VISION-BASED HUMAN-COMPUTER INTERACTION USING LASER POINTER

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

İBRAHİM AYKUT ERDEM

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

THE DEPARTMENT OF COMPUTER ENGINEERING

JULY 2003

Approval of the Graduate School of Natural and Applied Sciences.

Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Ayşe Kiper
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Volkan Atalay
Supervisor

Examining Committee Members

Prof. Dr. A. Enis Çetin

Assoc. Prof. Dr. Volkan Atalay

Assoc. Prof. Dr. Sibel Tarı

Assist. Prof. Dr. Uğur Güdükbay

Dr. Adem Yaşar Mülayim

ABSTRACT

VISION-BASED HUMAN-COMPUTER INTERACTION USING LASER POINTER

Erdem, İbrahim Aykut

M.Sc., Department of Computer Engineering

Supervisor: Assoc. Prof. Dr. Volkan Atalay

July 2003, 45 pages

By the availability of today's inexpensive powerful hardware, it becomes possible to design real-time computer vision systems even in personal computers. Therefore, computer vision becomes a powerful tool for human-computer interaction (HCI). In this study, three different vision-based HCI systems are described. As in all vision-based HCI systems, the developed systems requires a camera (a webcam) to monitor the actions of the users. For pointing tasks, laser pointer is used as the pointing device.

The first system is Vision-Based Keyboard System. In this system, the keyboard is a passive device. Therefore, it can be made up of any material having a keyboard layout image. The web camera is placed to see the entire keyboard image and captures the movement of the laser beam. The user enters a character to the computer by covering the corresponding character region in the keyboard layout image with the laser pointer. Additionally, this keyboard system can be easily adapted for disabled people who have little or no control of their hands to use a keyboard. The disabled user can attach a laser pointer to an eyeglass

and control the beam of the laser pointer by only moving his/her head. For the same class of disabled people, Vision-Based Mouse System is also developed. By using the same setup used in the previous keyboard system, this system provides the users to control mouse cursor and actions. The last system is Vision-Based Continuous Graffiti[1]-like Text Entry System. The user sketches characters in a Graffiti^{TM}-like alphabet in a continuous manner on a flat surface using a laser pointer. The beam of the laser pointer is tracked during the image sequences captured by a camera and the corresponding written word is recognized from the extracted trace of the laser beam.

Keywords: human-computer interaction, computer vision.

[1] Graffiti is a registered trademark of Palm, Inc.

ÖZ

LAZER İŞARETÇİSİ KULLANARAK BİLGİSAYARLI GÖRMEYE DAYALI İNSAN-MAKİNE ETKİLEŞİMİ

Erdem, İbrahim Aykut

Yüksek Lisans, Bilgisayar Mühendisliği Bölümü

Tez Yöneticisi: Assoc. Prof. Dr. Volkan Atalay

Temmuz 2003, 45 sayfa

Günümüzdeki ucuz ve güçlü donanımlar sayesinde, kişisel bilgisayarlarda bile eşzamanlı çalışabilen bilgisayarlı görmeye dayalı sistemlerin gelistirilmesi mümkün olmuştur. Böylece bilgisayarlı görme, insan-makine etkileşiminde (İME) güçlü bir araç haline gelmiştir. Bu çalışmada bilgisayarlı görmeye dayalı üç farklı İME sistemi anlatılmaktadır. Tüm bilgisayarlı görmeye dayalı sistemlerde olduğu gibi bu geliştirilen sistemler de kullanıcıların hareketlerini izlemek için bir kameraya (web kamerası) ihtiyaç duymaktadırlar. İşaretleme işlemleri için de lazer işaretçisi kullanılmıştır.

Birinci sistem Bilgisayarlı Görmeye Dayalı Klavye Sistemi'dir. Bu sistemde klavye pasif bir aygıttır, böylece klavye karakter dizilimini içeren imgenin basılı olduğu herhangi bir maddeden yapılabilir. Web kamerası tüm klavye imgesini görecek şekilde yerleştirilir ve lazer işaretçisinin izini izler. Kullanıcı bilgisayara karakter girdisi yapmak istediğinde klavye diziliminde o karaktere karşılık gelen bölgeyi lazer işaretçisinin iziyle kapar. Ek olarak, bu sistem normal bir klavyeyi ya hiç ya da çok az kullanabilen engelli kullanıcılara uyarlanabilir. Engelli kullanıcı

bir lazer işaretçisinin iliştirilmiş olduğu bir gözlüğü takarak lazer işaretçisini kafa hareketleriyle kontrol edebilmektedir. Benzer engelli kullanıcıların kullanımına yönelik Bilgisayarlı Görmeye Dayalı Fare Sistemi de geliştirilmiştir. Bu sistemde de klavye sistemiyle benzer bir kurulum kullanılarak engelli kullanıcıların fare imlecini kontrol etmesi sağlanmıştır. Son sistem ise Bilgisayarlı Görmeye Dayalı Devamlı Graffiti^{TM}Benzeri Metin Girdi sistemidir. Kullanıcı, Graffiti^{TM}benzeri alfabedeki karakterleri devamlı bir şekilde düz bir yüzeye yazar. Lazer işaretçisinin izi, kamera tarafından yakalanan imge serisi boyunca izlenir ve yazılmak istenen kelime çıkarılan lazer işaretçisinin izlerinden tanımlanır.

Anahtar Kelimeler: insan makine etkileşimi, bilgisayarla görme.

To my family

ACKNOWLEDGMENTS

I would like to thank to Volkan Atalay and A. Enis Çetin for their guidance and support throughout this study. I would also like to thank my family for their encouragement and assistance.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

Human-computer interaction (HCI) is the study of interactions between people and computers. It is an interdisciplinary field, computer side includes computer science, engineering and ergonomics; human side includes psychology, physiology, sociology and cognitive sciences. HCI concerns with design, evaluation and implementation of systems for human use. It mainly concerns with the problem of usability of the systems. Nielsen [2] defines five criteria of usability as follows:

- Learnability: The system should be easy to learn so that the user can rapidly start getting some work done with it.
- Efficiency: The system should be efficient to use, so that once the user has learned the system, a high level of productivity is possible.
- Memorability: The system should be easy to remember, so that the casual user is able to return to the system after some period of not having used it, without having to learn everything all over again.
- Errors: The system should have a low error rate, so that users make few errors during the use of the system, and so that if they do make errors they can easily recover from them. Further, catastrophic errors must not occur.

- Satisfaction: The system should be pleasant to use, so that users are subjectively satisfied when using it; this means that they like it.

As the role of computers in our daily life increases, it is expected that many computer systems will be embedded into our environment. Since today's interfaces in HCI systems are far from ideal according to the criteria discussed, there is a need for new types of HCI with interfaces that are more natural and easy to use. With the growth of inexpensive powerful hardware, it becomes possible to design real-time computer vision systems even in personal computers. Therefore, computer vision becomes an alternative solution to new types of HCI systems.

The basic elements of vision-based interaction systems are environment, user and camera. User actions result changes in the environment and the corresponding action is recognized by the computer by examining any change in the camera image. Therefore detection, identification and tracking techniques in computer vision can be used in HCI systems. Users can employ various pointing device such as some real world objects like pen or his/her body parts like fingers, head, etc. Even the activities of users like walking, typing, hand/head gestures, etc. can be seen as user actions.

One of the first examples of vision-based HCI systems is Wellner's concept of DigitalDesk [3, 4]. It uses a couple of a camera and a video projector to form an augmented reality environment. Users can interact with papers on the desk with bare fingers or pens and all the data can be stored in digital form. Although it requires a highly constrained environment, it shows that visual input provides a natural way for interacting with computers. Following DigitalDesk, several different projection systems have also been developed. As an example Crowley, *et al.* developed a system called MagicBoard [5] where an ordinary white board is turned out to be a digital one. All markings on the board are also kept in digital form and some high level operations like copying, moving, zooming, etc. are possible by projecting these changes on the board via a projector.

There are also several studies related to vision-based new input devices. In the system named VisualPanel [6], an ordinary piece of paper can be used as

virtual mouse, keyboard and a 3D controller. The user can use his/her fingers or other tip pointers to enter text and performs some action like clicking, dragging, etc. In another system called FingerMouse [7], the finger of the user is tracked by a camera mounted above the keyboard, and mouse cursor is moved accordingly.

In the case of using some devices such as electronic tablets or digitizers to obtain the handwriting data, the pen trajectories can also be tracked by inspecting the visual data. These systems are composed of only a camera and a pointing device such as a pen. In a work by Munich and Perona [8], such a vision-based method is proposed for obtaining on-line handwriting data. Later, this system is extended by a word recognition module in [9]. Alternatively in [10], a vision-based system for recognizing isolated characters is introduced. The characters are drawn by a stylus or a laser pointer on a flat surface or the forearm of a person and the user's actions are captured by a head mounted camera. The beam of the laser pointer is detected in each camera image and the corresponding characters are recognized from these traces of the laser beam.

Additionally, visual inputs are very suitable in the systems developed for the handicapped. For example in [11], American sign language is recognized from video input sequence in real-time. IBM also developed vision-based TouchFree Switch [12] for disabled people where the switch can be activated by a choice of large or small body movements. Infrared cameras can also be used to track head gestures and eye gazes. For example, Zhai et. al. developed a eye tracker system called MAGIC [13]. In addition, some commercial products are available in the market. The examples are Tracker-2000 by Madentec [14], Head-mouse by Origin Instruments Corporation [15] and ViewPoint EyeTracker by Arrington Research [16]. Since these systems requires special hardware devices, their costs are around $2000 per system.

1.2 Purpose and Improvements Achieved

In this study, we present three different vision-based HCI systems. The setups for all systems include only a camera and an ordinary laser pointer as the pointing

device. The first system, Vision-based Keyboard System is designed for wearable computers and personal digital assistants (PDAs). The system needs a keyboard layout image that can be printed on any material and the user can enter a character by controlling the laser pointer. Alternatively, this keyboard system can be easily adapted for disabled people who has little or no control of their hands to use a standard mouse or a keyboard. As a companion system, Vision-Based Mouse System is developed for the same class of disabled people where the system provides the users to control the mouse cursor. The main advantage of the developed system for the handicapped is that the systems requires only a USB webcam, and an ordinary laser pointer. Therefore, the overall cost is very low compared to similar products. The last system that we developed is Continuous Graffiti-like Script Based Text Entry System. This is an extension of the text entry system described in [10]. This time, instead of drawing isolated characters the user sketches the Graffiti alphabet in a continuous manner on his or her left arm using a pointer or a stylus or a finger.

1.3 Organization of the Thesis

The organization of the thesis is as follows. In Chapters 2,3,4, the proposed vision-based HCI systems, Vision-Based Keyboard System, Vision-Based Mouse System and Vision-Based Continuous Graffiti^{TM}-like Text Entry System, are described, respectively. The thesis concludes with Chapter 5 in which the presented study is discussed and the future work is stated.

CHAPTER 2

VISION-BASED KEYBOARD SYSTEM

2.1 Introduction

Unistroke keyboards provide a good trade off between 10-finger typing and continuous handwriting recognition. In this study, we present a computer vision-based unistroke keyboard. The concept of computer vision-based regular QWERTY type keyboards is independently proposed by us and Zhang et.al. [6]. Also, another keyboard system is developed by Tomasi *et al.* [17] which is based a different technology. In this sytem, the keyboard image is projected on a smooth surface and special-purpose sensors are used to locate the user's fingers in 3-D space and track the intended keystrokes.

Our vision-based keyboard system is based on a soft keyboard system called CIRRIN (CIRculaR INput device) developed by Mankoff *et al.* [18]. CIRRIN was designed for tablet computers and the user draws one stroke per word on a touch-sensitive screen. Key locations are placed circularly and to enter a word the user traces out a path determined by the characters forming the word using a stylus. This layout structure can be seen in Figure 2.1. Whenever the stylus enters a key location the corresponding character is recognized by the tablet computer.

The proposed keyboard system is based on computer vision and it is especially designed for wearable computers and PDAs. In this approach, the keyboard is a

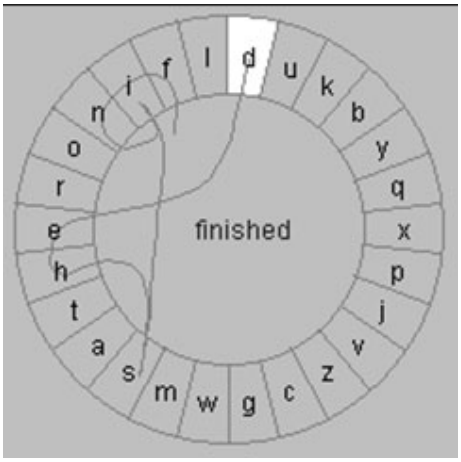


Figure 2.1: The circular layout of CIRRIN. The word "finished" is entered as an example (reproduced from [18]).

passive device. Therefore, it can be made out of paper, fabric or foldable plastic having an image of a QWERTY or other type of regular size keyboard. The keyboard layout image can even be displayed on a computer screen. Instead of circular shape of CIRRIN, our keyboard layout consists of three rectangles each divided into equally sized character regions and these rectangles are placed as up side down U like shape as shown in Figure 2.2. In our experiments we use a paper containing an image of this layout design as the keyboard. A webcam is placed in a way that the entire keyboard layout image can be seen in the camera image. But, the main principle is same as in CIRRIN, i.e. the user enters a character to the computer by covering the corresponding character region with a laser pointer. The actions of the user are captured by the camera and the corresponding covered key is recognized. The user can place his or her hand in the middle which elliminates the occlusion problem. In this way the camera can be placed on the users' forehead or shirt pocket.

Alternatively, this keyboard system can be easily adapted for disabled people with Quadriplegia, Cerebral Palsy, Multiple Sclerosis, Muscular Dystrophy, ALS, Carpal Tunnel Syndrome and any other disability where the user has little or no control of their hands to use a standard mouse or a keyboard. Speech recognition based systems may partially solve the text entry problem for disabled people

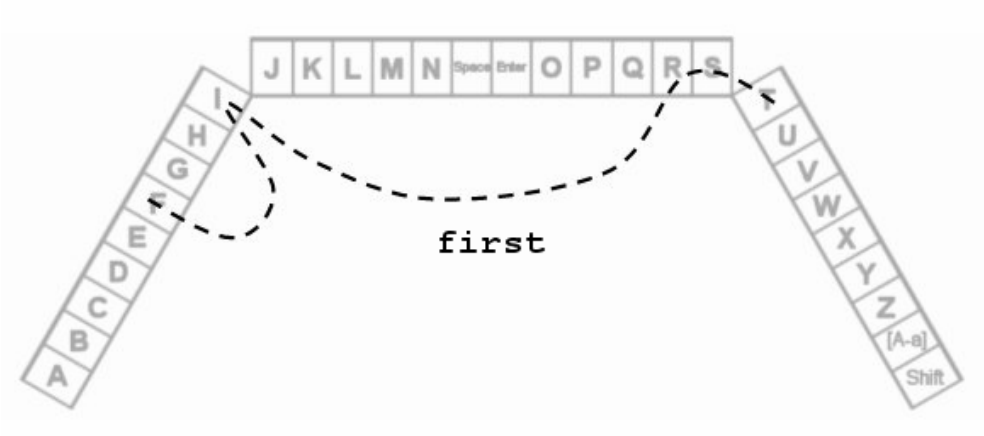


Figure 2.2: Up side down U shaped keyboard. The word "first" is entered as an example.

in some languages. However, in many agglunative languages including Turkish spoken by more than 200 million people there are no large vocabulary speech recognition systems.

In this keyboard version adapted for the handicapped, the keyboard image is displayed on the lower part of the screen and a USB web camera is placed in front of the monitor to capture the movement of the laser beam as shown in Figure 2.3. The user wears an eyeglass to which a laser pointer is attached and the user controls the beam of this laser pointer by moving his or her head as demonstrated in Figure 2.4. In this way, an inexpensive system is realized for entering text to a computer because there is no need to have a special purpose hardware to realize this keyboard system which is implemented in software. Whereas in commercial head-mouse and keyboard systems, an optical sensor is used to track a small reflective target dot worn on one's forehead or eyeglasses [19]. This special purpose sensor increases the cost of the system. On the other hand the cost of an ordinary red laser pointer and a USB web camera is about 50 US dollars.

Major functional parts of the image and video analysis algorithm include calibration, and tracking of the beam of the laser pointer in the camera images.

Figure 2.3: Keyboard system developed for the handicapped.

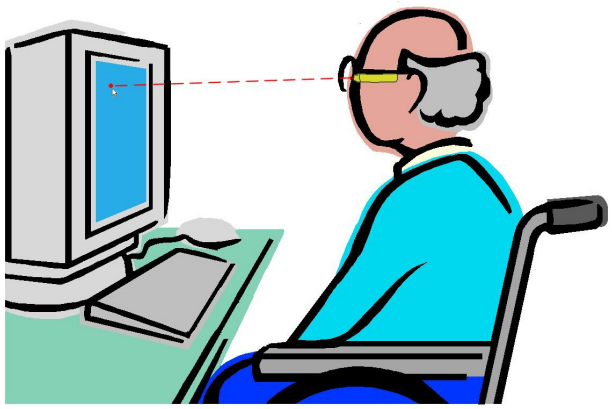


Figure 2.4: User wears an eyeglass to which a laser pointer is attached.

2.2 Calibration

In this system, the camera is positioned so that the entire keyboard image can be viewed. As it can be observed in Figure 2.2 and Figure 2.3, the keyboard consists of three rectangles, each divided into equally sized key regions. In this step, the aim is to determine the height and line equations of three rectangular segments. These line segments are shown as l_{AB}, l_{BC} and l_{CD} in Figure 2.5. As it can be observed in the figure, these line segment equations can be easily determined if the 4 lower corner positions are known. The height of a rectangular segment can also be determined by the facts that each key region is equally spaced and the ratio between width and height of a key region in the keyboard image is

known. The width of a key region in the camera image can be found by dividing the distance between the corner points with the total number of key regions in the corresponding rectangular segment. Once these line equations are found, the boundaries of each key region can be estimated since we assume that the camera is placed in a way that the view is nearly orthographic. For the two different version of keyboard systems, only finding these corner positions differs.

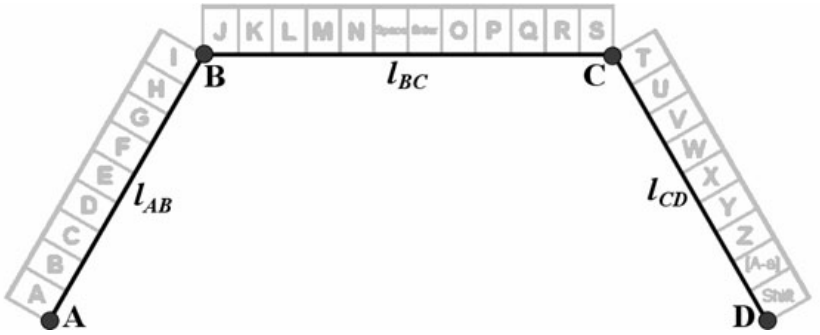


Figure 2.5: Three line segments for each rectangle forming the keyboard layout.

In the first keyboard system, these points can be either specified by the user by clicking in the camera image, or detected automatically by placing colored markers in these corner points and then applying color thresholding.

For the system designed for the handicapped, since we control the images on the screen, detecting these points can be easily performed automatically. When the system is turned on, at first a blank image having black background is displayed. Then the corner points are displayed on this image, and by taking the difference of these two images, the positions of these points can be easily determined.

2.3 Tracking

2.3.1 Finding position of the beam of the laser pointer

The beam of the laser pointer is determined by detecting the moving pixels in the current image of the video and from the color information. Moving pixels are estimated by taking the image difference of two consecutive image frames. In Figure 2.6(a) and (b), a sample original camera image and its corresponding

binary difference image is shown. Then by using the fact that the beam of the laser pointer is brighter than its neighbor pixels, the tracking process can be performed in a robust way. By calculating the center of the mass of the bright red pixels among the moving pixels, the position of the beam of the laser pointer is determined. The overall process is shown in Algorithm 1.

Algorithm 1 Finding the position of the beam of the laser pointer.

Given two consecutive camera images I_j and I_{j-1} and lines l_{AB}, l_{BC} and l_{CD},

(i) Determine the binary difference image I_{diff} between I_j and I_{j-1}.

(ii) By masking I_{diff} over I_j, form the image I_{mask}.

(iii) Determine maximum intensity value i_{max} over the pixels in I_{mask}.

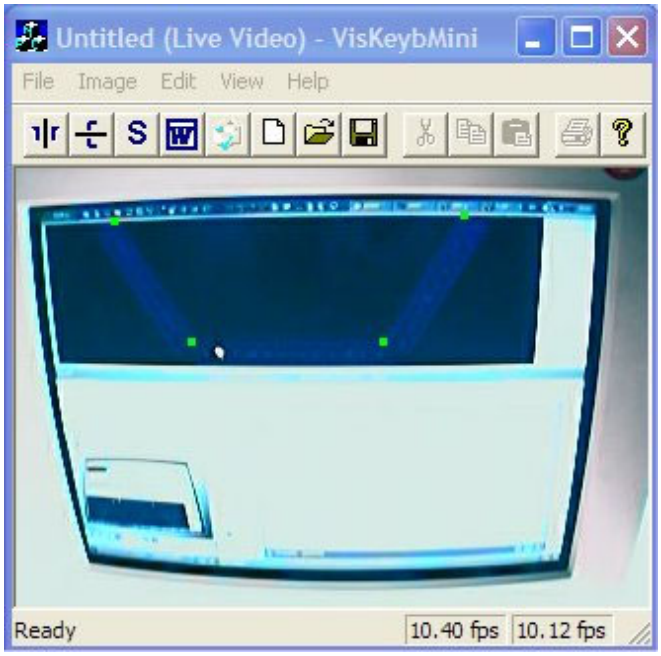
(iv) Set the intensity threshold t to $0.9 \times i_{max}$.

(v) For all pixels p_j where $i_{p_j} > t$, calculate the position of the beam of the laser pointer by taking center of mass as follows:

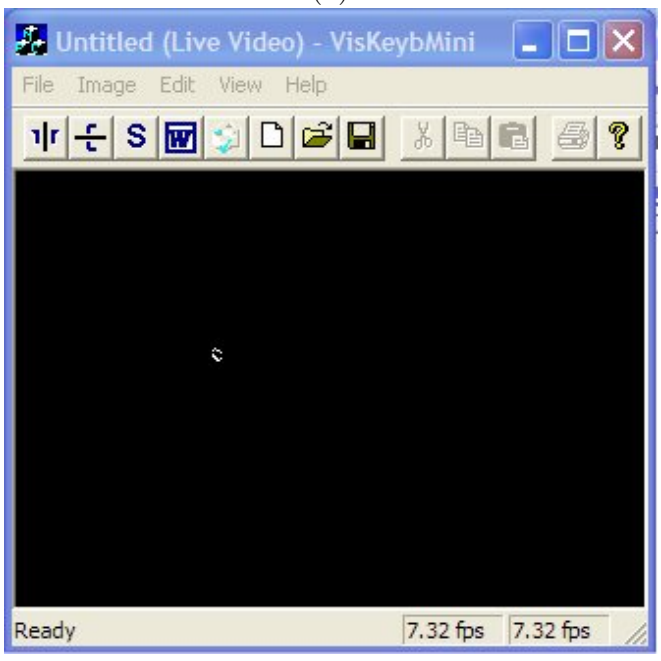
$$c_x = \frac{1}{n}\sum_{j=0}^{n} p_{j_x}, \quad c_y = \frac{1}{n}\sum_{j=0}^{n} p_{j_y}$$

2.3.2 Determining which character is to be written

A character is entered to the computer when the beam of the laser pointer enters the corresponding key region. To determine whether the beam of the laser pointer is in one of rectangular segments of keyboard, the relative position and distance of the beam to the line segments are examined. Then if the beam is over a key region, the position of the beam is projected on to the corresponding line segment and a final decision is reached by inspecting the position found after projection with respect to each key region. The beam must stay in that character region for at least two image frames for recognition. If this is the case, a key down message is sent to the computer. The overall process is shown in Algorithm 2. In Figure 2.7, the character "K" is entered to the computer. This image is obtained by overlapping a sequence of images. Small white squares indicate the location of the beam in past images of the video.



(a)



(b)

Figure 2.6: (a) Original camera image. Four lower corners are marked with squares. (b) Corresponding binary difference image.

2.4 Evaluation

The experimental system setup is shown in Figure 2.3. An ordinary web camera is placed in a place that the keyboard layout image can be seen in the camera

Algorithm 2 Determining which character is entered to the system.

Given the position of the beam of the laser pointer (c_x, c_y) and rectangular segments,

(i) Test whether (c_x, c_y) is in one of the rectangular segments:

1. $l(c_x, c_y) \geq 0$, i.e. (c_x, c_y) must be above the line l.

2. The distance $D((c_x, c_y), l) \leq h$ where h is the height of the rectangular segment and l is the corresponding line.

(ii) If (c_x, c_y) is in one of the rectangular segments,

Project (c_x, c_y) onto line l.

Determine which key is entered by inspecting the position of the projection point.

(iii) If the key is same as the one found in the previous frame,

Send a KEY_DOWN(key) message to the computer.

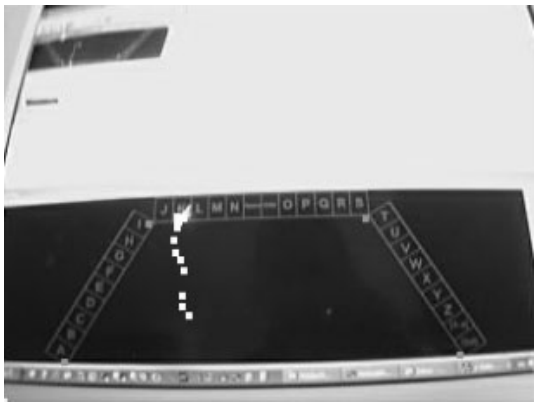


Figure 2.7: Character "K" is entered. Small white squares indicate the location of the beam in the past images.

image and the user enters text to the system by controlling the beam of a red laser pointer. The USB camera produces 320 pixel by 240 pixel color images at about 13 frames per second. All of the processing of the vision-based keyboard system is carried out in real time on a Pentium IV- 1.7Ghz processor with 512 MB memory. The keyboard system is developed using the Microsoft Vision SDK library. This library is a free vision library for Microsoft Operating Systems developed by the

Vision Technology Research Group in Microsoft Research.

In order to estimate the performance of the system, we use a test set of 50 words where these words are written by different users at least 15 times using our keyboard system. According to the experiments, we come up with an average recognition rate of 94% and the text entry speed is about 8 words per minute (wpm).

In addition, when we examine recognition rate versus word length which is given as a graph in Figure 2.8, we can infer that there is no direct relation of word length with the recognition rate. We can conclude that recogition rate is mainly related with the key strokes between each character of the written word. Furthermore, mean completion time of written word versus word length graph as shown in Figure 2.9 shows that writing time increases linearly with the increase in word length.

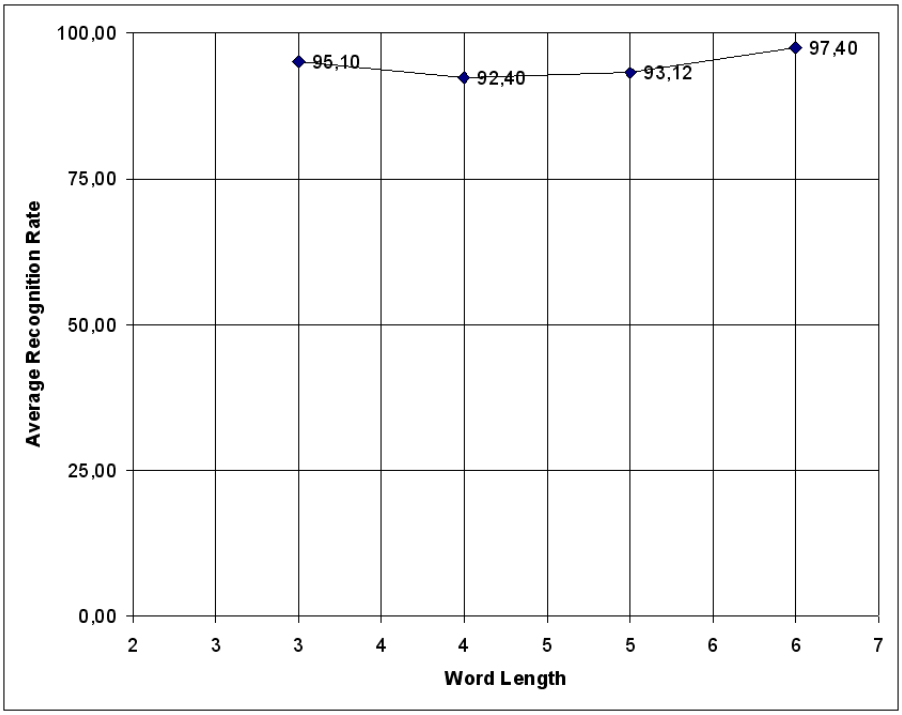


Figure 2.8: Recognition rate versus word length.

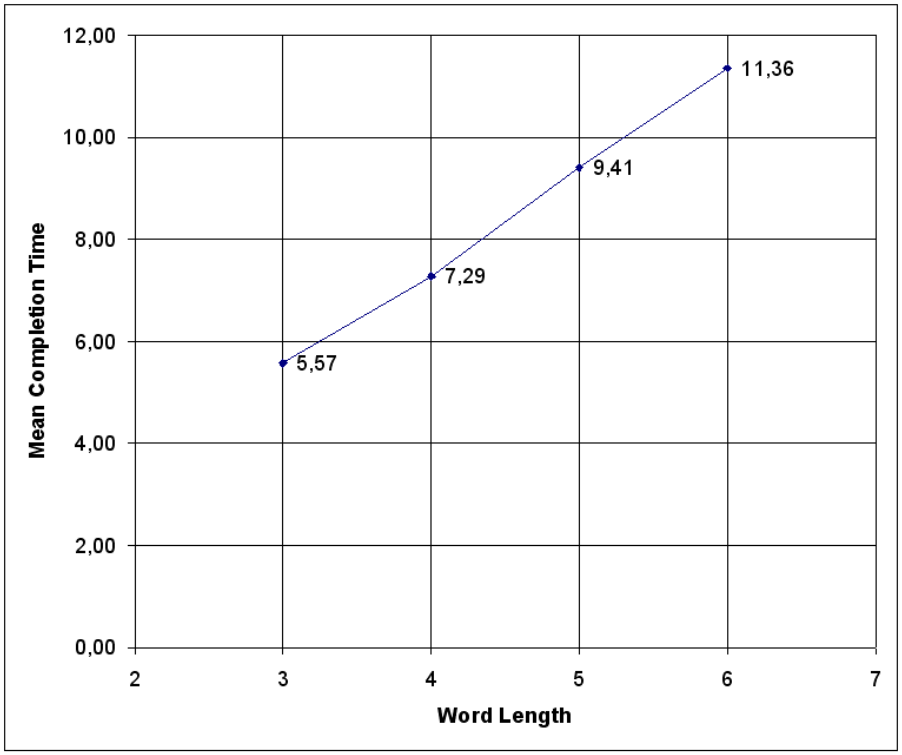


Figure 2.9: Mean completion time versus word length.

CHAPTER 3

VISION-BASED MOUSE SYSTEM

Our vision-based mouse system can be seen as a companion system to the keyboard system designed for the handicapped described in Chapter 2. This system also requires a laser pointer attached to an eye glass and the user can control the mouse cursor by his or her head movements. In this system, clicking action can be either performed by using switch devices developed for the handicapped or by holding the laser pointer on a specific location for a while and just like the keyboard sytem, this system also requires calibration and tracking phases.

3.1 Calibration

Calibration involves the detection of the screen region in the camera image, which means to determine the projective mapping between the image space and the screen space. Calibration is necessary since we need to find a transformation function which maps the position of beam of the laser pointer detected in the camera image to the exact position in screen coordinates. Our calibration step is very similar to the one described in [20]. This projective mapping can be described by a homography matrix (a 3x3 matrix) as in equation 3.1.

$$x_{screen} = \mathbf{H} x_{image} \tag{3.1}$$

In theory, the elements of this 3x3 matrix can be determined by using 4

pairs of camera image - screen correspondence points. However, we need more corresponding point pairs to find an accurate homography matrix. Therefore instead of using only 4 pairs, a calibration pattern consists of grid of circles are used. The calibration pattern can be seen in Figure 3.1.

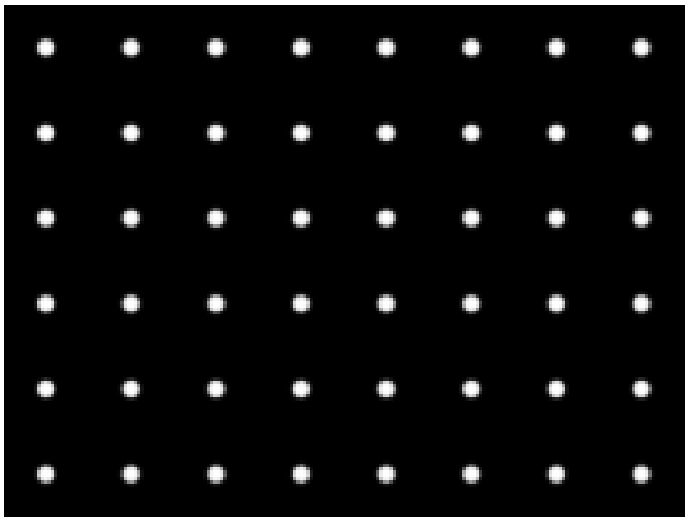


Figure 3.1: The pattern used in the calibration phase.

Image - screen correspondence point pairs can be easily obtained since the position of circle centers are known a priori and the position of circle centers can be easily determined in the camera image by using a search window strategy. To determine the size of the search window, firstly a simple pattern having only four circles at the far corners of the screen is displayed and these points are detected accordingly. This second pattern is shown in Figure 3.2. Then according to the distances between these points, the size of search window is estimated and the circles in the actual calibration pattern are found by moving around this search window in the camera image. An example result of this process can be seen in Figure 3.3. Once these correspondence point pairs are known, the homography matrix can be determined by using Direct Linear Transformation (DLT) Algorithm. The details of this algorithm are given in Appendix B.

In calculating the homograpy matrix, the assumption is that grid points lie on the same plane. But since most monitors are not flat, this is not the case, therefore we need to make some extra error correction. The circles on the calibra-

Figure 3.2: Second pattern used to determine the size of search window.

tion pattern are mapped back to the screen by using the computed homography matrix and the error vectors for each grid point are determined by subtracting approximate values from the real one. These error vectors are stored for the tracking phase. The use of this error correction is discussed in the tracking phase. The overall calibration process is shown in Algorithm 3.

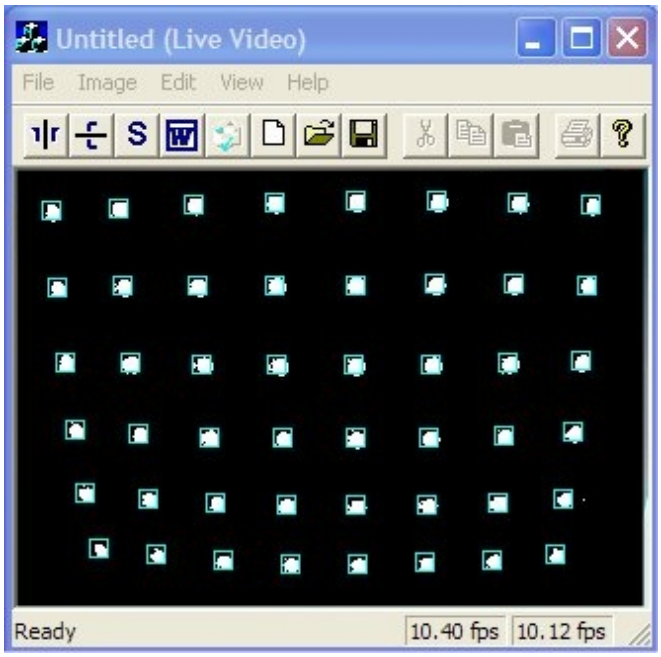


Figure 3.3: Detected circles in the calibration pattern.

Algorithm 3 The Calibration Phase of Vision-Based Mouse System

(i) Display a blank black image on the whole screen and store the corresponding camera image I_1.

(ii) Display the pattern image having only four circles at the far corners of the screen and store the corresponding camera image I_2.

(iii) Find the centers of the circles by from the difference image between I_1 and I_2.

(iv) Determine the size of the search window from the distances between corner points.

(v) Display the actual calibration pattern on the whole screen and store the corresponding camera image I_3.

(vi) Determine the centers of each circle by moving the search window around the difference image between I_1 and I_3.

(vii) Determine the camera image - screen correspondence points.

(viii) Compute the homography matrix H by using DLT Algorithm.

(ix) Calculate the approximate centers of the circles in the screen space by using H as in the equation 3.1.

(x) Determine the residual vectors for each circle and store these vectors.

3.2 Tracking

3.2.1 Finding position of the beam of the laser pointer

The position of the laser beam in the camera images are found exactly the same way as described for the keyboard system in Section 2.3.1. In addition, to make tracking process more reliable, "High Constrast" is selected from "Accessibility Options" menu which is a standard feature available in all Windows^{TM}operating systems.

3.2.2 Updating mouse cursor position

Once the position of the laser beam is determined, it is mapped to the screen coordinates using the homography matrix obtained in calibration phase. Then the error correction vector for that point is calculated using the bilinear interpolation [21] (see Appendix A) with the previously stored error vectors of the nearest four grid points and the correct screen position is calculated by adding this vector to the precomputed screen position. At the end the position of the mouse cursor is updated accordingly by sending a system message to the operating system. The tracking phase can be summarized as in Algorithm 4.

Algorithm 4 Updating the mouse cursor position.

(i) Determine the position P of the beam of the laser pointer in the camera image using Algorithm 1.

(ii) Map P to the screen space by using the homography matrix using Equation 3.1.

(iii) Calculate the corresponding error correction vector by using bilinear interpolation among the residual vectors of the nearest four circles in the grid.

(iv) Compute the actual screen position by adding the error correction vector to the precomputed approximate screen position.

(v) Update the mouse cursor position accordingly.

3.3 Evaluation

The system is examined according to two different aspects. First the performance of the system is tested with standard Fitts' pointing task and second the success of calibration phase is inspected.

3.3.1 Fitts' Law

Fitts' Law is a psychological model of human movement proposed in 1954 [1]. The model is actually formulated from an information-theoretic perspective based on the ideas from Shannon and Weaver [22]. Fitts' Law simply states that the time

to acquire a target is a function of the distance to and size of the target along a single dimension. This is expressed in the following form:

$$T = a + b\, log_2(\frac{A}{W} + 1) \tag{3.2}$$

where T is the acquisition time of a target of width W at distance A from the starting point and a and b are empirically determined constants. The logarithmic term is called as the index of difficulty (ID) and the reciprocal of b is referred as the index of performance (IP).

In the original work by Fitts [1], a reciprocal tapping apparatus shown in Figure 3.4 was used to investigate how response time is affected by the distance moved and by the size of the target. In the experiments, the subjects were tapping the metal plate with stylus and the task was to hit the center plate in each group alternately without touching either side plate.

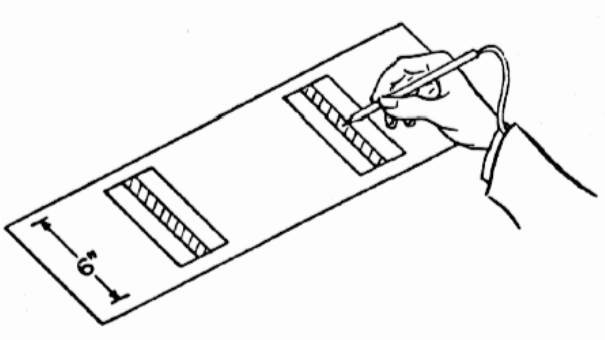


Figure 3.4: The reciprocal tapping apparatus used in the experiments by Fitts (reproduced from [1]).

Since Fitts' Law is proven to be an accurate model of describing performance of motor activities, it has also a use in human computer interaction studies in both design and evaluation of computer input devices [23] and as well as user interface design [24]. In analogy with the original experiment, in human computer interaction the task is clicking on an on-screen target with the cursor controlled by the mouse or other pointing device as illustrated in Figure 3.5.

Fitts' Law is extended to 2-dimensional tasks in [25]. In our experiments, the original formula is applicable since we use circular targets. For the experiments,

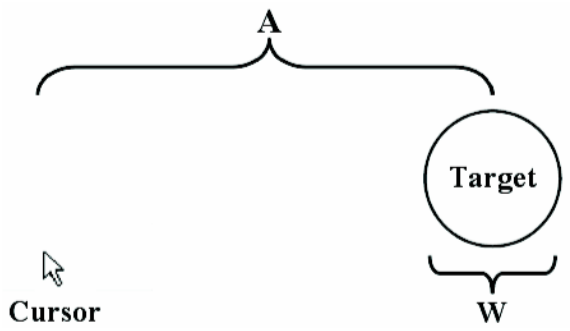


Figure 3.5: Experiment adapted for human computer interaction in Fitts' Law.

an OpenGL application shown in Figure 3.6 is developed.

In the experiments, we have performed a similar testing process to the one presented in [13]. The subjects are asked to point and click at circular targets as quickly as possible. If the subject misses the target (i.e. clicks out the target), an extra trial is added and at the end only the correct trials are collected for time performance analysis.

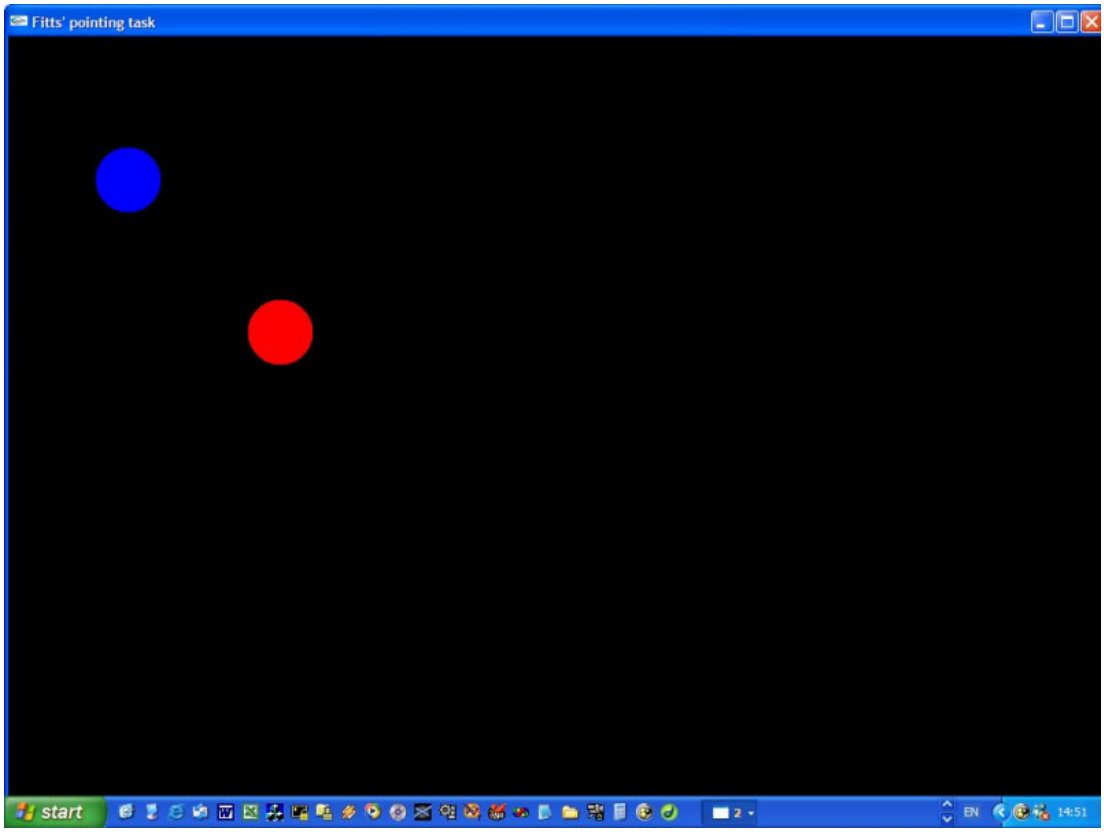


Figure 3.6: OpenGL Application developed for experiments.

In experiments, a 17 inch CRT monitor and a resolution of 1024 by 768 pixels are used and two different subjects, each completes 154 trials of pointing and clicking task. During the experiments, the following factors are changed randomly:

- Targets have two different sizes; 20 pixels (0.23 in) and 60 pixels (0.7 in) in diameter.
- Targets are displayed at three different distances; 200 pixels (2.34 in), 500 pixels (5.85 in) and 800 pixels (9.38 in).
- Pointing directions are horizontal, vertical and diagonal. (For 800 pixels target distance, diagonal pointing direction can not occur).

According to the experiments, the average completion time is 1.499 secs. As expected, the pointing time is significantly related with the target size as shown in Figure 3.7. Also Figure 3.8 shows the relation between pointing time and target distance. Additionally, IP of the system is calculated as 5.20 bits/sec. In the work by MacKenzie *et al.* [25], IP of a standard mouse is calculated as nearly 6 bits/sec. and for an eye tracker named MAGIC [13], the IP of the system is 4.76 bits/sec. From these results, we can infer that the proposed vision-based mouse system is not as powerful as a standard mouse, but it is a good alternative among the systems developed for disabled people.

3.3.2 Success of Calibration

The success of calibration phase is directly related with the computed homography matrix. The accuracy of the calibration is increased by the help of some additional error correction to the result obtained by using homography matrix. In testing the overall success of calibration, a similar task to the one described in Section 3.3.1 is selected. Circles having radii 16 pixels are displayed on screen with a resolution 1024 by 768 pixels and the subjects are asked to point center of these circles.

According to the experiments, the average error is calculated as 7 pixels. This error range is fairly acceptable for a resolution of 1024 by 768 pixels. Furthermore, the effect can be decreased by designing the graphical user interface(sizes of buttons, icons, menus, etc.) according to this error measure.

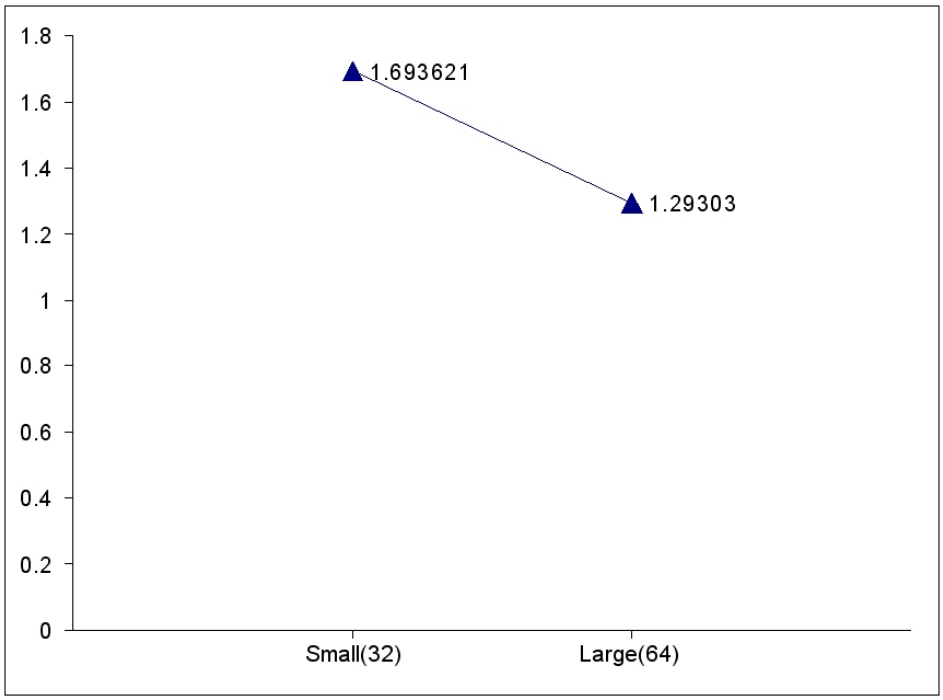


Figure 3.7: Target size vs. average completion time graph.

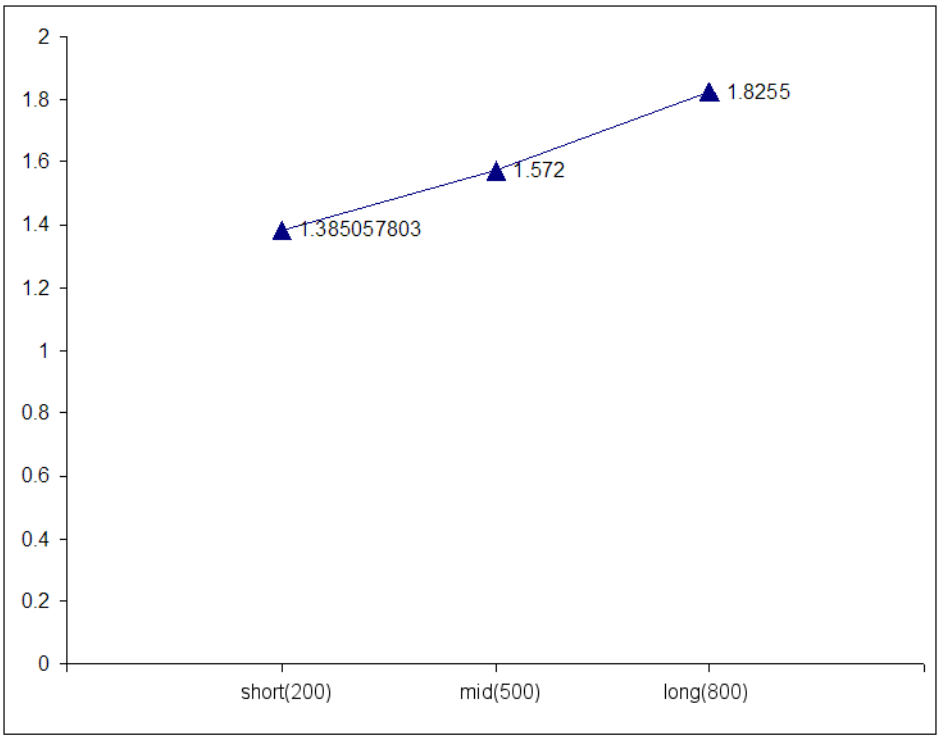


Figure 3.8: Target distance vs. average completion time graph.

CHAPTER 4

VISION-BASED CONTINUOUS GRAFFITI^{TM}-LIKE TEXT ENTRY SYSTEM

Unistroke isolated character recognition systems are successfully used in personal digital assistants in which people feel easier to write rather than type on a small size keyboard [26, 27]. In this approach it is assumed that each character is drawn by a single stroke as an isolated character. One of the alphabets that has this property is the Graffiti^{TM}. In [10], a vision-based system for recognizing isolated Graffiti^{TM} characters is proposed. In this system, the user draws characters by a pointer or a stylus on a flat surface or the forearm of a person. In our study, we extend the work in [10], isolated Graffiti^{TM} recognition problem, to continuous Graffiti^{TM} recognition. To increase to recognition accuracy of the system, we have modified the original Graffiti^{TM} alphabet. Modified alphabet can be seen in Figure 4.1.

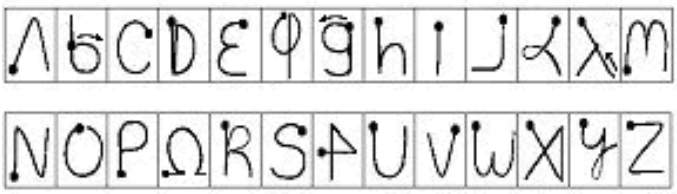
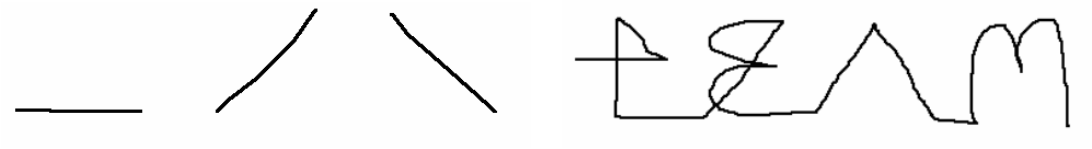


Figure 4.1: Modified alphabet. Heavy dot indicates starting point.

In this handwriting method, the transitions from a character to another are

also restricted to the three possible strokes shown in Figure 4.2(a). Transition from one character to another can be done with a horizontal line segment, a monotonically decreasing convex curve or a monotonically increasing convex curve. An example word "team" is written in continuous Graffiti^{TM} in Figure 4.2(b).



(a) (b)

Figure 4.2: (a) Character to character transition strokes, (b) Word "team" written in continuous Graffiti^{TM}-like alphabet.

In the current system, the user writes in continuous Graffiti^{TM} using a laser pointer on the forearm captured by a camera mounted on the forehead or a shirtpocket. The video is segmented to image sequences corresponding to each written word. The image sequence starts with a laser pointer turn-on action and terminates when the user turns off the laser pointer. In each image in this sequence, the beam of the laser pointer is located by the tracker module and after obtaining these sample points, the recognition module outputs the recognized word. As the overall system architecture is shown in Figure 4.3, the system is composed of tracking and recognition phases.

The advantages of our vision-based text entry systems compared to other vision-based systems [28, 11, 9] are the following:

- The background is controlled by the forearm of the user. Furthermore, if the user wears a unicolor fabric then the tip of the finger or the beam of the pointer can be detected in each image of the video by a simple image processing operation such as thresholding.

- It is very easy to learn a Graffiti^{TM}-like alphabet. Only a few characters are different from the regular Latin alphabet. Although it may be easy to learn other text entry systems such as [19, 28, 11], some people are reluctant to

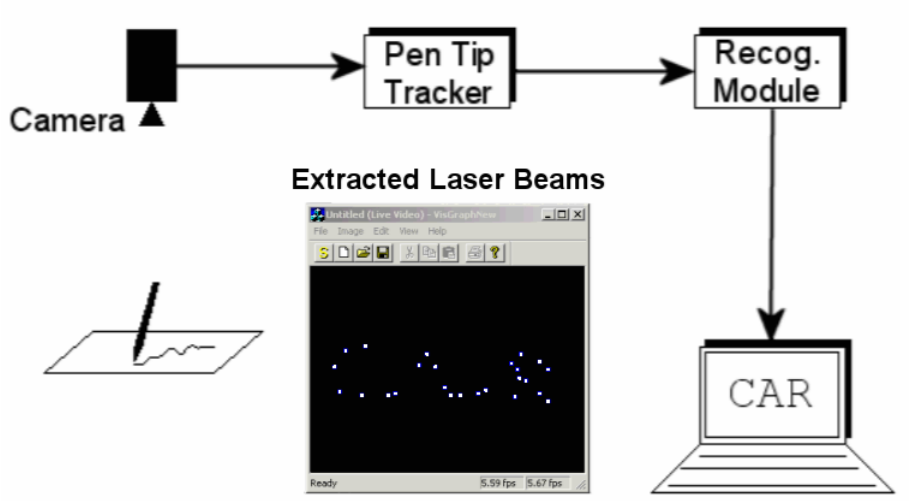


Figure 4.3: Overall system architecture of Vision-Based Continuous Graffiti^{TM}-like Text Entry System.

spend a few hours to learn unconventional text entry systems. Furthermore, in addition to the regular characters other single stroke characters can be defined by the user to be used as bookmarks, pointers to databases etc.

- Computationally efficient, low power consuming algorithms exist for the recognition of unistroke characters and they can be implemented in real time with very high recognition accuracy. After a few minutes of studying the Graffiti^{TM}-like alphabet, recognition accuracy is very high compared to regular handwriting recognition method described in [9].

- Computer vision-based text entry systems are almost weightless.

4.1 Tracking

The tracking phase is same as the one described in Section 2.3.1. The beam of the laser pointer is determined by detecting the moving pixels in the current image of the video and from the color information and then by taking the image difference of two consecutive image frames, the moving

pixels can be determined. Among those moving pixels, the position of the beam of the laser pointer is determined by examing color information.

4.2 Recognition

As shown in Figure 4.4, the position of the pen tip and pen up/down information extracted in tracking phase is applied as an input to the recognition system. First the chain code is extracted from the relative motion of the beam of the laser pointer between consequitive camera images. Then the extracted chain code of the word is analyzed and all possible words conforming the extracted chain code are determined. At the end by performing a lexical analysis, the recognized word(s) are displayed on the screen.

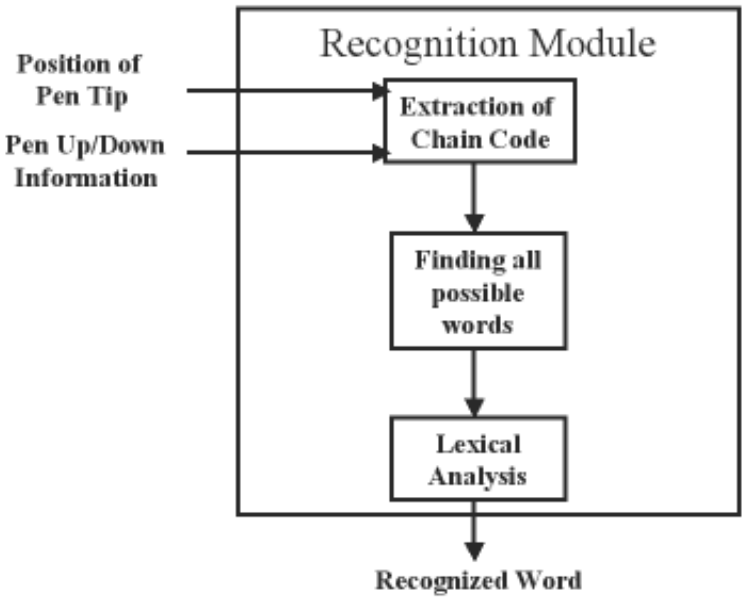
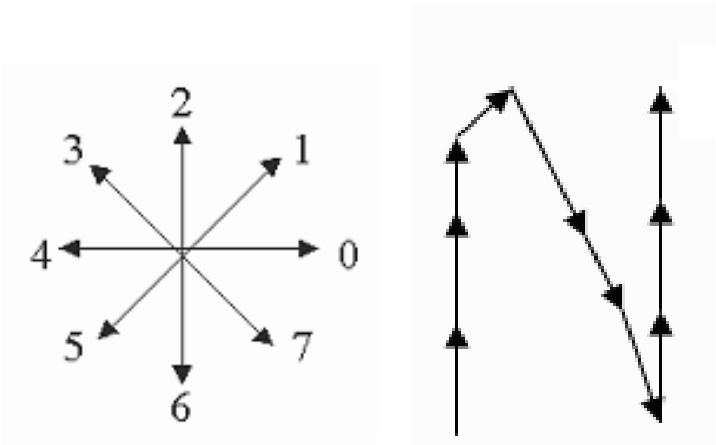


Figure 4.4: The inner structure of Recognition Module.

4.2.1 Extraction of chain code

In our system unistroke characters are described using a chain code which is a sequence of numbers between 0 and 7 obtained from the quantized angle of the beam of the laser pointer in an equally time sampled manner

as shown in Figure 4.5(a). Chain coded representation of characters are generated according to the angle between two consecutive positions of the beam of the laser pointer. A sample chain coded representation of the word "N" is shown in Figure 4.5(b).



(a) (b)

Figure 4.5: (a) Chain code values for the angles, (b) A sample chain coded representation of the character "N" is [2,2,2,1,7,7,6,2,2,2].

4.2.2 Finding all possible words

Each character in the alphabet and transition strokes are all represented by a distinct finite state machine (FSM) (see Table 4.1). If we have an extracted chain code of a character, we can recognize that character by examining it according to the FSMs representing each character in the alphabet. As an example, in Figure 4.5(b), the character "N" is characterized by the chain code [2,2,2,1,7,7,6,2,2,2] where the finite state machine for the character "N" is shown in Figure 4.6. The first three inputs, 2,2 and 2, do not produce any error when applied to the first state of the FSM representing the character "N". The forth number of the chain code, 1, leads to an error and an increase in the error counter by 1. The next input 7 makes the FSM to go to the next state and the subsequent 7, and 6 let the machine to remain there. Whenever the input becomes 2, the FSM moves to the third

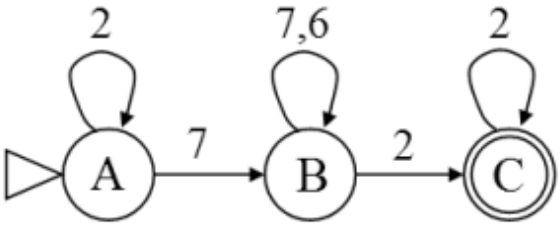


Figure 4.6: Finite State Machine for the character "N".

state. The machine stays in this state until the end of the chaincode and the FSM terminates with an error value of 1. When we extend this analysis over all FSMs, we come up with the character recognition algorithm shown in Algorithm 5.

It is preferred that a word can be segmented into characters by examining the transition strokes. But in general, this may not be possible since these detected transition strokes can also be a sub-stroke of a character. Therefore, our recognition module works in a recursive manner and outputs all possible words of the extracted chain code. As described before, each FSM representing a character returns an error value, the ones having minimum errors are selected and for each one, the next chaincode inputs will be passed to all the FSM's for the next character. This process continues until the

Algorithm 5 Character Recognition Algorithm based on analysis using FSMs.
Given the extracted chain code of a character

(i) The chain code is applied as input to all FSMs representing each character.

(ii) State changes are determined and additionally an error counter is increased by one if a change is not possible according to the current FSM.

(iii) If a chain code does not terminate in the final state, the corresponding character is eliminated.

(iv) Errors in each state are added up to find the final error for each character.

(v) Character with the minimum error is the recognized one.

Table 4.1: The table of FSMs for each character in the alphabet.

Character	*Corresponding FSM*
A	12 76
B	107 654 32
C	345 67
D	67 12 076 54
E	345 670 345 670
F	456 07 12 345 6
G	345 67 012 654 3210
H	6 6 210 76
I	6 6
J	6 54
K	65 4321 07
L	7 23 5
M	21 67 210 76
N	12 7 12
O	45 67 012 34
P	2 01 76 54
Q	234 10 765
R	6 12 076 54 076
S	345 607 543
T	0 23 56
U	6 071 2
V	65 32
W	67 012 670 12
X	7 12 56
Y	67 012 654 321
Z	0 5 0

end of chaincode is reached.

It is observed that the FSM based recognition algorithm is robust as long as the user does not move his arm or the camera during the writing process of a letter. Characters can be also modeled by Hidden Markov Models which are stochastic FSM's instead of the deterministic FSM's to further increase the robustness of the system at the expense of higher computational cost. In addition, in order to prevent noisy state changes, look-ahead tokens can be used which acts as a smoothing filter on the chain code.

4.2.3 Lexical Analysis

At the end of the step described in Section 4.2.2, a list of all possible words is obtained. In lexical analysis step, the meaningless words are eliminated by looking up a 18000 word dictionary which is composed of most common English words. At the end, only the words found in the dictionary are displayed as recognized ones in sorted order according to their total error count. This can be seen in Figure 4.7.

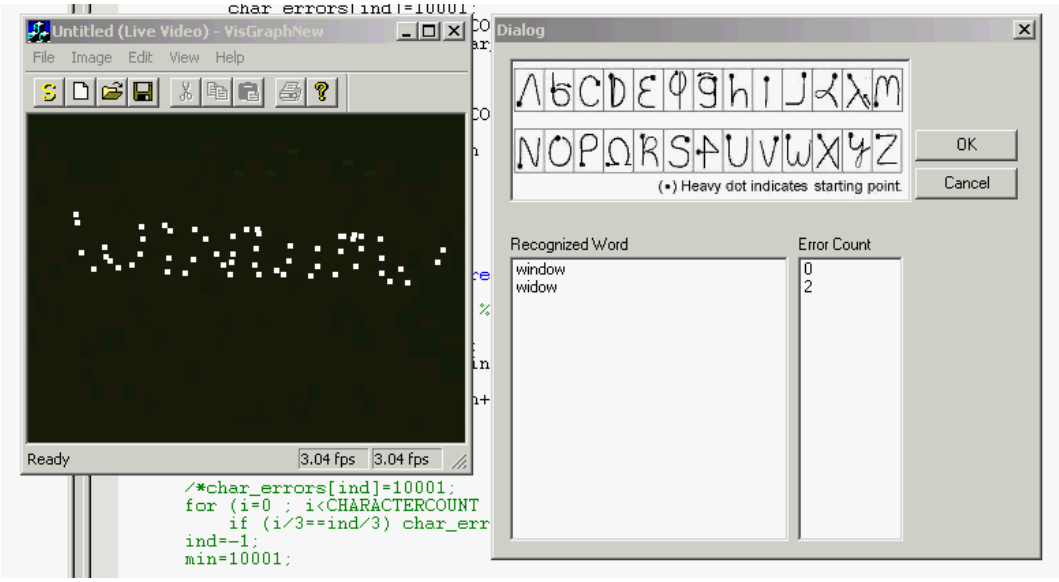


Figure 4.7: The result of lexical analysis for the written word "window".

4.3 Evaluation

In our experiments, we have a PC with Intel Pentium IV-1.7Ghz processor with 512 GB memory, a webcam producing 320x240 pixels color images at 13.3 fps and an ordinary laser pointer. The user draws continuous Graffiti^{TM} characters using the laser pointer on the dark background material. In Graffiti^{TM}-like recognition systems, very high recognition rates are possible [27].

To examine the performance of our system, the system is tested with a word dataset consisting of 30 words in various lengths and these words are written at

least 15 times by different people. In our system, in spite of the existence of perspective distortion, it is possible to attain a recognition rate of 93% at the word level and the writing speed is about 8 wpm.

In addition, when we examine recognition rate versus word length graph shown in Figure 4.8, we can infer although word length has an importance, recogition rate is mainly related with the difficulty of writing the words in Continous Graffiti^{TM}-like alphabet. Also, mean completion time of written word versus word length graph which is given in Figure 4.9 shows that writing time increases linearly with the increase in word length.

It is also observed that the recognition process is writer independent with little training and we believe that we can achieve higher writing speed rates with the advances in digital camera and wearable computer technology.

The perspective distortion plays some role in the recognition accuracy of the system. In our experiments, we have observed that the degradation in recognition is at most 10% around 30 degree difference between the plane on the which writing is performed and the camera.

Several tests are also carried out under different lighting conditions. In day/incandescent/fluorescent light, the average intensity of the background is about 50/180/100 whereas the intensity value of the beam of the laser pointer is about 240/250/240. In all cases, the beam of the laser pointer can be easily identified from the dark background.

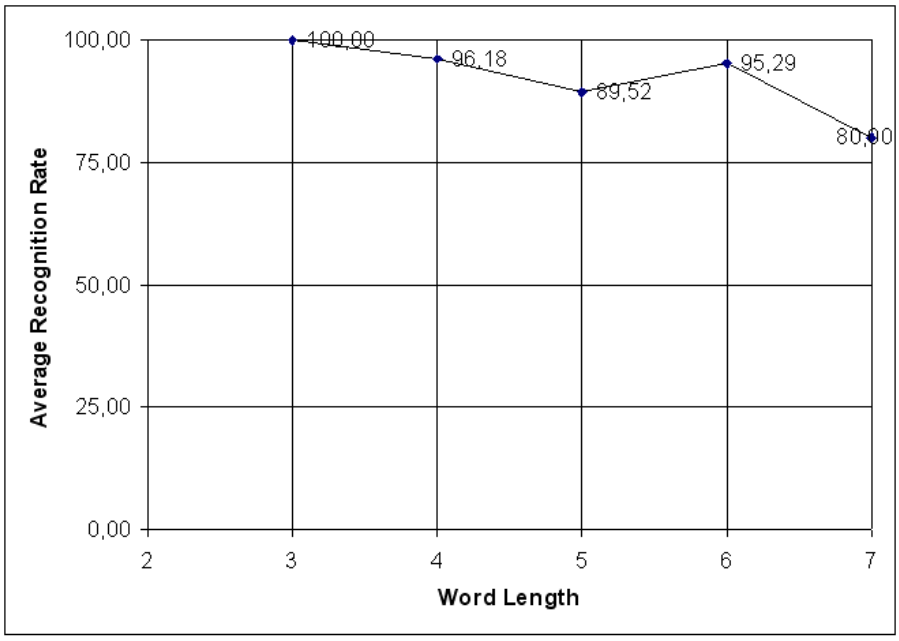


Figure 4.8: Recognition rate versus word length.

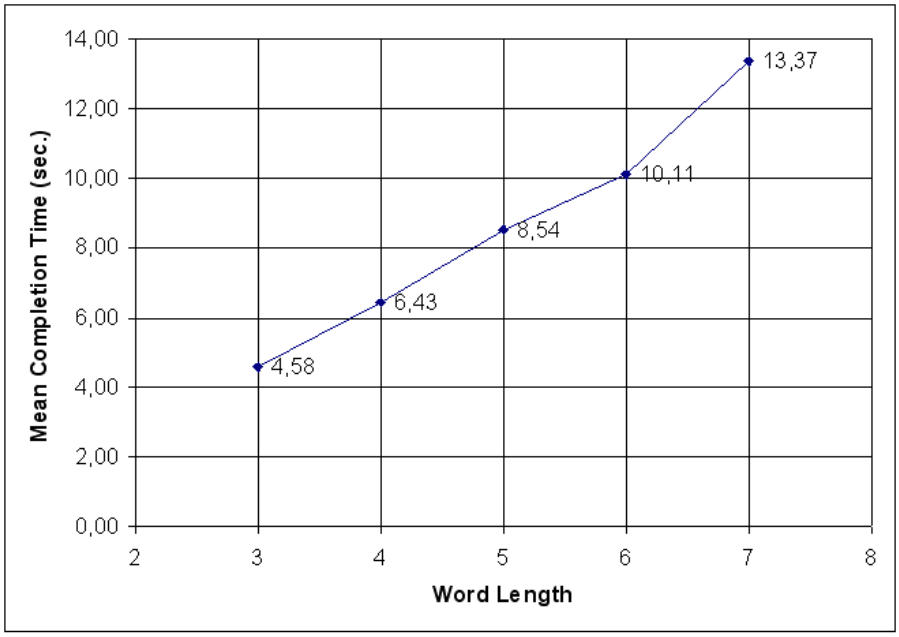


Figure 4.9: Mean completion time versus word length.

CHAPTER 5

SUMMARY AND CONCLUSIONS

5.1 Summary

Computer vision provides a powerful tool for human-computer interaction (HCI). In this study, three different vision-based HCI systems are described: Vision-Based Keyboard System, Vision-Based Mouse System and Vision-Based Continuous Graffiti^{TM}-like Text Entry System. All systems require only a webcam and an ordinary laser pointer as the pointing device. Each system constantly monitors the beam of the laser pointer in the camera images and performs the corresponding actions.

In Vision-Based Keyboard System, the keyboard is a passive device therefore it can be made up of any material having a keyboard layout image. The web camera is placed to see the entire keyboard image and captures the movement of the laser beam. The user enters a character to the computer by covering the corresponding character region in the keyboard layout image with the laser pointer. Futhermore, this keyboard system can be easily adapted for disabled people who have little or no control of their hands to use an ordinary keyboard. The disabled user wears an eyeglass to which a laser pointer is attached and controls the beam of the laser pointer by his/her head movements.

For the same class of disabled people, Vision-Based Mouse System is developed

as a companion system. This time, the webcam is placed to see the whole screen and the system provides a solution to disabled users for controlling the mouse cursor by head movements.

In Vision-Based Continuous Graffiti^{TM}-like Text Entry System, a Graffiti^{TM}-like alphabet is developed where the users can write characters in a continuous manner on a flat surface using the laser pointer. The video is segmented to image sequences corresponding to each written word. Every image sequence starts with a laser pointer turn-on action and ends when the user turns off the laser pointer. In each image in this sequence, the beam of the laser pointer is tracked and the written word is recognized from the extracted trace of the laser beam. Recognition is based on finite state machine representations of characters in the alphabet.

5.2 Conclusions

According to the experiments, average recognition rate of Vision-Based Keyboard System is determined as 94% with a text entry speed of 8 words per minute(wpm). In addition, the system is writer-independent and there is no direct relation of word length with the recognition rate. Recognition is mainly related with the key strokes between each character of the written word. Furthermore, the writing time increases linearly with the increase in word length as expected.

Vision-Based Mouse is tested with standard Fitts' pointing task [1] where the subjects are asked to point and click at circular targets and as a result, a measure called Index of Performance (IP) of the system is determined. According to the experiments, the average completion time of the pointing tasks is around 1.5 secs. and IP of the system is calculated as 4.97 bits/sec where as a standard mouse has nearly IP=6 bits/sec. [25] and an eye tracker system named MAGIC [13] has an IP=4.76 bits/sec. From these results, we can conclude that our system is not as powerful as a standard mouse, but it is a good alternative among the systems developed for disabled people.

Recognition rate of Vision-Based Graffiti^{TM}-like Text Entry System is measured as 93% at the word level and the writing speed as around 8 wpm. It is also

observed that the system is writer-independent and requires little training for learning the alphabet. Although word length has an importance on recognition, recogition is mainly related with the difficulty of writing the words in Continuous Graffiti^{TM}-like alphabet. Also, the writing time increases linearly with the increase in word length.

5.3 Future Work

The current layout of the keyboard is based on alphabetical ordering. With this layout structure, new users can start using the Vision-Based Keyboard System with little training. Although it is easy to learn, the current writing speed of 8 wpm is not very high. As a future work, the writing speed of the system can be increased in two different ways by using a word dataset:

- The keyboard layout can be optimized by minimizing the median distance of pen travels between letters according to word dataset.
- Like T9^{TM} technology [29] used in cellular phones, the intended word can be completed before it is fully written.

Since we use laser pointer as the pointing device, tracking the beam in real-time is not a complicated process. As future work, possibility of using some other pointing devices (e.g. finger, an ordinary pen, etc.) can be investigated. But this time to track the tips of these pointers, some complex feature trackers (e.g. Kanade-Lucas-Tomasi (KLT) point-based feature tracker [30]) in a combination with a Kalman filter [31] can be used.

REFERENCES

[1] J. Mankoff and G. D. Abowd. Cirrin: A word-level unistroke keyboard for pen input. In *Proceedings of Symposium on User Interface Software and Technology (UIST'98)*, pages 213–214, 1998.

[2] P.M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47:381–391, 1954.

[3] J. Nielsen. *Usability Engineering*. Academic Press, Inc., 1993.

[4] P. Wellner. The digitaldesk calculator: Tangible manipulation on a desk top display. In *Proceedings of ACM SIGGRAPH Symposium on User Interface Software and Technology*, pages 107–115, 1991.

[5] P. Wellner. Interacting with paper on the DigitalDesk. *Communications of the ACM*, 36(7):86–97, 1993.

[6] J. L. Crowley, J. Coutaz, and F. Brard. Perceptual user interfaces: things that see. *Communications of the ACM*, 43(3):54–64., 2000.

[7] Z. Zhang, Y. Wu, Y. Shan, and S. Shafer. Visual panel: Virtual mouse, keyboard and 3d controller with an ordinary piece of paper. In *Proceedings of ACM Perceptive User Inferface Workshop(PUI01)*, Florida, November 2001.

[8] F. Quek, T. Mysliwiec, and M. Zhao. Fingermouse: A freehand pointing interface. In *Proceedings of the International Workshop on Automatic Face- and Gesture-Recognition*, pages 372–377, 1995.

[9] M.E. Munich and P. Perona. Visual input for pen-based computers. In *Proceedings of 13th Internation Conference on Pattern Recognition*, pages 33–37, 1996.

[10] G. A. Fink, M. Wienecke, and G. Sagerer. Video-based on-line handwriting recognition. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 226–230. IEEE, 2001.

[11] O. Faruk Ozer, O. Ozun, V. Atalay, and A. Enis Cetin. Visgraph: Vision based single stroke character recognition for wearable computing. *IEEE Intelligent Systems and Applications*, 16(3):33–37, May-June 2001.

[12] T. Starner, J. Weaver, and A. Pentland. A wearable computing based american sign language recognizer. In *Proceedings of the 1st International Symposium on Wearable Computers*, October 1997.

[13] http://www.research.ibm.com/ecvg/tfs/tfs.html as accessed on, July 2003.

[14] C. Morimoto S. Zhai and S. Ihde. Manual and gaze input cascaded (magic) pointing. In *Proceedings of ACM CHI'99 Conference on Human Factors in Computing Systems*, pages 246–253, 1999.

[15] http://www.madentec.com as accessed on, July 2003.

[16] http://www.orin.com as accessed on, July 2003.

[17] http://www.arringtonresearch.com as accessed on, July 2003.

[18] C. Tomasi, A. Rafii, and I. Torunoglu. Full-size projection keyboard for handheld devices. *Communications of the ACM*, 46(7):70–75., 2003.

[19] http://www.handykey.com as accessed on, July 2003.

[20] Z. Zhang and Y. Shan. Visual screen: Transforming an ordinary screen into a touch screen. In *Proceedings of IAPR Workshop on Machine Vision Applications (MVA 2000)*, pages 215–218, 2000.

[21] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes in C: The Art of Scientific Computing (2nd ed.)*. Cambridge University Press, 1992.

[22] C. E. Shannon and W. Weaver. *The Mathematical Theory of Communication*. The University of Illinois Press, 1949.

[23] I.S. MacKenzie. Fitts law as a research and design tool in human computer interaction. *Human Computer Interaction*, 7.

[24] J. Accot and S. Zhai. Beyond Fitts' law: models for trajectory-based HCI tasks. In *Proceedings of ACM CHI'97 Conference on Human Factors in Computing Systems*, pages 295–302, 1997.

[25] I. S. MacKenzie and W. Buxton. Extending fitts' law to two-dimensional tasks. In *Proceedings of ACM CHI'92 Conference on Human Factors' in Computing Systems*.

[26] D. Goldberg and C. Richardson. Touch-typing with a stylus. In *Proceedings of the INTERCHI'93 Conference on Human Factors in Computing Systems*, pages 80–87, 1993.

[27] I.S. MacKenzie and S. Zhang. The immediate usability of graffiti. In *Proceedings of Graphics Interface'97*, pages 129–137, Toronto, 1997.

[28] J. A. Robinson A. Vardy and L-T Cheng. The wristcam as input device. In *Proceedings of the Third International Symposium on Wearable Computers*, pages 199–202, October 1999.

[29] http://www.t9.com as accessed on, July 2003.

[30] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical Report CMU-CS-91132, Carnegie Mellon University School of Computer Science, Pittsburgh, 1991.

[31] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

[32] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2000.

APPENDIX A

BILINEAR INTERPOLATION

This section is inspired from Numerical Recipes in C: The Art of Scientific Computing (2nd ed.) [21].

In two dimensions, we imagine that we are given a matrix of functional values `ya[1..m][1..n]`.We are also given an array `x1a[1..m]`, and an array `x2a[1..n]`.

The relation of these input quantities to an underlying function $y(x_1, x_2)$ is

$$\texttt{ya[j][k]} = y(\texttt{x1a[j], x2a[k]}) \tag{A.1}$$

We want to estimate, by interpolation, the function y at some untabulated point (x_1, x_2).

An important concept is that of the grid square in which the point (x_1, x_2) falls, i.e. the four tabulated points that surround the desired interior point. For convenience, we will number these points from 1 to 4, counterclockwise starting from the lower left (see Figure A). More precisely, if

$$\begin{aligned} \texttt{x1a[j]} \leq x_1 \leq \texttt{x1a[j+1]} \\ \texttt{x2a[k]} \leq x_2 \leq \texttt{x2a[k+1]} \end{aligned} \tag{A.2}$$

defines `j` and `k`, then

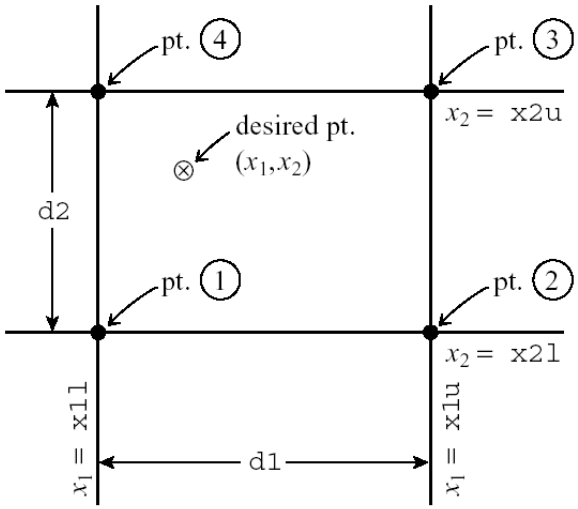


Figure A.1: Labeling of points used in bilinear interpolation.

$$\begin{aligned} y_1 &\equiv \texttt{ya[j][k]} \\ y_2 &\equiv \texttt{ya[j+1][k]} \\ y_3 &\equiv \texttt{ya[j+1][k+1]} \\ y_4 &\equiv \texttt{ya[j][k+1]} \end{aligned} \tag{A.3}$$

Then the formulas for bilinear interpolation on the grid square are:

$$\begin{aligned} t &\equiv (x_1 - \texttt{x1a[j]})/(\texttt{x1a[j+1]- x1a[j]}) \\ u &\equiv (x_2 - \texttt{x2a[k]})/(\texttt{x2a[k+1]- x2a[k]}) \end{aligned} \tag{A.4}$$

(so that t and u each lie between 0 and 1), and

$$y(x_1, x_2) = (1-t)(1-u)y_1 + t(1-u)y_2 + tuy_3 + (1-t)uy_4 \tag{A.5}$$

APPENDIX B

THE DIRECT LINEAR TRANSFORMATION (DLT) ALGORITHM

This section is inspired from Multiple View Geometry in Computer Vision [32].

Direct Linear Transformation (DLT) algorithm is for determining the transformation matrix H between 2D to 2D point correspondences, $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$. This transformation can be expressed as $\mathbf{x}'_i = H\mathbf{x}_i$. In this equation, $\mathbf{x}'_i$ and $H\mathbf{x}_i$ are homogeneous vectors, i.e. they have the same direction but may differ in magnitude by a non-zero scale factor. Therefore, the equation can be written in terms of the vector cross product as $\mathbf{x}'_i \times H\mathbf{x}_i = \mathbf{0}$.

If the j-th row of the matrix H is denoted by ${\mathbf{h}^j}^\top$, then we may write

$$H\mathbf{x}_i = \begin{pmatrix} {\mathbf{h}^1}^\top \mathbf{x}_i \\ {\mathbf{h}^2}^\top \mathbf{x}_i \\ {\mathbf{h}^3}^\top \mathbf{x}_i \end{pmatrix}.$$

Writing $\mathbf{x}'_i = (x'_i, y'_i, w'_i)\top$, the cross product can be written as

$$\mathbf{x}'_i \times H\mathbf{x}_i = \begin{pmatrix} y'_i {\mathbf{h}^3}^\top \mathbf{x}_i - w'_i {\mathbf{h}^2}^\top \mathbf{x}_i \\ w'_i {\mathbf{h}^1}^\top \mathbf{x}_i - x'_i {\mathbf{h}^3}^\top \mathbf{x}_i \\ x'_i {\mathbf{h}^2}^\top \mathbf{x}_i - y'_i {\mathbf{h}^1}^\top \mathbf{x}_i \end{pmatrix}.$$

Since ${\mathbf{h}^j}^\top \mathbf{x}_i = {\mathbf{x}_i}^\top \mathbf{h}^j$ for $j = 1, \ldots, 3$, the previous cross product gives a set

of three equations in the entries of H, which may be written as

$$\begin{bmatrix} \mathbf{0}^\top & -w_i'\mathbf{x}_i{}^\top & y_i'\mathbf{x}_i{}^\top \\ w_i'\mathbf{x}_i{}^\top & \mathbf{0}^\top & -x_i'\mathbf{x}_i{}^\top \\ -y_i'\mathbf{x}_i{}^\top & x_i'\mathbf{x}_i{}^\top & \mathbf{0}^\top \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}. ThomasWolfe'u' \tag{B.1}$$

If the 3×9 matrix in the previous equation is denoted by $\mathbf{A}_i$, then we may write $\mathbf{A}_i\mathbf{h} = \mathbf{0}$, where $\mathbf{h}$ is a 9-vector composed of the elements of the matrix H,

$$\mathbf{h} = \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix}, \quad H = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \tag{B.2}$$

with h_i is the i-th element of $\mathbf{h}$.

The equation $\mathbf{A}_i\mathbf{h} = \mathbf{0}$ is linear in the unknown $\mathbf{h}$. The elements of matrix $\mathbf{A}_i$ can be determined from the known correspondence points. Although there are three equation in Equation B.1, only two of them are linearly independent. The third row can be obtained, up to scale, from the sum of x_i' times the first row and y_i' times of the second row. Therefore, we can omit the third equation for solving H and obtain the following system of equations:

$$\begin{bmatrix} \mathbf{0}^\top & -w_i'\mathbf{x}_i{}^\top & y_i'\mathbf{x}_i{}^\top \\ w_i'\mathbf{x}_i{}^\top & \mathbf{0}^\top & -x_i'\mathbf{x}_i{}^\top \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}. \tag{B.3}$$

This will be written as $\mathbf{A}_i\mathbf{h} = \mathbf{0}$ where this time $\mathbf{A}_i$ is a 2×9 matrix. The equations hold for any homogeneous coordinate representation $(x_i', y_i', w_i')\top$ of the point $\mathbf{x}_i'$. If we choose $w_i' = 1$, (x_i', y_i') is the coordinates measured in the image.

B.1 Solving for H from a set of four point correspondences

One point correspondence gives two independent equations for the elements of H. By using a set of four point correspondences, we obtain a set of equations $\mathbf{A}\mathbf{h} = 0$, where $\mathbf{A}$ is the equation coefficient matrix built from the rows of matrix $\mathbf{A}_i$ of each point correspondence, and $\mathbf{h}$ is the vector of unknown elements of H.

Since the obvious solution $\mathbf{h} = 0$ is of no interest to us, we seek for a non-zero solution. In building $\mathbf{A}$ either Equations B.1, or B.3 can be used, but A has rank 8 for all cases and has a 1-dimensional null-space which provides a solution for $\mathbf{h}$ with a non-zero scale factor. However, H is in general only determined up to scale, so the solution $\mathbf{h}$ gives the required H. A scale may be arbitrarily chosen for $\mathbf{h}$ by a requirement on its norm such as $\|\mathbf{h}\| = 1$.

B.2 Over-determined solution

If we have more than four point correspondences $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$, then the set of equations $\mathbf{Ah} = \mathbf{0}$ derived from Equation B.3 is over-determined. If the point correspondences are exact, then the matrix $\mathbf{A}$ will still have rank 8, a one dimensional null-space, and there is an exact solution for $\mathbf{h}$. If there is noise in the point correspondences, there will be no exact solution for $\mathbf{h}$ apart from the zero solution. Therefore, we determine an approximate solution for $\mathbf{h}$ that minimizes the norm $\|\mathbf{Ah}\|$. This is identical to the problem of finding the minimum of the quotient $\|\mathbf{Ah}\|/\|\mathbf{h}\|$ and its solution is the unit eigenvector of $\mathbf{A}^\top\mathbf{A}$ with least eigenvalue. Equivalently, the solution for $\mathbf{h}$ is the unit singular vector corresponding to the smallest singular value of $\mathbf{A}$. The overall algorithm, known as the basic DLT algorithm, is shown in Algorithm 6.

Algorithm 6 The basic Direct Linear Transformation (DLT) Algorithm

Given n ≥ 4 2D to 2D correspondences $\{x_i \leftrightarrow x'_i\}$, determine the 2D homography matrix H such that x'_i=Hx_i.

(i) For each correspondence $x_i \leftrightarrow x'_i$ compute the matrix $\mathbf{A}_i$ from Equation B.3.

(ii) Assemble the n 2×9 matrices $\mathbf{A}_i$ into a single $2n \times 9$ matrix $\mathbf{A}$.

(iii) Obtain the SVD of $\mathbf{A}$. The unit singular vector corresponding to the smallest singular value is the solution $\mathbf{h}$.

(iv) The matrix H is determined from $\mathbf{h}$ as in Equation B.2.