# FACIAL MOTION RECOGNITION USING CAMERA APPLICATION FOR MOVEMENT OF MOUSE (I-SIGHT MOUSE)

**HASRUL REEZA BIN MUSTAFFA**

**A report submitted in partial fulfillment of the requirement for the awarded of the Degree of Bachelor Computer Science (Software Engineering)**

**Faculty of Computer Systems & Software Engineering**

**Universiti Malaysia Pahang**

**APRIL 2010**

## ABSTRACT

i-Sight Mouse is developed to help detect and track facial movement to able to control mouse application by using machine vision which is webcam. Existing system has been used and still got a lot of work for improvement so that's why i-Sight Mouse is developed. It is used for people with disabilities that are no hands and need to interact with computer just using the movement of head and facial part. Using Extreme Programming for the software development model helped a lot in develops the system. Extreme Programming (XP) is a software engineering methodology , which is intended to improve software quality and responsiveness to changing customer requirements. As a type of agile software development, it advocates frequent "releases" in short development cycles (timeboxing), which is intended to improve productivity and introduce checkpoints where new customer requirements can be adopted. There are 2 modules in this system which is face recognition and face tracking. By implementing image-processing techniques the system is able to do the face recognition process and face tracking process. Face recognition is a module to capture video frame, enhancement the image and do extraction so that it can be verify either it is a face of human or not. Face tracking is based on predicting the place of the feature in the current frame based on its location in previous ones; template matching and some heuristics are applied to locate the feature's new coordinates. By tracking the eyes the system able to operate right clicks and mouse click operation as user blinking their right eye and left eye. The nose tip is tracked to use its movement and coordinates as the movement and coordinates of the mouse pointer.

# ABSTRAK

*i-Sight Mouse* dibangunkan untuk membantu mengesan dan menjejaki pergerakan wajah untuk mengendalikan aplikasi tetikus dengan menggunakan visi mesin yang dikenali sebagai webcam. Sistem yang sedia ada telah digunakan dan masih banyak kelemahab untuk tujuan penambahbaikan dan kerana itulah i-Sight Mouse dibangunkan. Hal ini digunakan untuk orang kurang upaya atau cacat yang tidak mempunyai tangan dan perlu berinteraksi dengan komputer hanya menggunakan pergerakan kepala dan bahagian wajah. Menggunakan *Extreme Programming* untuk model pembangunan perisian banyak membantu dalam pembangunan sistem. *Extreme Programming (XP)* adalah metodologi kejuruteraan perisian, yang bertujuan untuk meningkatkan kualiti perisian dan responsif terhadap perubahan keperluan pelanggan. Sebagai salah satu jenis pembangunan perisian yang pantas, ia menyokong hasil "keluaran" dalam kitaran pembangunan pendek (*timeboxing*), yang bertujuan untuk meningkatkan produktiviti dan memperkenalkan pos pemeriksaan di mana keperluan pelanggan baru dapat dipertimbangkan. Terdapat 2 modul iaitu pengenalan wajah dan mengikuti pergerakan wajah. Dengan menerapkan teknik *image processing* ia mampu melakukan proses pengenalan wajah dan menghadapi proses merekod pergerakan wajah. Pengenalan wajah adalah modul untuk menangkap video frame, peningkatan imej dan melakukan ekstraksi sehingga dapat mengesahkan wajah manusia atau tidak. Dengan kerlipan mata sistem dapat beroperasi operasi klik kanan dan klik tetikus sebagai user berkelip mata kanan dan mata kiri. Bucu hidung dikesan menggunakan gerakan dan koordinat sebagai gerakan dan koordinat dari penunjuk tetikus.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF APPENDICES

# ABBREVIATIONS

| | |
|---|---|
| BTE | - Between the Eyes |
| FMR-CAM | - Facial Motion Recognition using Camera Application for the Movement of Mouse |
| GUI | - Graphical User Interface |
| HCI | - Human Computer Interaction |
| IT | - Information Technology |
| JMF | - Java Media Framework |
| PC | - Personal Computer |
| RGB | - Red Green Blue |
| ROI | - Range of Interest |
| SDLC | - Software Development Life Cycle |
| SSD | - Sum Square Difference |
| SSR | - Six-Segmented Rectangular |
| SVM | - Support Vector Machine |
| UMP | - Universiti Malaysia Pahang |
| USB | - Universal Serial Bus |
| XP | - Extreme Programming |

# CHAPTER 1

## INTRODUCTION

In the past few years high technology has become more progressed, and less expensive. With the availability of high speed processors and inexpensive webcams, more and more people have become interested in real-time applications that involve image processing. One of the promising fields in artificial intelligence is Human Computer Interaction (HCI), which aims to use human features (e.g. face, hands) to interact with the computer [1].

One way to achieve that is to capture the desired feature with a webcam and monitor its action in order to translate it to some events that communicate with the computer. By developing i-Sight Mouse will help people who have hands disabilities that prevent them from using the mouse by designing an application that uses facial features (nose tip and eyes) to interact with the computer.

The nose tip was selected as the pointing device; the reason behind that decision is the location and shape of the nose; as it is located in the middle of the face it is more comfortable to use it as the feature that moves the mouse pointer and defines its coordinates, not to mention that it is located on the axis that the face rotates about, so it basically does not change its distinctive convex shape which makes it easier to track as the face moves. Eyes were used to simulate mouse clicks, so the user can fire their events as he blinks.

## 1.1 Problem Statement

There are people, often children or teenagers, who were born with physical disabilities (typically with Cerebral Palsy or other neurological disorders). These folks may never have learned to read and are not familiar with using a computer. They might not have any communication system at all. They might not understand cause and effect, that actions on their part can cause events in the world. The other kinds of disability people are adults who learned to read and use a computer but then lost their physical ability to use the computer because of an accident or stroke or degenerative disease.

So these people have their right to get the opportunity to have a normal life like having jobs, communicate with each other and also entertainment, and be treated like normal person.

## 1.2 Objective

The identified objectives that are to be archived by developing the i-Sight Mouse are listed:

1. To develop an application for helping people with disabilities to control the mouse pointer on a computer by moving their head and eyes blinking.
2. To implement image processing techniques for face recognition and face tracking.

## 1.3 Scope

The main user for i-Sight Mouse application is people who do not have reliable control of a hand but who can move their head. The system environment is using windows operating system, which fits with JAVA Runtime Environment (JRE) and JAVA Media Framework (JMF). To control the mouse pointer, various points were tracked ranging from the middle distance between the eyes, to the nose tip. In able to do that Face Detection Algorithms is been used. Six Segmented Rectangular (SSR) filter used to reduce the area in which focus looking for the face and the eyes. In order to facilitate the use of SSR filters an intermediate image representation called integral image will be used. With this representation calculating the sectors of the SSR filter becomes fast and easy. The restriction on using this i-Sight Mouse is the face position must be in the center of the screen and distance is about 35 cm from the screen. The light source must be control to frontal to the face.

## 1.4 Thesis Organization

This thesis consists of 6 chapters ranging from Chapter 1 until Chapter 6. Chapter 1 gives an overview of the study conducted. It also supply with the problem statement, objective and the scope of the study. Meanwhile, Chapter 2 reviews the previous research works that was conducted by other researches. All the relevant technical paper, journals, and books taken from those researches will be discussed in detail. Chapter 3 reveals the techniques and the algorithms that will be used in performing this study. It will discuss about the process flow in detail of this research. Details of the implementation of the study will be discussed in Chapter 4. Results of the testing are to be expounding in Chapter 5. Lastly, Chapter 6 concludes the entire thesis.

# CHAPTER 2

# LITERATURE REVIEW

## 2.1 Overview

The current evolution of computer technologies has enhanced various applications in human-computer interface. Face and gesture recognition is a part of this field, which can be applied in various applications such as in robotic, security system, drivers, monitor, and video coding system.

Since human face is a dynamic object and has a high degree of variability, various techniques have been proposed previously. Face detection techniques can be classified into two categories: feature based approach and image-based approach. The techniques in the first category makes used of apparent properties of face such as face geometry, skin color, and motion. Even feature-based technique can achieve high speed in face detection, but it also has problem in poor reliability under lighting condition [1].

For second category, the image based approach takes advantage of current advance in pattern recognition theory. Most of the image based approach applies a window scanning technique for detecting face, which requires large computation. Therefore, by using only image based approach is not suitable enough in real-time application.

## 2.2 Case Study on Existing System

There is a system that has been developed to capture head movement for the control of cursor movement to help people with disabilities use the computer. The system is named as Camera Mouse, which the original idea was developed, by Prof. Margrit Betke (then at Boston College, now at Boston University) and Prof. James Gips (Boston College). Camera Mouse works as a mouse replacement system for Windows computers so it should work with just about any application program. For example people use Camera Mouse with entertainment programs, education programs, communication programs, web browsers, and so on. Camera Mouse works best with application programs that require only a mouse and a left click and that do not have tiny targets. It's easier to use Camera Mouse with application programs that do not require extreme accuracy. Right now, this Camera Mouse system supported used in Windows Vista or Windows XP system and it needs a standard USB webcam [2].

The system involves two computers: the vision computer, which does the visual tracking, and the user's computer, which runs a special driver and any application software the user wishes to use.

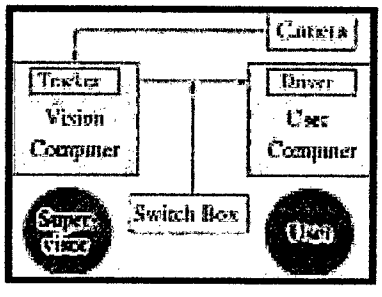


**Figure 2.1** The Camera Mouse system [3].

*The Vision Computer*

The vision computer is a 550 MHz Windows NT machine with a Matrox Meteor-II video capture board. The vision computer receives 30 frames per second from a Sony EVI-D30 camera mounted above or below the monitor of the user's computer. The image used is of size 320 by 240 pixels. The image sequence from the camera is displayed in a window on the vision computer by the visual tracking program [3].

Initially the operator uses the camera remote control to adjust the pan-tilt-zoom of the camera so that the person's face is centered in the image. The operator uses the mouse to click on a feature in the image to be tracked, perhaps the tip of the user's nose. The vision computer draws a green 15 by 15 pixels square centered on the point clicked and outputs the coordinates of the center of the square. These will be used for the mouse coordinates by the user's computer.

Thirty times per second the vision computer receives a new image from the camera and decides which 15 by 15 square sub image is closest to the previous selected square. The vision computer program examines 400 15 by 15 trial square sub images around the location of the previously selected square. The program calculates the normalized correlation coefficient $r(s,t)$ for the selected sub image $s$ from the previous frame with each trial sub image $t$ in the current frame [3].

$$r(s,t) = \frac{A\sum s(x,y)t(x,y) - \sum s(x,y)\sum t(x,y)}{\sigma_s \sigma_t}$$

where $A$ is the number of pixels in the sub image, namely 225, and

$$\sigma_s = \sqrt{A\sum s(x,y)^2 - (\sum s(x,y))^2} \quad \text{and} \quad \sigma_t = \sqrt{A\sum t(x,y)^2 - (\sum t(x,y))^2}$$

The trial sub image with the highest normalized correlation coefficient in the current frame is selected. The coordinates of the center of this sub image are sent to the user computer. The process is repeated for each frame. If the program completely loses the desired feature the operator can intervene and click on the feature in the image and that will become the center of the new selected sub image.

*The User's Computer*

The user's computer is a Windows 98 machine running a special driver program in the background. The driver program takes the coordinates sent from the vision computer, fits them to the current screen resolution, and then substitutes them for the mouse coordinates in the system. The driver program is based on software developed for the EagleEyes system; electrodes based system that allows for control of the mouse by changing the angle of the eyes in the head [3].

Any commercial or custom software can be run on the user's computer. The visual tracker acts as the mouse. The NumLock key is used to switch from the regular mouse to the visual tracker and back. The user moves the mouse pointer by moving his head (nose) or finger in space. The driver program contains adjustments for horizontal and vertical "gain." High gain causes small movements of the head to move the mouse pointer greater distances, though with less accuracy. Adjusting the gain is similar to adjusting the zoom on the camera, but not identical [3].

Many programs require mouse clicks to select items on the screen. The driver program can be set to generate mouse clicks based on "dwell time." With this feature, if the user keeps the mouse pointer within, typically, a 30 pixel radius for, typically, 0.5 second a mouse click is generated by the driver and received by the application program [3].

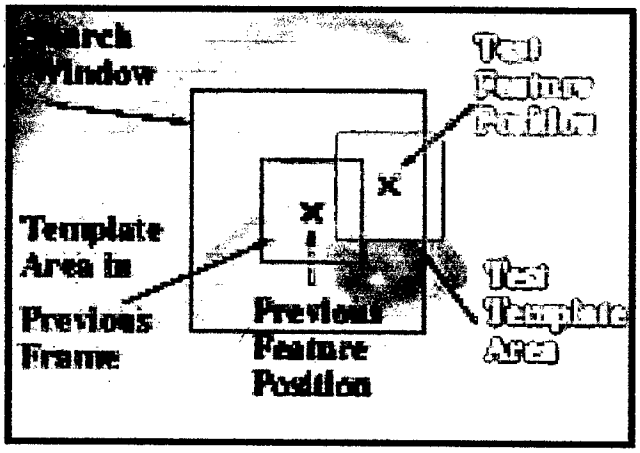


**Figure 2.2** The illustrates template and search window positions [3].

The tracking performance of the Camera Mouse is a function of template and search window sizes and the velocity of the feature's motion. It also depends on the choice of the feature, and the speed of the vision computer's processor. A large *search window* is useful for finding a feature that moves quickly. A large *template* size is beneficial because it provides a large sample size for determining the sample means and variances in the computation of the normalized correlation coefficient. Small templates are more likely to match with arbitrary background areas because they often do not have enough brightness variations, e.g., texture or lines, to be recognized as distinct features.

Large template or search window sizes require computational resources that may reduce the frame rate substantially. If many incoming frames are skipped, this means that the rate of the frames that are used for tracking drops well below 30 Hz, the constant brightness assumption may not hold for the tracked feature, even if it is still located within the search window. For the worse, when frames are skipped, it is likely that the feature moves outside the search window, far away from its previous position. To quantify tracking performance, a match between a template and the best-matching sub image within the search window is called *sufficient* if the normalized correlation

coefficient is at least 0.8. Correlation coefficients below 0.8 describe *insufficient matches*. Insufficient matches occur when the feature cannot be found in the search window because the user moved quickly or moved out of the camera's field of view [3].

This results in an undesired match with a feature that is different from the initially selected feature. For example, if the right eye is being tracked and the user turns his or her head quickly to the right, so that only the profile is seen, the right eye becomes occluded. A nearby feature, for example, the top of the nose, may then be cropped and tracked instead of the eye. The threshold of 0.8 was chosen after extensive experiments that resulted in an average correlation of 0.986 over 800 frames for a match between template and best correlated sub image, while the correlation for poor matches varied between 0.7 and 0.8. If the correlation coefficient is above 0.8, but considerably less than one, the initially selected feature may not be in the center of the template anymore and attention has "drifted" to another nearby feature. In this case, however, tracking performance is usually sufficient for the applications tested [3].

A variety of features were tested in an attempt to find body points that can be easily moved by the user and reliably tracked by the system. Since the appearance of body features differs among people, tracking performance varies between individuals. Several features, however, were reliably tracked across the test group. Figure 2.3 illustrates the tracking performance of several features, while Figure 2.4 illustrates the brightness variations of the template images.

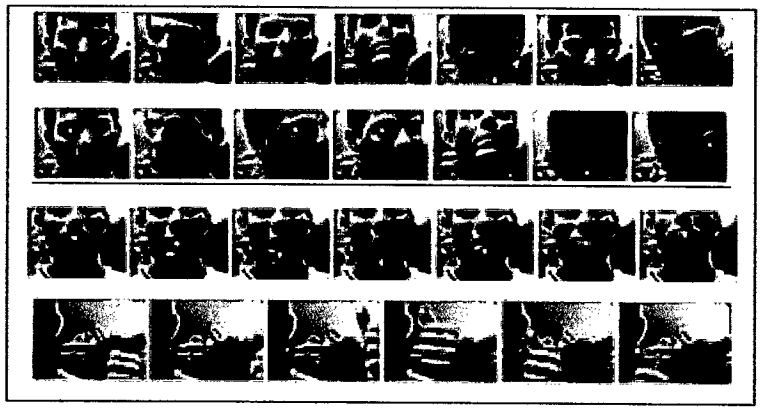

**Figure 2.3** The tracking performance of several features [3].

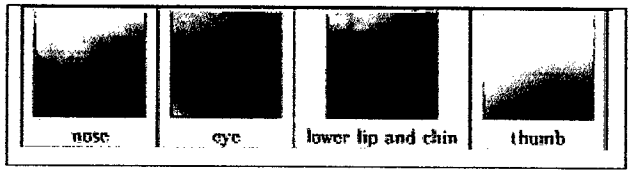


**Figure 2.4** The brightness variations of the template images [3].

*A. Nose Tracking*

The nose is a desirable tracking feature for several reasons. First, it is easy for a computer user to point his or her nose in a particular direction while watching the screen. The nose is essentially in the center of the face and does not become occluded when the user's head moves significantly. Second, the nose template tends to contain a good amount of brightness contrast to its surrounding features [3].

The majority of testing with the Camera Mouse was done under normal overhead lighting. In such an environment, the nose tends to be brighter than the rest of the face, as it is slanted and therefore oriented toward the overhead light. In image in Figure 2.3, the tip of the nose was tested as a feature. It was tracked throughout the 1000 recorded frames and not lost once. This excellent tracking performance can be

reproduced for arbitrarily long time periods as long as the user understands the constraints of the system and cooperates accordingly. If he or she moves so quickly that the nose tip leaves the search window, the system cannot track it. This can only occur with a very drastic motion, for example, a violent shaking of the head or an extremely jerky movement. The system also loses the nose feature if the user covers the nose or moves out of the camera's field of view. The drifting phenomenon may occur for some users. For example, the template may slowly drift up the bridge of the nose.

For such users, the bottom part of the nose, i.e., the area between the nostrils, is a more useful feature. It works better because the neighboring nostrils provide good contrast points and the shadow that is generally present under the nose distinguishes the bottom part of the nose from the cheeks or lips.

*B. Eye Tracking*

There has been some success in tracking the eye, but not to the extent of determining gaze direction. Image in Figure 2.3 shows the eye being tracked at various positions. Note that it is not the pupil but the whole eye that is being tracked. The brightness contrast between white eye sclera and dark iris and pupil, along with the texture of the eyelid, provides a distinctive template. Although the eye can be tracked well, it has not been used effectively with a Camera Mouse application because it is a relatively difficult feature to move while viewing the screen. Also, rotating the head may cause the eye to be blocked by the nose and not be visible at all [3].

*C. Lip Tracking*

As shown image in Figure 2.3, tracking the area of the lower lip and cleft has also been tested extensively. This feature can be tracked successfully on many individuals. It is a good tracking location because of the brightness difference between the lip and the cleft. People with large lips tend to have a shadow cast upon the cleft that enhances the brightness difference between the upper and lower image portions of the template and makes vertically drifting templates very unlikely [3].

Furthermore, the outline of the lip forms a curved line that helps control lateral drifting and keeps the template centered on the lips. The range of muscle control varies widely between people with severe disabilities, and head movement is not always an option. Opening and closing the mouth is a possibility for many people, though, and can allow cursor motion in either a vertical or a horizontal direction.

*D. Thumb Tracking*

To test other body features, not just facial features, the thumb was selected. Although it was tracked successfully, as shown in Figure 2.3, it has two main flaws as a tracking point. First, the camera has difficulties in focusing on it. As can be seen, the thumb takes up such a small portion of the screen that the camera's autofocus mechanism focuses on the objects in the background and not the thumb. This distorts the outline of the thumb and makes it difficult to track. Thus, if the thumb is used as a tracking point, it should be held close to the body or some other object in the background, so that the camera is able to focus on the thumb correctly. Another problem in using the thumb is its small surface area, which can move out of the search window easily. If this occurs, the tracking program will lose the thumb entirely and begin tracking a new point in the background. This means that if the thumb is used, slow movements are necessary [3].