of those expert how have already traded this past before and hence become master of it and as a result, our leader. So we would like to take this opportunity to take all those individuals how have helped us in visualizing this project.

We express our deep gratitude to our project guide **Prof. K.S.Suresh Babu** and **Prof Manjusha Deshmukh** for providing timely assistant to our query and guidance that he gave owing to his experience in this field for past many years. He had indeed been a lighthouse for us in this journey.

We extend our sincerity appreciation to all our Professors from PIIT for their valuable inside and tip during the designing of the project. Their contributions have been valuable in so many ways that we find it difficult to acknowledge of them individual.

We also grateful to our HOD **Prof. Sharvari Govilkar** and our principal **Dr. R.I.K.Moorthy** for extending their help directly and indirectly through various channel in our project work.

Gauri V. Kulkarni
Babasaheb Khedkar
Animesh Kotwal
Mohit Modi

ABSTRACT

Our project, Head Tracking Virtual Mouse, analyzes the biometric identification and tracking related technologies of human-computer interaction. Based on face detection algorithm, we propose a position-based head motion detection algorithm, which does not depend on the specific biometric identification and tracking. It uses feature classification method to detect eye opening and closing actions. We also design a software system to operate computer by image detection of head and eye movements. The combinations of head and eye movements, are mapped to various mouse events, including move, click and drag, and so on. This system can be used for the upper limb disabled who failed to use the traditional mouse and keyboard. Furthermore, it can also be used for general computer users to do neck rehabilitation training, computer somatic games, etc.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

Head Tracking Virtual Mouse is an application that uses the feature classification method to map the mouse pointer on the screen to the movements of head and eye in frames through a camera. The system analyzes the relationship between different combinations of the detected head and eye open and closing action, and then maps them to mouse events on the computer system. Our aim is to use this application mainly for the upper limb disabled who are unable to use the traditional mouse.

1.2 MOTIVATION OF THE PROJECT

The previous proposed systems used complex algorithms. They were based on the biometric identification techniques. Some needed to mount devices on the user like Lasers which was not feasible. Hence, our aim is to devise an application that will be cost effective and not be dependent on the biometrics but on the feature classifications of the user. It should use less hardware and simpler algorithms. The objective is to use such a system that will help the upper limb disabled who cannot use the traditional mouse or keyboard.

1.3 PROBLEM STATEMENT

The existing system is limited to the biometric identification. To enhance this, we have used the feature classification method.

There are certain problems in existing system as follows

-Mounting devices: These systems needed a mounted device like lasers or cameras on the user which became tedious.

-Biometric identification: The system used biometric identification for which the users had to register themselves before using the system. It wasn't open for all which has been rectified by the proposed application.

-Complex algorithms: The previous systems used many complex algorithms that needed a lot of calculations to be done depending on various markers. The new proposed algorithm that is the HAAR classification algorithm is easier and faster, thus increasing the response time for quicker access.

CHAPTER 2

REVIEW OF LITERATURE

2.1 STUDY OF EXISTING SYSTEM

Chen et al developed a system that contains an infrared transmitter mounted onto the user's eye glasses, a set of infrared receiving modules that substitute the keys of a keyboard, and a tongue-touch panel to activate the infrared beam. Hutchinson et al studied the eye gaze direction to operate the computer by measuring the corneal reflection. These methods presented above are contact, which mostly use contact sensors to measure human reflections or activities. The contact methods can accurately detect features, but they commonly require expensive auxiliary equipment. Therefore, it is difficult to be applied widely. Recently, along with the development of image processing technology and the improvement of computer performance, it has been a hot research area of using the non-contact and image processing methods to study human interactions. Betke et al. proposed a system that tracks the biometric features and then translates them into the movements of the mouse pointer on the screen. Nabati and Behrad presented a novel approach to estimate the 3D head pose from a monocular camera images using various algorithms for the control of mouse pointer movements on the screen and clicking events. These non-contact methods are more comfortable and convenient for the users and involve less expensive communication devices. However, these methods have high requirements for the camera, as well as high performance of computer image processing and computing. Currently, no contact technology is mainly used in some special application systems. A specialized software or system to control computer through this technology is still not common.

We study the algorithms of detecting head movement and eye status. Based on these algorithms, we design a Head-Trace Mouse to operate computer by detecting the movements of head and eye.

2.2 PROPOSED SYSTEM

2.2.1 THE BASIS OF HEAD MOVEMENTS

The face detection is implemented by using Adaboost algorithm in our system, in which each frame of video streaming captured by a camera is input signal. Then we propose an algorithm for detection of head movements by analyzing face locations. We defined five motions as the basis of head movements, namely, standard head, head left, head right, head up, and head down, as shown in Fig. 2.1

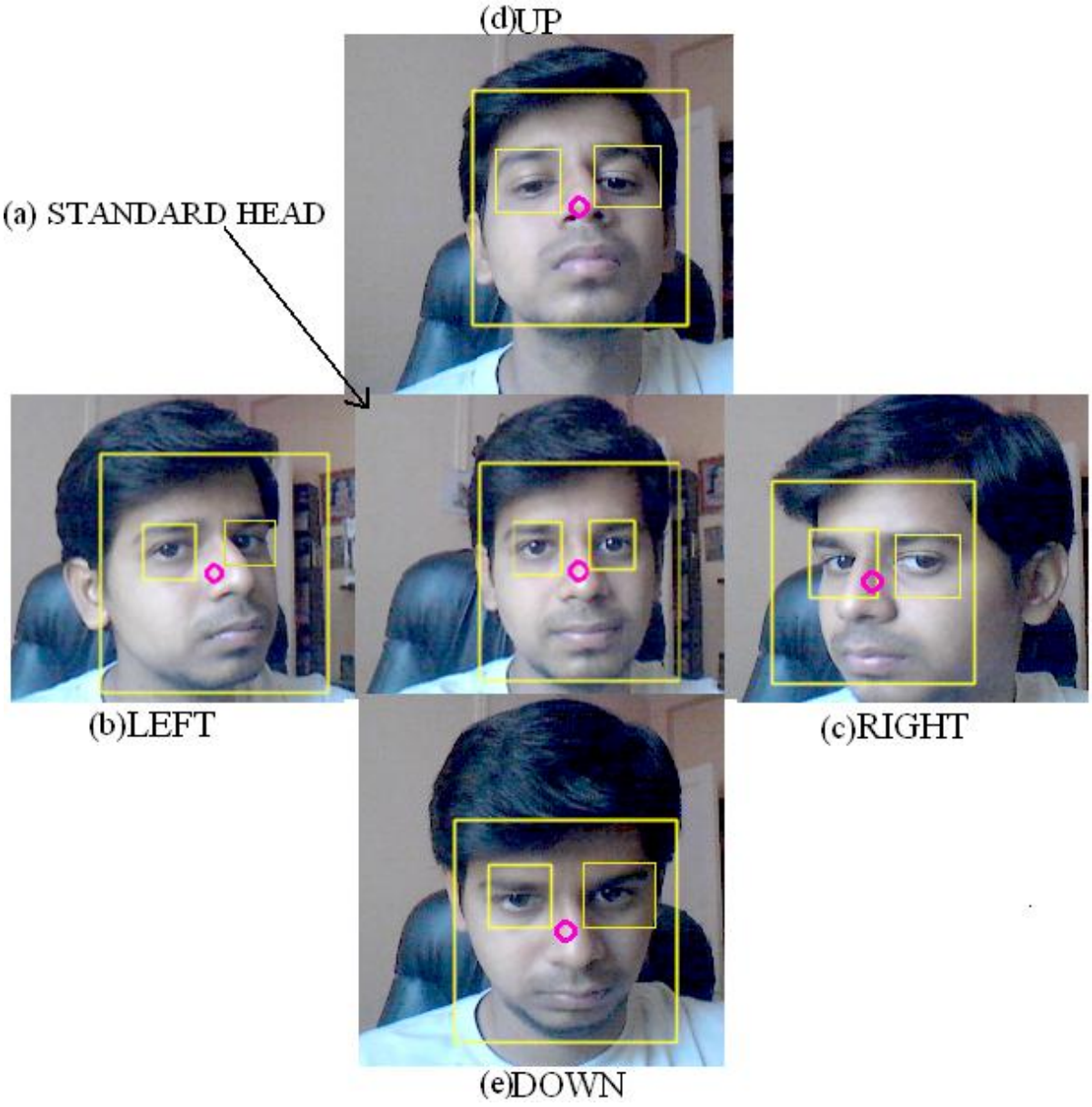


Fig.2.1 Basic head movements

Fig.2.1 includes five sub-graphs from (a) to (e). There are two rectangles in each sub-graph. The outer rectangular frames the face which represents the detected head (the head in this article refers to front area of the face), and the inner one frames the eyes which represents the detected eye open area.

2.2.2 PRINCIPLE OF HEAD-TRACE MOUSE SYSTEM

While the system is running, if standard head and eyes (as Fig.2.1 (a)) are detected, **Start** option can activate the system. After that, user can operate mouse by the head and eye movements. The working process and principles of the system are described as follows:

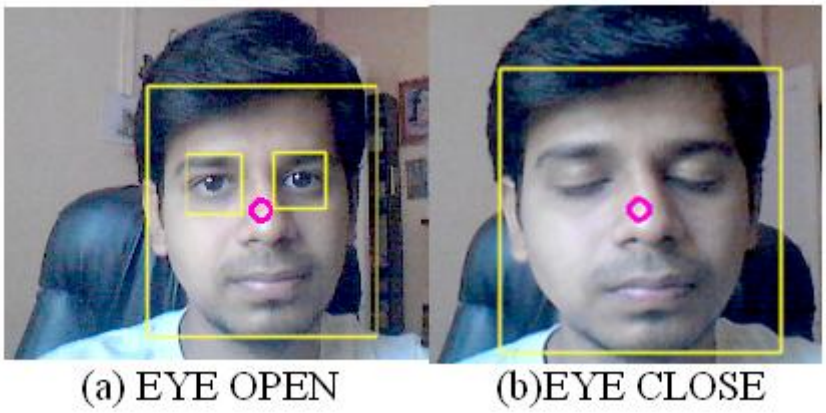


Fig. 2.2 Eye opening and closing

(1) If standard head is detected, indicating the head is still, the mouse is in the state of idle

(2) In the state of idle, if the movement of head left (as Fig. 2.1(b)) is detected, the cursor will move left horizontally. If the user keeps the action of head left, the cursor keeps moving left. When the user moves back to standard head, the cursor stops moving and goes back to the idle state. Mouse cursor moving right, up and down have the same principle as moving left.

(3) In the state of idle, if closed eye is detected, the selected option of clicking is activated. In the state of selection, closing of the eye performs the selected option of left click, right click, double-click or scroll commands. Usually, the movement of closing eye lasts about 2 seconds. Opening eyes again makes the cursor go back to idle.

(4) Motion and selection can be switched through the closing eye movement.

(5) If the threshold gets wrongly mapped then the **Reinitialize** button refreshes the threshold and sets a new threshold based after pressing the **Start** option again.

(6) **Stop** option stops the system.

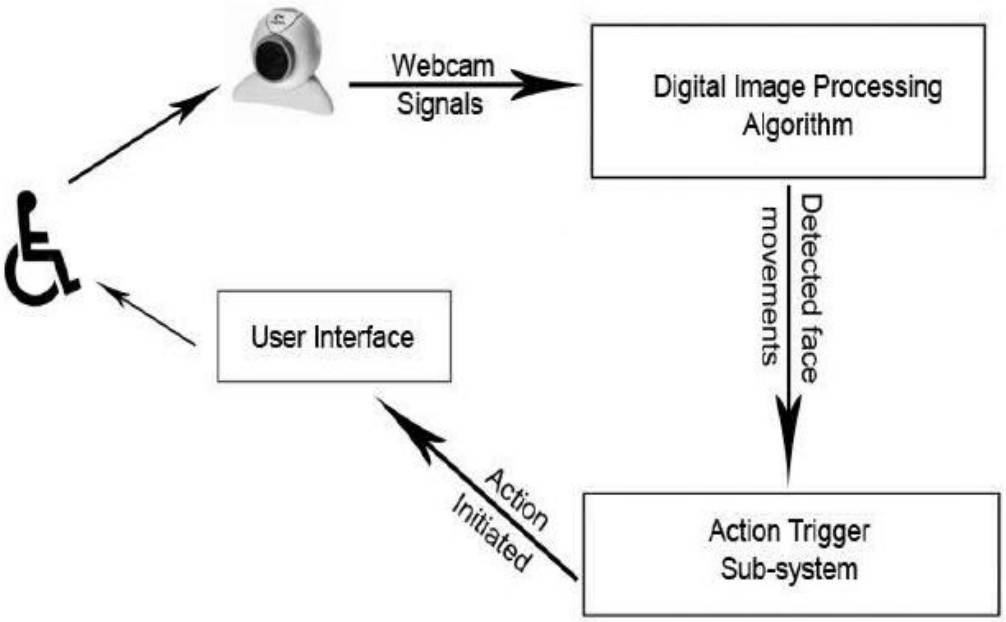


Fig.2.3 System flow diagram

2.2.3 ADVANTAGES

1. Quick response time
2. Customized processing
3. Small memory factor
4. Highly secure
5. Really helpful for disabled people

CHAPTER 3

REQUIREMENT ANALYSIS

Requirement analysis in systems engineering and software engineering, encompasses those tasks that go into determining the needs or conditions to meet for a new or altered product, taking account of the possible conflicting requirements of the various stakeholders, such as beneficiaries or users. It is an early stage in the more general activity of requirements engineering which encompasses all activities concerned with eliciting, analyzing, documenting, validating and managing software or system requirements.

3.1 FEASIBILITY ANALYSIS

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that is spent on it. Feasibility study lets the developer foresee the future of the project and the usefulness. A feasibility study of a system proposal is according to its workability, which is the impact on the organization, ability to meet their user needs and effective use of resources. Thus when a new application is proposed it normally goes through a feasibility study before it is approved for development.

The document provide the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as Technical, Economic and Operational feasibilities. The following are its features:

3.1.1 TECHNICAL FEASIBILITY

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs and procedures. Having identified an outline system, the investigation must go on to suggest the type of equipment, required method developing the system, of running the system once it has been designed.

Technical issues raised during the investigation are

Does the existing technology sufficient for the suggested one?

Can the system expand if developed?

The project should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed within latest technology.

Through the technology may become obsolete after some period of time, due to the fact that never version of same software supports older versions, the system may still be used. So there are minimal constraints involved with this project. The system has been developed using Java the project is technically feasible for development.

3.1.2 ECONOMIC FEASIBILITY

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

3.1.3 OPERATIONAL FEASIBILITY

Operational feasibility is a measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development. Our proposed system overcomes all the problems related to present complex system and satisfies all the scope as defined.

3.1.4 BEHAVIORAL FEASIBILITY

This includes the following questions

- Is there sufficient support for the users?
- Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible.

3.2 FUNCTIONAL REQUIREMENT ANALYSIS

The Functional Requirements Definition documents and tracks the necessary information required to effectively define the process and this document is created during the Planning Phase of the project. From the functional perspective the project is Measurable, Realistic and Complete. Our project doesn't require any special training to use it and can be used by any common man. Considering the problems with the current system, understanding the user's needs and expectations the requirements for the proposed system is collected and found to be complete. Also the system does not require any additional features that may cause delay in the release of the project. This makes our project functionally measurable, realistic and complete.

3.3 NON-FUNCTIONAL REQUIREMENTS

3.3.1 QUALITY OF SERVICE

Quality Of Service (QOS) is a major issue in desktop applications. In the proposed system, there is constant transfer of data between the peripheral (here camera) and the system. The simplicity of the algorithm enhances the response time, thus bettering the QOS of the system.

3.3.2 AVAILABILITY

The user does not need to register to the system before using it. It can be used by any Human due to the feature classification method used in the system. It is easy to use and available to all.

3.3.3 SUPPORTABILITY

This system is supported by any Java application. It can be used with any web camera. The accuracy of the system though may depend on the specifications of the camera.

3.4 TECHNOLOGY USED

3.4.1 JAVA TECHNOLOGY

Java technology is both a programming language and a platform. The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- Object Oriented
- Architecture neutral
- Portable

- Multithreaded
- High Performance
- Dynamic
- Robust
- Secure
- Distributed

In the Java programming language, all source code is first written in plain text files ending with .java extension. The source files are then compiled into .class files by the javac compiler. A .class file does not contain code that is native to your processor; it instead contains bytecodes – the machine language of Java Virtual Machine (JVM). The java launcher tool then runs your application with an instance of the JVM.

3.4.2 JAVA PLATFORM

A platform is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Microsoft Windows, Linux, Solaris OS and Mac OS. Most platforms can be described as a combination of the operating system and underlying hardware. The Java platform differs from most other platforms in that it's a software- only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

- The Java Virtual Machine
- The Java Application Programming Interface (API)

You've already been introduced to the Java Virtual Machine; it's the base for the Java platform and is ported onto various hardware-based platforms.

The API is a large collection of readymade software components that provide many useful capabilities. It is grouped into libraries of related classes and interfaces; these libraries are known as packages.

As a platform independent environment, the Java platform can be a bit slower than native code. However, advances in compiler and virtual machine technologies are bringing performance close to that of native code without threatening portability.

3.5 SYSTEM REQUIREMENTS

To be used efficiently, all computer software needs certain hardware components or other software resources to be present on a computer. These pre-requisites are known as system requirements and are often used as a guideline as opposed to an absolute rule.

Supported Operating Systems

- Windows XP/Vista/7
- Supports both 32 bit as well as 64 bit OS.

The software requires following applications installed on the server machine:

- JDK1.6 or above
- NetBeans IDE 7.0 or above.
- OpenCV

3.6 HARDWARE REQUIREMENTS

1. Web camera
2. Laptop or a computer

CHAPTER 4

PROJECT DESIGN

Project Design is the process of creating a new product to be sold by a business to its customers. It is efficient and effective generation and development of ideas through a process that leads to new products.

4.1 DESCRIPTION OF OVERALL SYSTEM ARCHITECTURE

A system architecture or systems architecture is the conceptual model that defines structure, behavior and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structure of the system which comprises system components, the externally visible properties of those components and provides a plan from which products can be procured and systems developed, that will work together to implement the overall system.

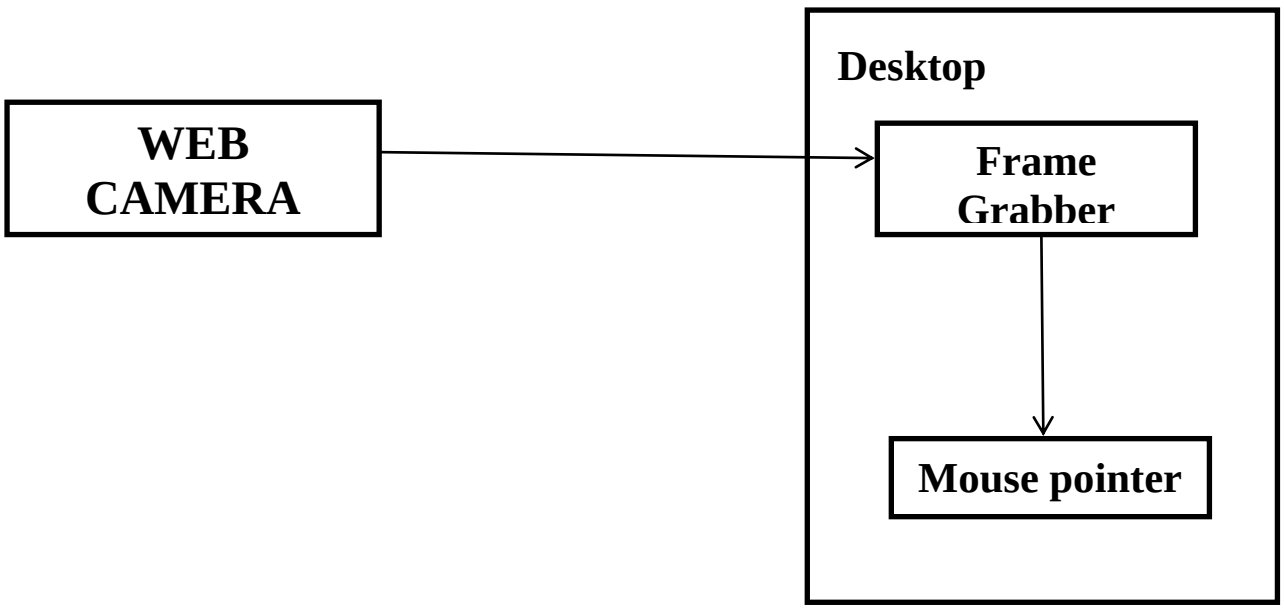


Fig.4.1 Architectural Representation

Detailed description about the System architecture:

1. Web camera

A webcam is a video camera that feeds its image in real time to a computer or computer network. It is a hardware device that inputs images of the User and supplies it to the frame grabber. Just like a digital camera, it captures light through a small lens at the front using a tiny grid of light-detectors that converts the picture in front of the camera into digital format. Unlike a digital camera, a webcam has no built-in memory chip. Hence, it transmits them immediately to a computer.

2. Desktop

Most of the processing takes place on the desktop. It only has to display the image received from the web camera and provide display to the user. It has the following components:

- **Frame grabber**: A frame grabber is an electronic device that captures individual, digital still frames from an analog video signal or a digital video stream. It is usually employed as a component of a computer vision system, in which video frames are captured in digital form and then displayed, stored or transmitted in raw or compressed digital form.
- **Mouse Pointer**: The mouse cursor, or mouse arrow, or mouse pointer is often shaped like an arrow or a small hand with the index finger pointing towards the top of the display device. The mouse pointer moves as the user moves his or her head and blinking of eyes triggers clicking events.

4.2 DATA FLOW DIAGRAM AND CONTROL FLOW DIAGRAM

4.2.1 DATA FLOW DIAGRAM

A data flow diagram (DFD) is a graphical representation of the ‘flow’ of data through an information system, modeling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing.

The camera grabs user photos at the rate of 30 frames per second and compares it with the already fed in and trained data to plot the rectangle on the face showing face detection.

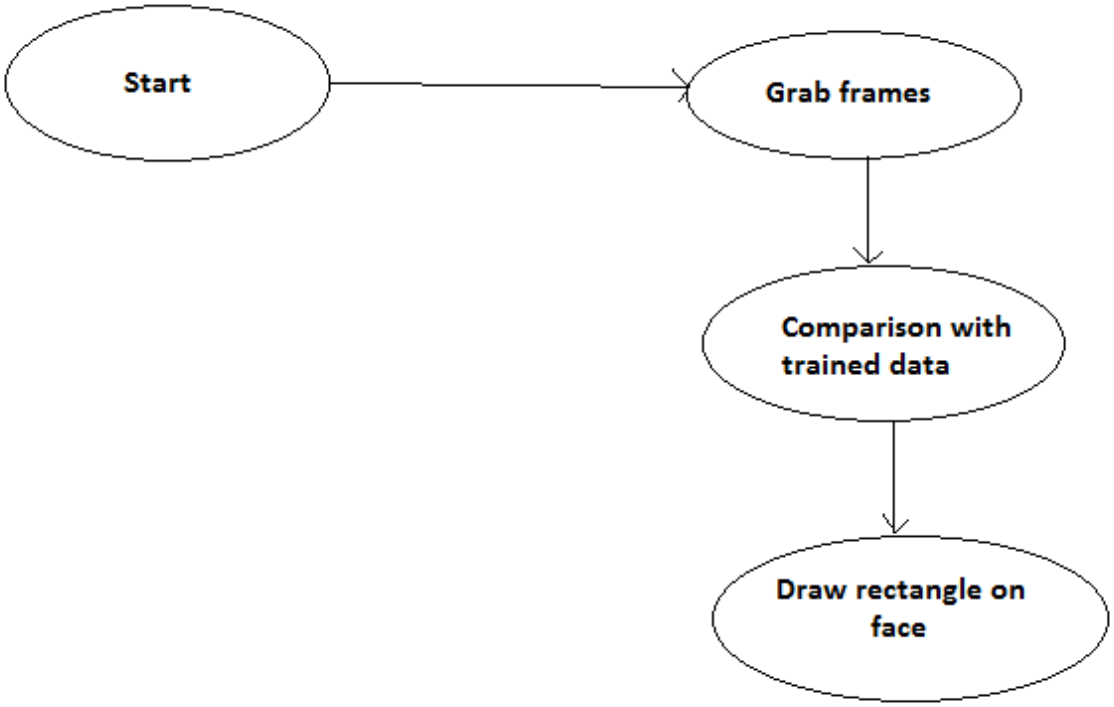


Fig 4.2 DFD level 0

After detecting the face, as movement occurs, the pointer on the screen moves accordingly by measuring gravity to the mapped movements of the mouse.

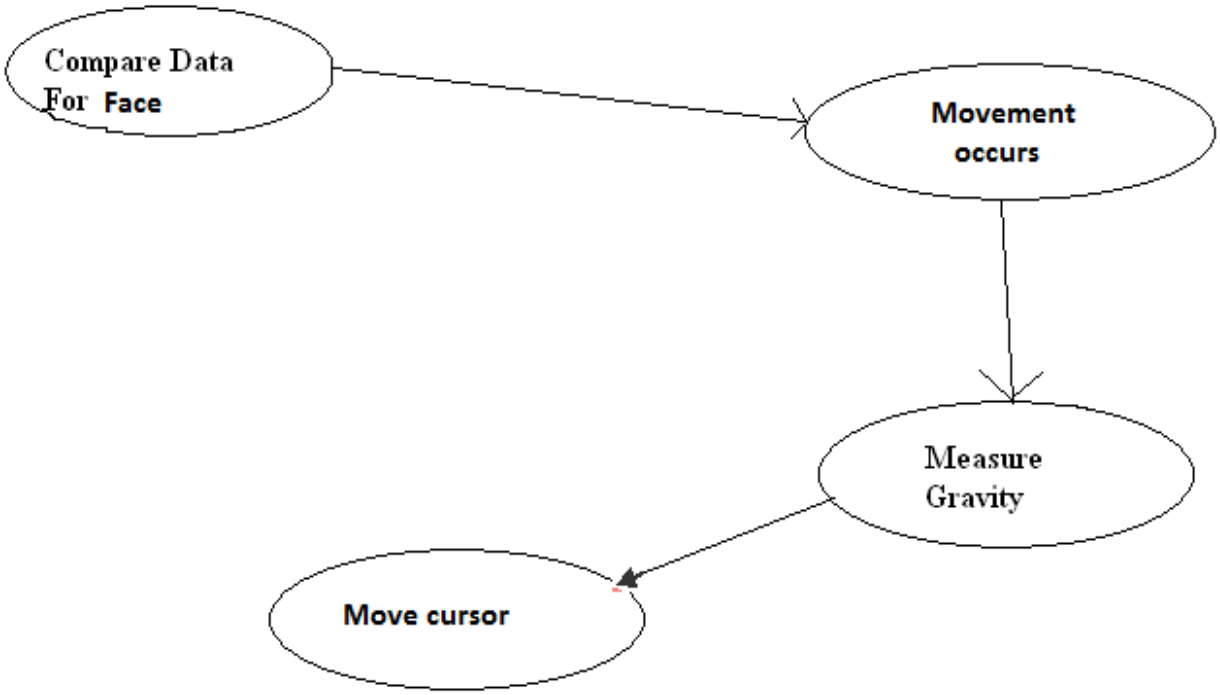


Fig 4.3 DFD level 1

For eye detection, the same procedure is carried out and a rectangle is drawn for detection. Now, by using algorithms of haar classification, the needed action of the mouse pointer is mapped with face movements and blinking of eyes.

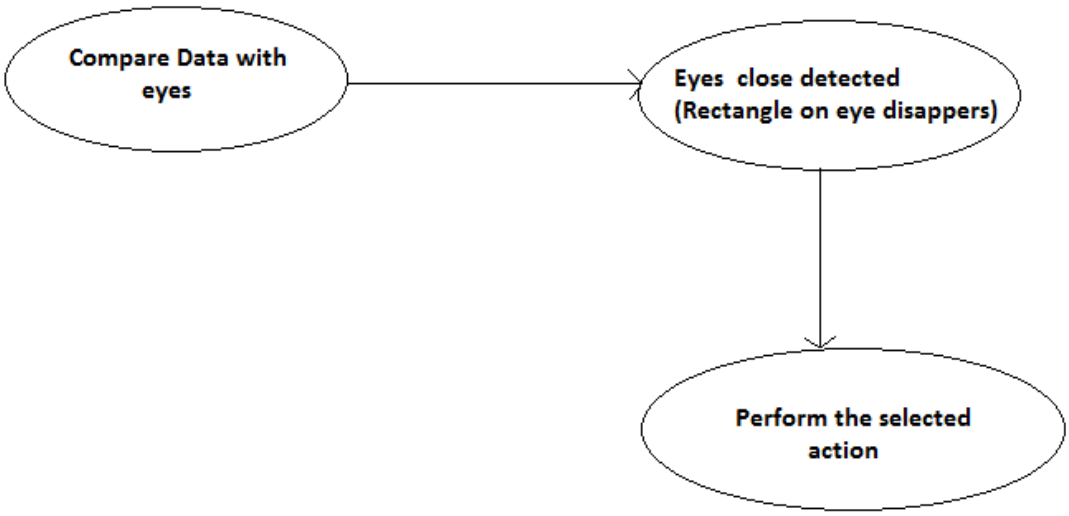


Fig.4.4 DFD Level 2

4.3 UML DIAGRAMS

Unified Modeling Language (UML) is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created, by the Object Management Group. It was first added to the list of OMG adopted technologies in 1997, and has since become the industry standard for modeling software-intensive systems.

4.3.1 USE CASE DIAGRAM

Use case diagrams are behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors). Each use case should provide some observable and valuable result to the actors or other stakeholders of the system. The User looks into the camera. The camera captures frames of the User and according to User head movements determines the motion of the pointer to the left, right, up and down. Similarly, User also judges whether to select mode 1 or mode 2 according to the mouth open and close motion.

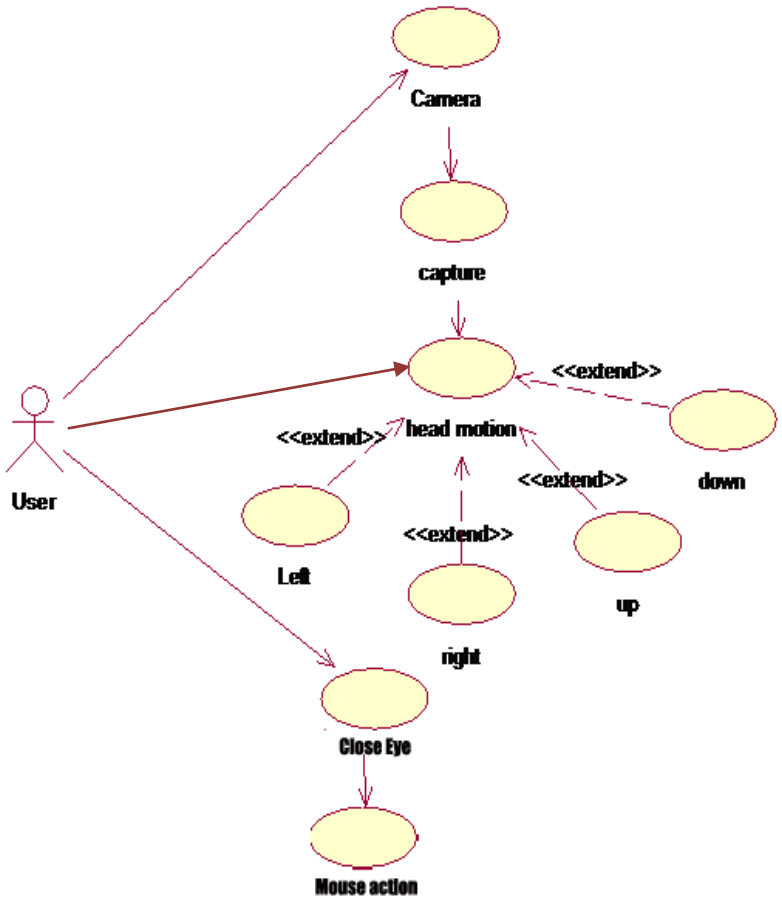


Fig 4.5 Use case diagram

4.3.2 CLASS DIAGRAM

A class diagram in Unified Modeling Language(UML) is a type of static structure diagram that describe the structure of a system by showing the system's classes, their attributes, operations(or) methods and relationship between the classes. The class diagram is the main building block in object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. The classes in a class diagram represent both main objects and or interaction in the application and the objects to be programmed.

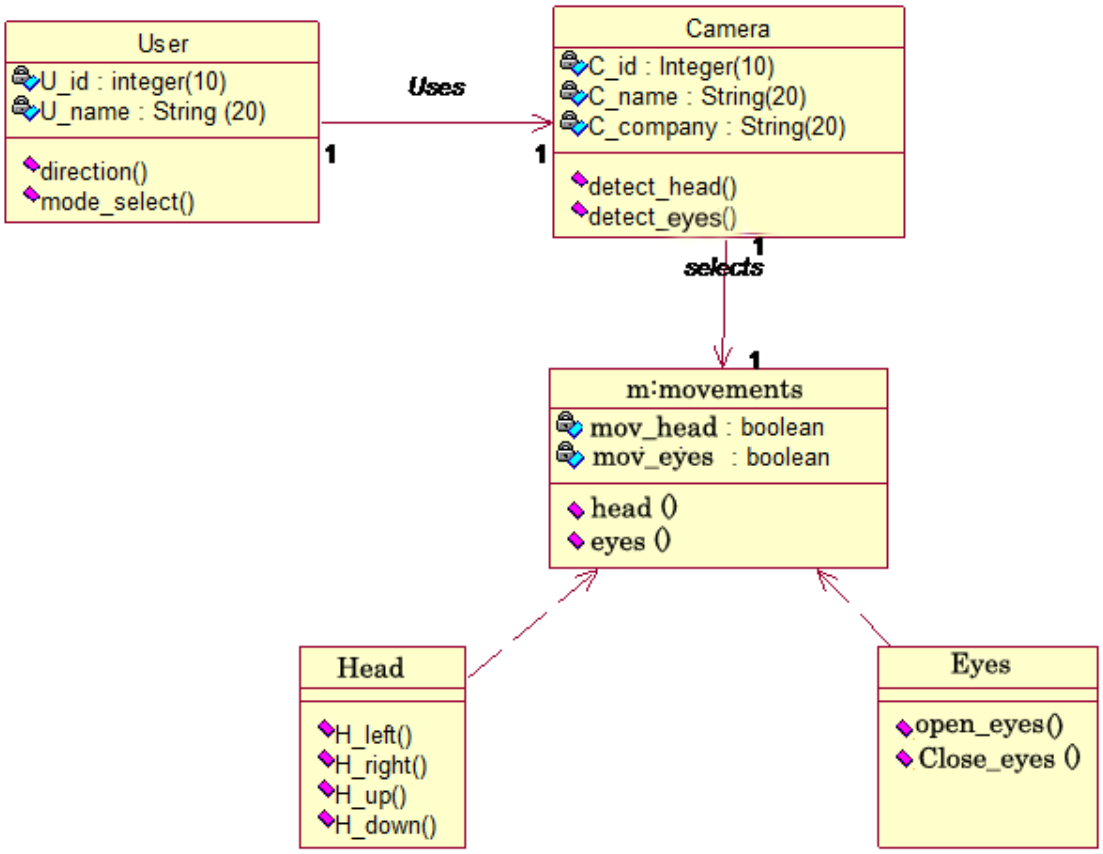


Fig 4.6 Class diagram

We define 3 classes as User, the Camera that acts as a medium to capture frames and Mode that determines the actions to be done according to the result of the algorithm for comparing frames and selecting the action.

User

The User moves the head and chooses the direction for the mouse pointer to be moved on the screen. The User also closes the eyes to perform clicking actions.

Camera

Camera grabs pictures and according to algorithms detects the head and eyes. It draws a rectangle on the eyes and head to show detection.

Movement

There are two movements mapped to the head and eyes. Head is used for moving of the pointer on the screen whereas eyes are used to select clicking options like left click, right click, scroll up and scroll down.

4.3.3 SEQUENCE DIAGRAM

A sequence diagram in a Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart.

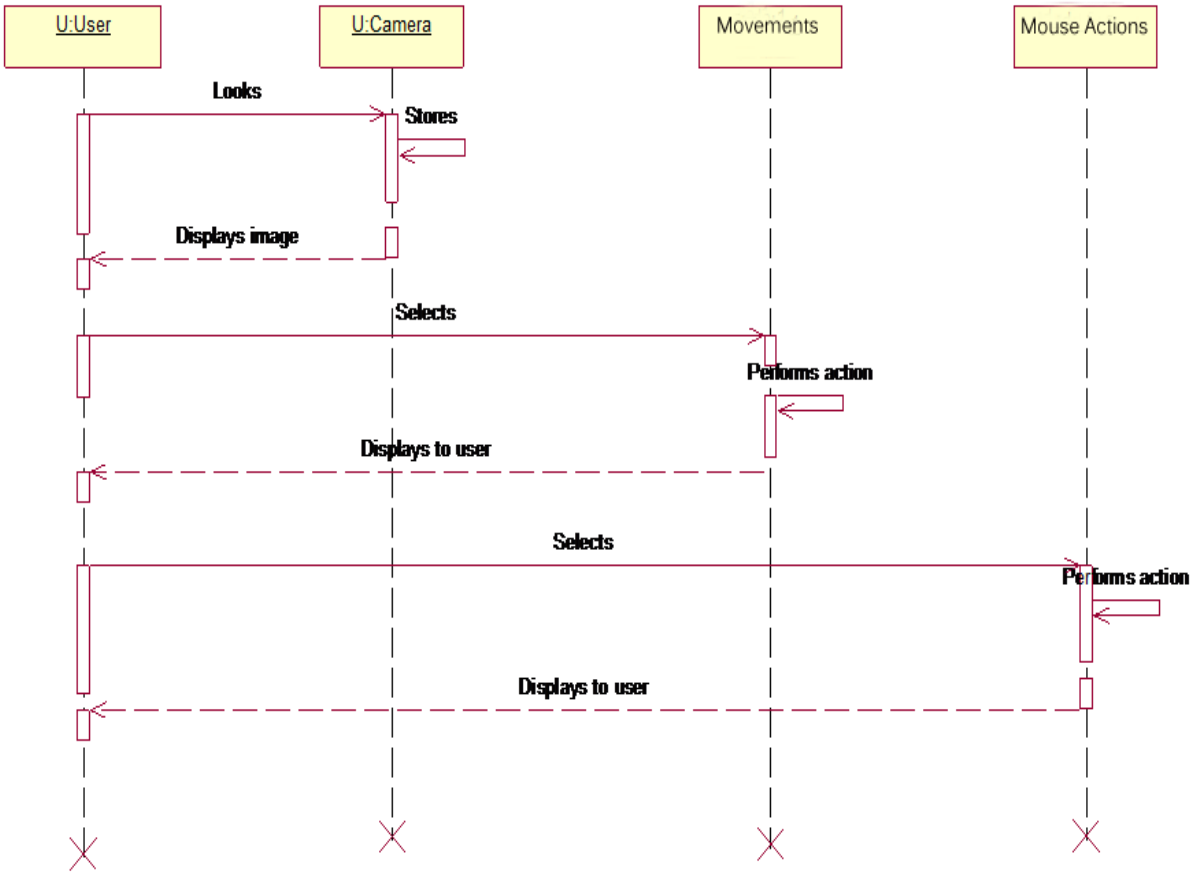


Fig 4.7 Sequence diagram

The User looks into the Camera. The camera stores the frames and displays the image to the User. The User selects the movements to be performed. If movements are detected, the pointer is accordingly moved on the screen and displayed to the user. If Mouse action is selected by the User, the needed selection mode is shown and displayed back to the User. When the application is closed, all the objects are stopped.

4.3.4 COLLABORATION DIAGRAM

A collaborating diagram also called a communication diagram or interaction diagram is an illustration of the relationships and interaction among software objects in the Unified Modeling Language (UML). The concept is more than a decade old although it has been refined as modeling paradigms have evolved.

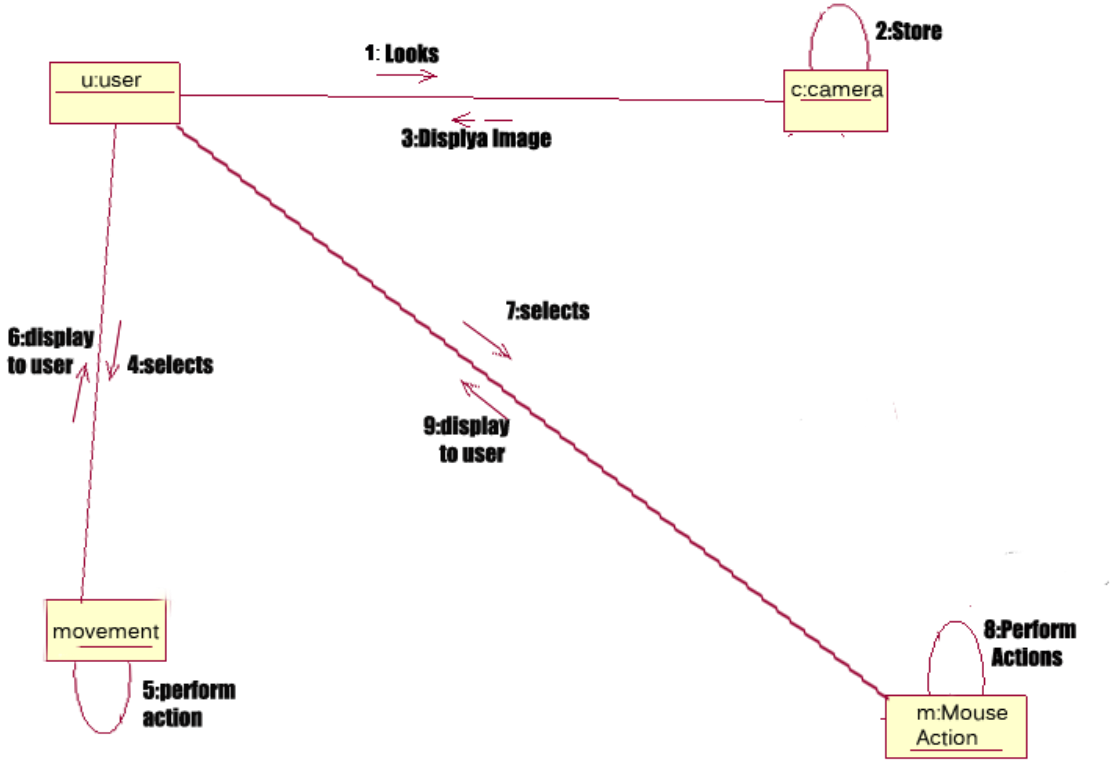


Fig. 4.8 Collaboration diagram

User looks into the camera. The camera grabs frames and displays image to the user. User selects the movements to be done by the pointer which are accordingly performed by the system and displayed to the User. User also selects Mouse actions to be performed which are then displayed back to the User.

4.3.5 STATE TRANSITION DIAGRAM

State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction. Many forms of state diagrams exist, which differ slightly and have different semantics. State diagrams are used to give an abstract description of the behavior of a system. This behavior is analyzed and represented in series of events that could occur in one or more possible states.

Following are the main purposes of using State chart diagrams:

- To model dynamic aspect of a system.
- To model life time of a reactive system.
- To describe different states of an object during its life time.
- Define a state machine to model states of an object

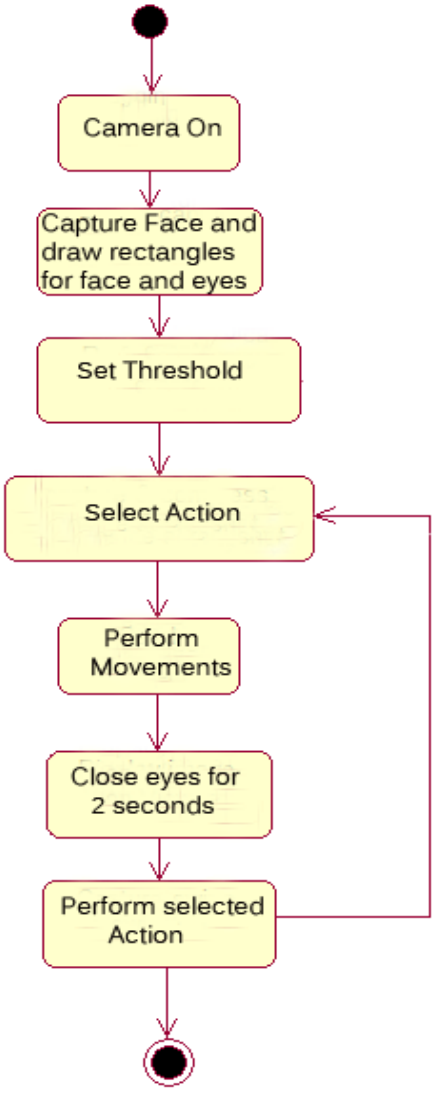


Fig. 4.9 State Transition Diagram

4.3.6 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of work flows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

Activity diagrams are constructed from a limited number of shapes, connected with arrows. The most important shape types:

- rounded rectangles represent activities;
- diamonds represent decisions;
- bars represent the start (split) or end (join) of concurrent activities;
- a black circle represents the start (initial state) of the workflow;
- An encircled black circle represents the end (final state).

Arrows run from the start towards the end and represent the order in which activities happen. The join and split symbols in activity diagrams only resolve this for simple cases; the meaning of the model is not clear when they are arbitrarily combined with decisions.

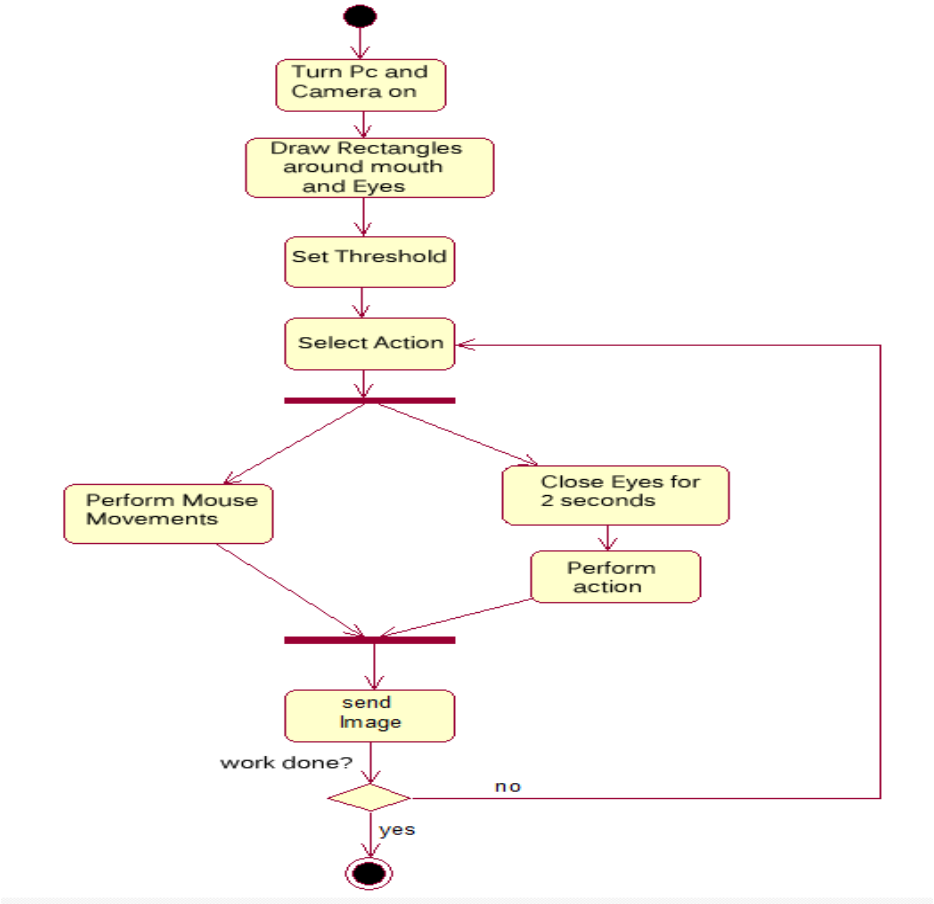


Fig 4.10 Activity diagram

4.3.7 COMPONENT DIAGRAM

In the component view, the component diagram (CPDs) shows the dependencies between compilation units, typically including sources files, and the runtime components of the final system. In the process view, the CDPs show the dependencies between processes (Schedulable units that may be selected for execution on the host machine's processor) in terms of the runtime components. Runtime components typically include:

- Executable
- Dynamically linked libraries(DLLs)
- Overlays
- Shell scripts or batch files

These diagram are intended to show the allocation of classes to compilation units and process respectively, although UML does not define how this is to occur. Cradle will support this mapping by providing a frame called CLASSES in which the names of the classes contained within the components or task can be listed.

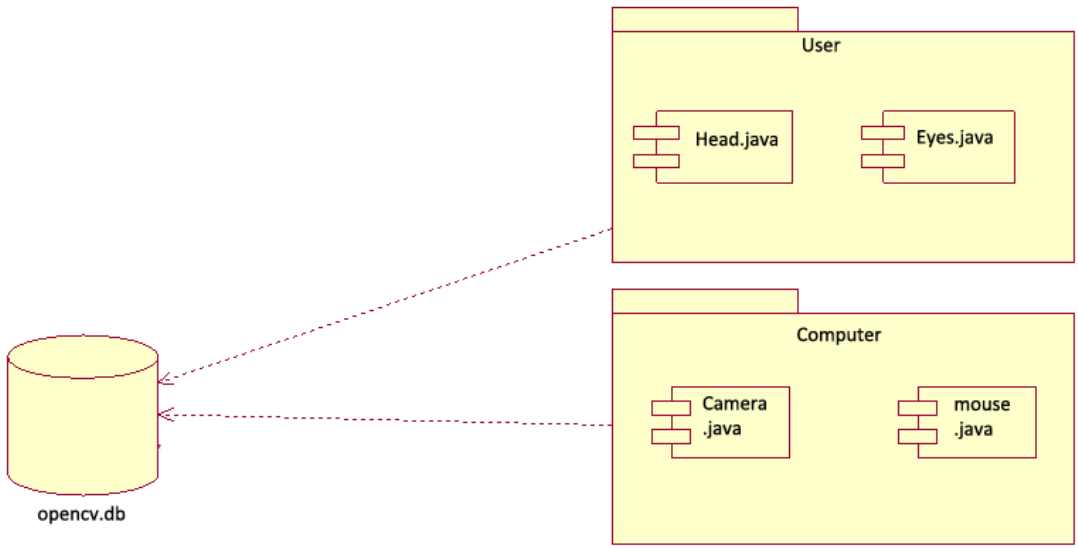


Fig. 4.11 Component diagram

The opencv database contains number of images and trained data that are to be compared with for the detection of Head and Eyes of the User. These User data are fed in the computer through the Camera. Accordingly after comparing with the database the computer performs the mapped action of the Mouse class.

4.3.8 DEPLOYMENT DIAGRAM

A descriptor of the deployment view of the architecture describes the various physical nodes for the most typical platform configuration. Also describes the allocation of the task to the physical nodes.

A deployment diagram depicts a static view of the run-time configuration of processing nodes and the components that run on those nodes. In other words, deployment diagrams show the hardware for your system, the software that is installed on that hardware, and the middleware used to connect the disparate machines to one another. You want to create a deployment diagram for applications that are deployed to several machines, for example a point-of-sales application running on a thin-client network computer which interacts with several internal servers behind your corporate firewall or a customer service system deployed using web services architecture such as Microsoft's .NET. Deployment diagrams can also be created to explore the architecture of embedded systems, showing how the hardware and software components work together.

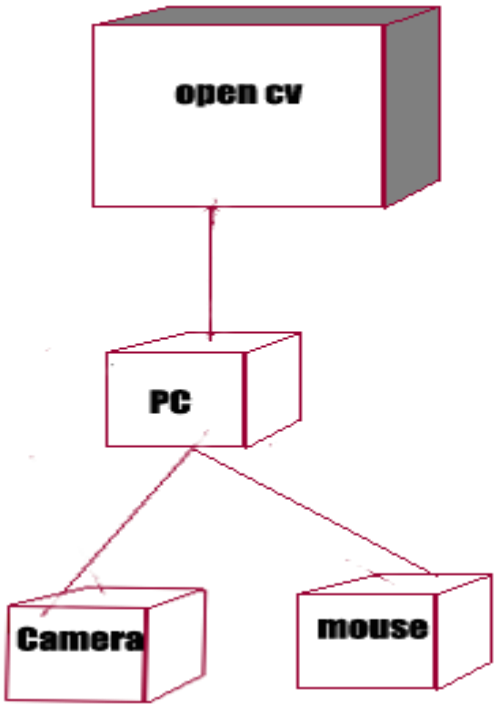


Fig.4.12 Deployment diagram

CHAPTER 5

IMPLEMENTATION DETAILS

5.1 ALGORITHM USED

First the system captures images by camera then detects the head area in the images. Let the origin coordinates (0, 0) be at the top left corner in the Figure. And the horizontal and vertical coordinate are noted x and y respectively, shown as an example in Fig. 5.1(a). The coordinate values are calculated in pixels. The rectangle which frames the face is the detected head area. We calculate the geometric center of the rectangle, and name it as head central coordinates, i.e. (Sx, Sy) in Fig. 5.1(b). Then we can analyze the specific head movement by time series relationship of the central coordinates. The algorithm HEADMOVE for detection of head movements is described as below.

(1) Initialization: User sits up in front of the computer. Let the Head-Trace Mouse run. If the head is detected, the head signals in the first 3 seconds are initialized by statistical methods, and then the head central coordinates (Sx, Sy) of the standard head is calculated, as shown in Fig.5.1.

(2) Set threshold value: Determine the threshold value (Kx, Ky) based on experience.

(3) Judge the head movements: Analyze the images after initialization. The head central coordinates of one image is noted as (Cx, Cy). We compare (Cx, Cy) with (Sx, Sy) to get the following conclusions:

If Cx - Sx> Kx, the judgment is that head moves left, abbreviated as left.

If Cx - Sx < -Kx, right.

If Cy - Sy < -Ky, up.

If Cy - Sy> Ky, down.

If | Cx - Sx | < Kx and | Cy - Sy | < Ky, standard head.

(4) Standard head relocation: If the standard head has been detected in several continuous images, the average value of these head central coordinates will be calculated as the new head central coordinates (Sx, Sy) of the standard head.

(5) Go back to step (2)

Fig.5.1 shows a standard head image captured by camera, where the outer rectangular frames the detected standard head. In Fig 5.1 the central point (Sx, Sy) in the rectangular with solid lines, is the central coordinates of the standard head. The solid lines frames rectangular shows the region of head motionless. In the course of system operation, if the

head central coordinates are within this region, the head is declared as motionless. If not, an associated movement is ensured. In Fig 5.1 the U, L, D, R parts are the head movement direction, meaning head move up, left, down, right, respectively, as noted in step (3) of HEADMOVE algorithm.

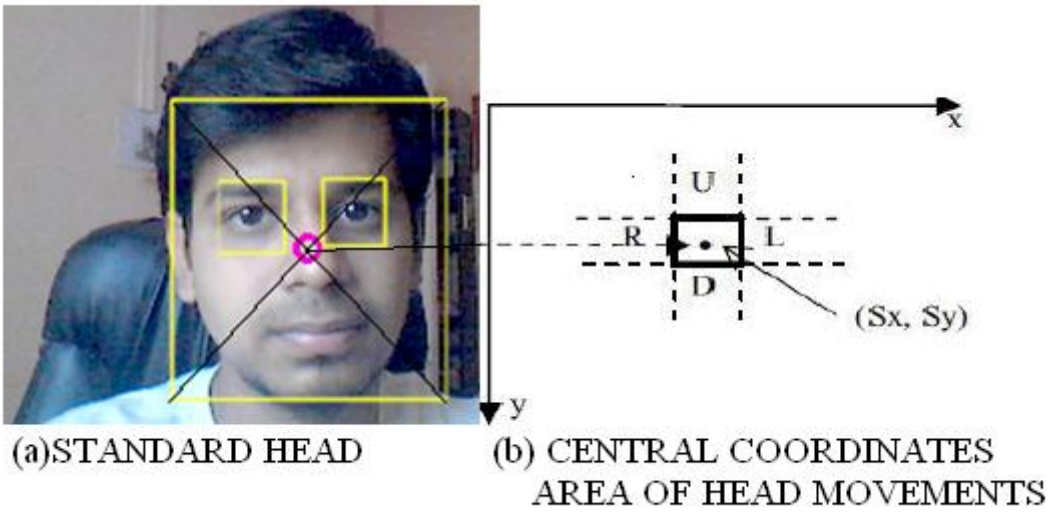


Fig 5.1 Selecting threshold value

5.2 OUTPUT SCREENS

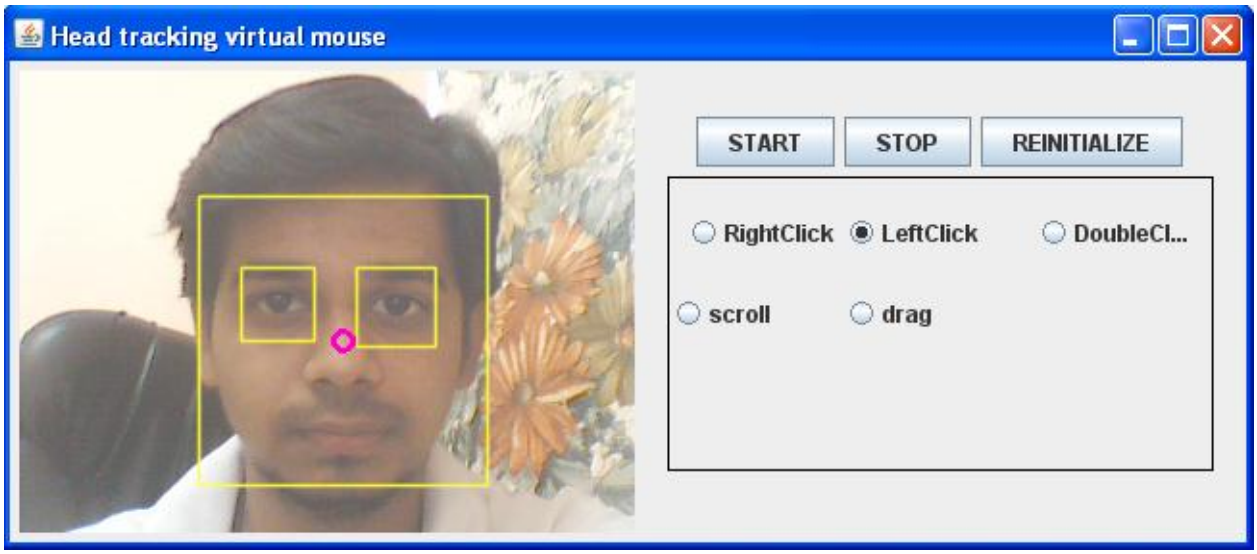


Fig.5.2 Face and eye detection

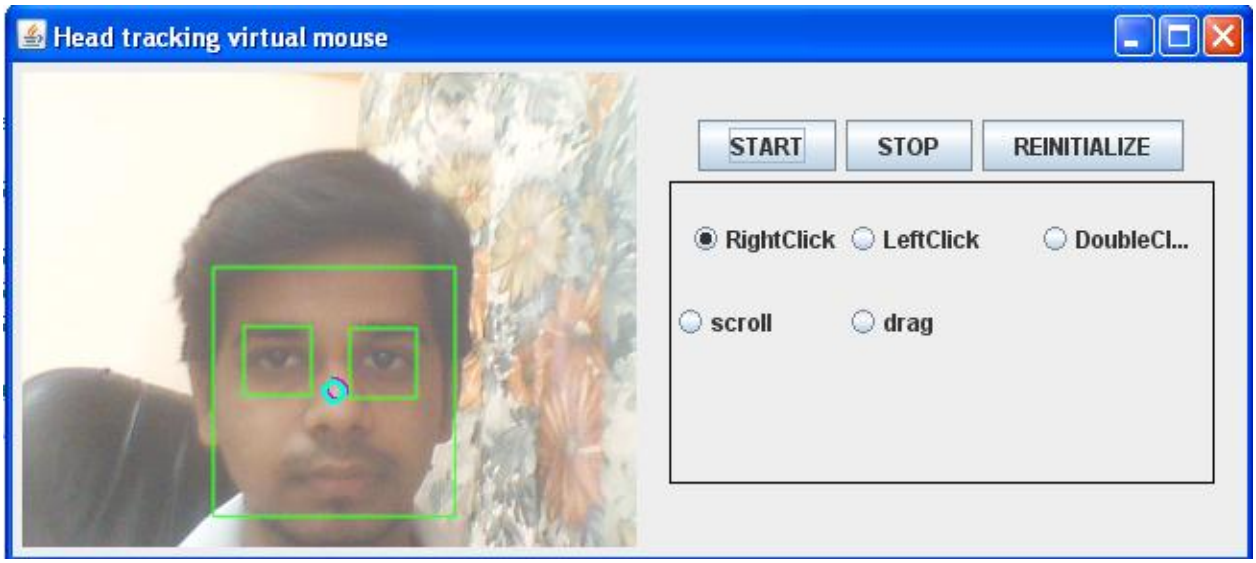


Fig.5.3 Threshold selected

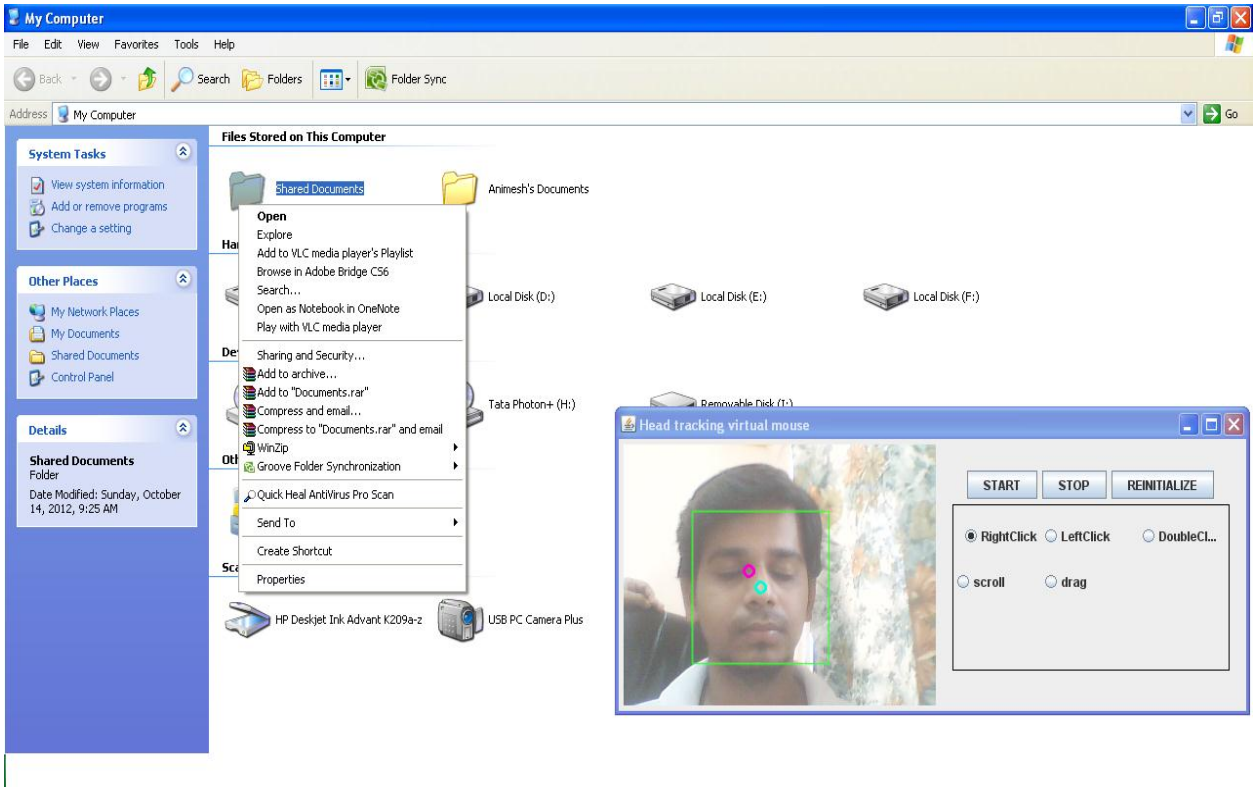


Fig.5.4 Right click event

CHAPTER 6

TESTING

6.1 UNIT TESTING

Unit testing is a procedure used to validate that individual units of source code are working properly. A unit is a smallest testable part of an application. Utilizing a set of unit tests can dramatically reduce the number of bugs and the risks with untested code. In our project we have performed unit testing to isolate each part of the program and show that the individual parts of the program work properly.

6.2 BLACK BOX TESTING

Black Box testing treats the software as a black-box without any understanding as to how the internals behave. Thus, we provided the input data and only saw the output from the test object. In this level of testing we provided thorough test cases to simply verify that for a given input, the output value is the same as the expected value specified in test cases.

6.3 WHITE BOX TESTING:

White Box testing is when the tester has access to the internal data structures, codes, algorithms. For this reason unit testing and debugging can be classified as white box testing and it usually requires writing code, or at a minimum stepping through it, and thus requires more skills than the black box tester. If the software in test is an interface, white box testing is almost always required.

6.4 INTEGRATION TESTING:

Integration testing is a logical extension of unit testing. In its simplest form, two units that have already been tested are combined into a component and the interface between them is tested. It is that phase of software testing in which individual software modules are combined and tested as a group. It follows unit testing and precedes system testing.

6.5 SYSTEM TESTING:

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic.

6.6 PERFORMANCE TESTING

Software performance testing is used to determine the speed or effectiveness of a computer, network, software program or device. This process can involve quantitative tests done in a lab, such as measuring the response time or the number of MIPS (millions of instructions per second) at which a system functions.

Qualitative attributes such as reliability, scalability and interoperability may also be evaluated. Performance testing is often done in conjunction with stress testing

6.7 TEST CASES

The following are the test cases in our project:

Test Objective: To test individual module of system.

Test Case	Case Description	Action	Expected Result	Actual Result	Pass/ Fail
Eyes not detected	User wearing spectacles	Compare with trained data	It should detect eyes	Eyes may not be detected	Fail
Eyes not detected	User is sitting more than 2 feet away	Compare with trained data	User's eyes should be detected	User is not properly detected	Fail
Fluctuation in detecting features	Brightness of the surrounding is more/less than needed	Compare with trained data	User features should be properly detected	Features are not properly detected	Fail
Eyes detected	Colour of eyes are not black	Compare with trained data	Eyes may not be detected	Eyes are properly detected	Pass
Face detected	Style of hair and features of face are not the usual	Compare with trained data	Face may not be detected	Face gets properly detected	Pass
Threshold not set rightly	Face is not rightly perpendicular to the camera	Compare with trained data	Threshold should be the center of the face	Threshold may shift from the center	Fail
Action performed	Eyes are not properly closed	Compare with trained data	Clicking action should take place	Clicking selected option is performed	Pass

Table 6.1 test cases

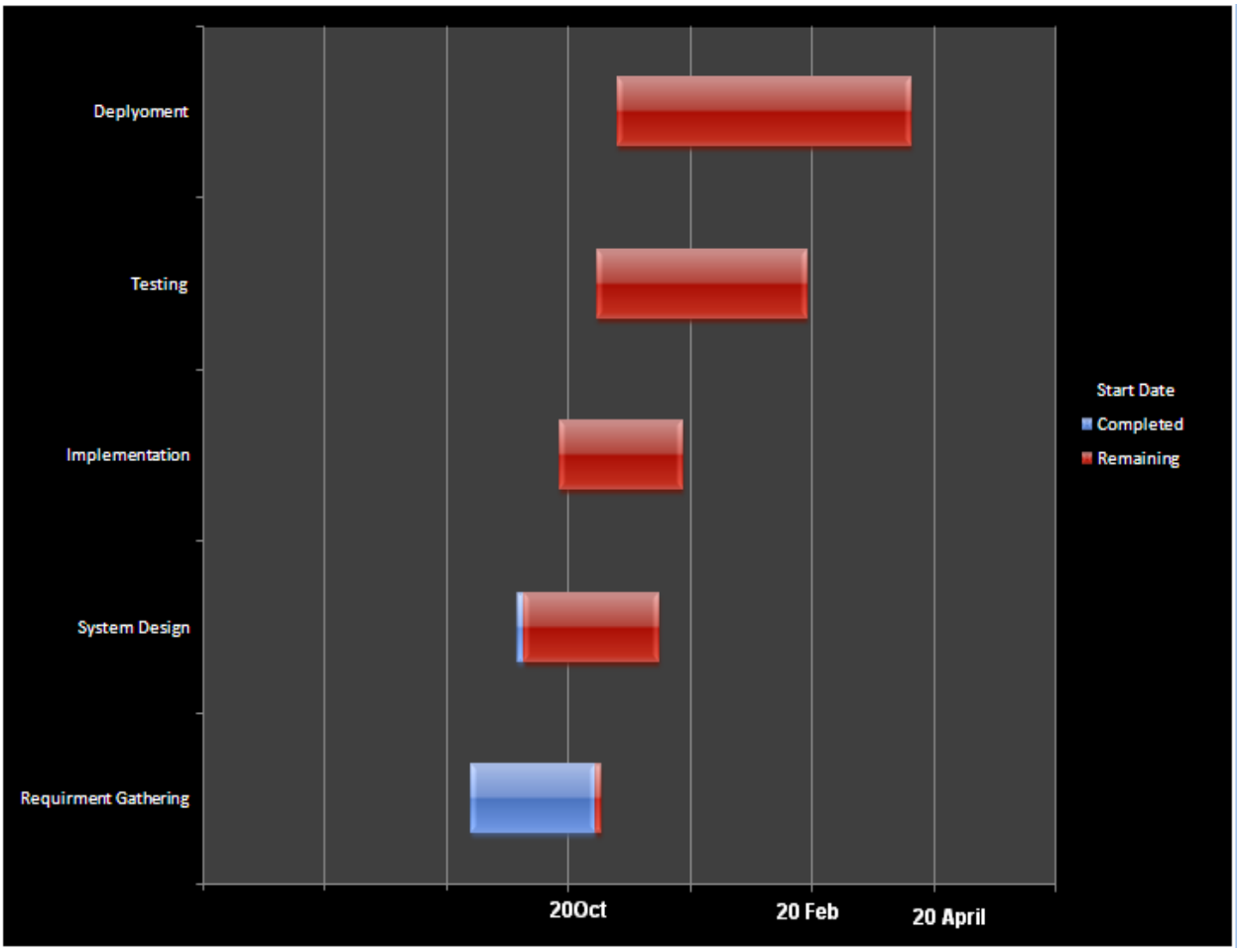
CHAPTER 7

PROJECT PLANNING AND SCHEDULING

7.1 TIMELINE CHART

Project milestones can be shown in a simple time line chart. While the chart doesn't look complicated, it can provide good amount of information on project progress in a simple and understandable chart.

Fig 7.1 Timeline chart



7.2 TASK SHEET

		AUGUST					SEPTEMBER		
		W 1	**W 2**	**W 3**	**W 4**	**W 1**	**W 1**	**W 2**	**W 3**
1	**REQUIREMENT GATHERING**								
a	**INFORMATION FROM INTERNET**								
b	**INFORMATION FROM BOOK**								
2	**REQUIREMENT ANALYSIS**								
a	**ANALYSIS OF INFORMATION RELATED TO OPENCV**								
b	**ANALYSIS OF INFORMATION RELATED TO HAAR CLASSIFICATION**								
c	**ANALYSIS OF INFORMATION RELATED TO FRAME GRABBER**								
d	**ANALYSIS OF NETBEANS IDE 6.9**								

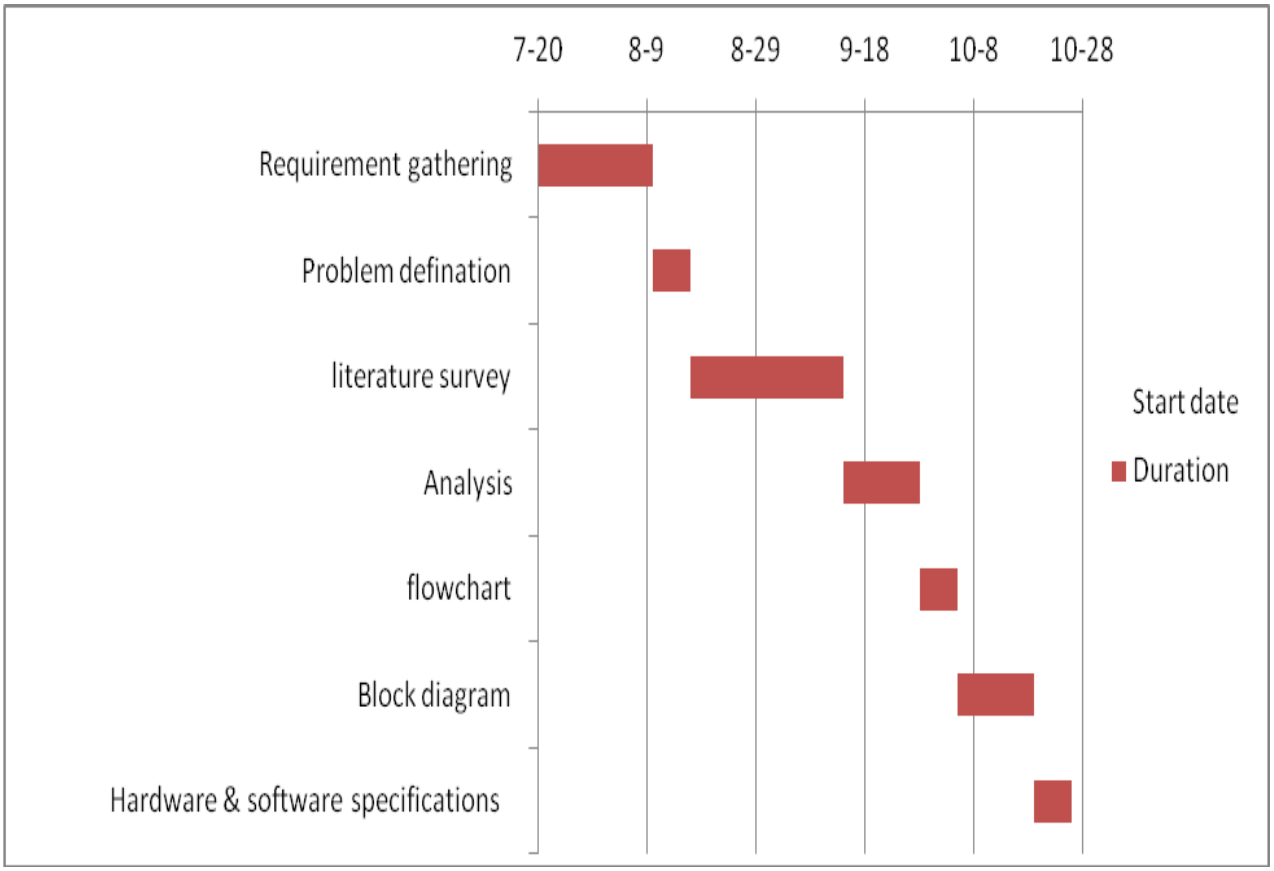
		OCTOBER				NOVEMBER			
		W 1	**W 2**	**W 3**	**W 4**	**W 1**	**W 2**	**W 3**	**W 4**
3	**PROBLEM DEFINITION**								
a	**MEET INTERNAL GUIDE**								
b	**IDENTIFY PROJECT CONSTRAINT**								
c	**ESTABLISH PROJECT STATEMENT**								
4	**FEASIBILITY**								
a	**ECONOMIC FEASIBILITY**								
b	**TECHNICAL FEASIBILITY**								
c	**BEHAVIORAL FEASIBILITY**								
5	**PLANNING**								
a	**SCHEDULING OF TASK**								
b	**TASK DIVISION AND TIME ALLOCATION**								

c	**EFFORT ALLOCATION**									
d	**RESOURCE ALLOCATION**									
6	**DESIGNING**									
a	**HAAR CLASSIFICATION LOGIC ARE STUDIED**									
b	**DESIGNING OF GUI**									
c	**TRAINING DATA**									
			FEBRUARY		**MARCH**				**APRIL**	
		W 1	**W 2**	**W 3**	**W 1**	**W 2**	**W 3**	**W 4**	**W1**	**W2**
7	**CODING**									
a	**CODING OF GUI**									
b	**CODING OF FACE AND EYE DETECTION**									
c	**CODING OF MOUSE CLASS**									
d	**CODING OF HEAD MOVEMENT**									

8	**IMPLEMENTATION DETAILS**									
a	**LINKING GUI AND PROJECT FILES**									
9	**TESTING**									
a	**UNIT TESTING**									
b	**INTEGRATION TESTING**									
c	**SYSTEM TESTING**									
10	**EVALUATION**									
a	**PROJECT EVALUATION**									
b	**DOCUMENTATION REVIEW AND RECOMMENDATION**									

Table 7.2 Task sheet

7.3 GNATT CHART



CHAPTER 8

CONCLUSION AND FUTURE WORK

We have implemented a system to access the mouse pointer on the computer screen using only Head and Eye movements. With the use of a camera and Java technology, the system architecture is prepared. User is able to view head and eye movements captured through the camera which is displayed on the screen, accordingly the user can move the mouse pointer as needed and also perform various mouse actions. The proposed system is feature based thus allowing any user to use the system without prior registration. This system is especially useful for the upper limb disabled.

Currently, we are extending our implementation to support keyboard press technology for the ease of the User to use the Keyboard hands free along with the already existing mouse movements provided by the system. This would then enable the User to access the computer owing to only facial features and movements without the use of traditional mouse and keyboard i.e Hands free system.

REFERENCES

BIBLOGRAPHY:

[1] P. Viola and M. Jones, "Computer Vision and Pattern Recognition" 2001

[2] T. Hutchinson, K. P. White Jr., W. N. Martin, K. C. Reichert, and L. A.Frey, "Human-computer interaction using eye-gaze input" 1989.

IEEE PAPERS

[1] D. G. Evans, R. Drew, and P. Blenkhorn, "Controlling mouse pointer position using an infrared head-operated joystick,"2000.

[2] M. Betke, J. Gips, and P. Fleming, "The Camera Mouse: Visual Tracking of Body Features to Provide Computer Access for People With Severe Disabilities," On Neural Systems And Rehabilitation Engineering, March 2002.

[3] M. Nabati and A. Behrad, "Camera Mouse Implementation Using 3D Head Pose Estimation by Monocular Video Camera and 2D to 3D Point and Line Correspondences," 2010 5th International Symposium on Telecommunications (IST'2010), 2010.

[4] Y. L. Chen, F. T. Tang, W. H. Chang, M. K. Wong, Y. Y. Shih, and T.S. Kuo, "The new design of an infrared controlled human-computer interface for the disabled," Dec. 1999.

WEBSITES

[1] www.disabled-world.com

[2] www.ncbi.nlm.nih.gov

[3] www.spd.org.sg