LOW BIT RATE VISUAL COMMUNICATION USING BINARY SKETCHES FOR DEAF SIGN LANGUAGE COMMUNICATION

MOORTHY, MANORANJAN

LOW BIT RATE VISUAL COMMUNICATION USING BINARY SKETCHES FOR DEAF SIGN LANGUAGE COMMUNICATION

BY

D. M. MANORANJAN, B. ENG.

AUGUST 1998

A THESIS SUBMITTED TO THE SCHOOL OF GRADUATE

STUDIES IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

MASTER OF ENGINEERING

FACULTY OF ENGINEERING AND APPLIED SCIENCE

MEMORIAL UNIVERSITY OF NEWFOUNDLAND

ST. JOHN'S, NEWFOUNDLAND

ST. JOHN'S NEWFOUNDLAND CANADA

To my parents

Abstract

Many commercial video conferencing applications are available to transmit color and gray level images along with voice and text data over telephone lines and Internet protocol networks. However, these applications cannot be used for deaf sign language communication because they offer very low temporal resolution of pictures, at less than a frame per second resulting in jerky movement of objects. Effective sign language communication requires a temporal resolution of the order of 8 to 12 frames per second, while the size of the images may be small.

The research for this thesis was aimed at providing a practical solution to enable deaf sign language communication using moving cartoons. Since sign language relies on hand shape, orientation, position and movement together with facial expressions, it can be adequately conveyed in moving cartoons or binary sketches. The cartoon or binary sketch is a very economical method of portraying a person or a scene, where emphasis is given to the location of object boundaries at the expense of shading and texture information. An efficient feature extraction algorithm is used to extract cartoon points. This algorithm is based on the postulate that perceptually significant features of the human face and hands, at which cartoon lines should be drawn in the image, occur wherever surfaces in object space are approximately tangential to the line of sight of the camera or viewer. Based on the above postulate, the detector exhibits primary sensitivity to valleys and secondary

sensitivity to edges in the image. Adaptive techniques based on the global statistics of the image are proposed to improve the functionality of the feature extractor. To improve the subjective quality of the binary images, irreversible preprocessing techniques, such as threshold hysteresis, isolated point removal and predictive filter are used and results are provided to compare their relative performances.

A simple and efficient recursive temporal filtering scheme is used to prefilter noisy image sequences from low cost cameras. This scheme uses the spatial homogeneity of the low frequency additive and multiplicative noise, which produces a throbbing or pulsing effect on the image sequences. Histograms of successive frames are used to segment moving and non-moving regions and to formulate a transfer function. The transfer function is used in the filtering operation. Different temporal and spatio-temporal filtering methods are implemented. Subjective and quantitative test results are presented to demonstrate their relative performances.

An efficient three-dimensional binary image compression scheme is used for binary sketches. This method uses conditional replenishment in the temporal dimension and quad tree or hierarchical coding in the two dimensional space. A complete system is developed for deaf sign language communication over low bandwidth telephone lines and Internet protocol networks. Subjective tests are performed on the system and the results confirm that the system can be used for sign language communication over telephone lines.

Acknowledgements

I would like to thank my supervisor, Dr. John A. Robinson, for the invaluable guidance he has provided, and the work he has put into helping me with my master's program. I would also like to thank him for the time he has spent assisting with revisons to the thesis despite his busy schedule.

I am grateful to my colleague Mr. Li-Te Cheng, whose assistance and advice was tremendously helpful in all stages of my master's program. I also would like to thank all my colleagues at the Multimedia Communications Laboratory for their useful comments and suggestions regarding my research.

I would like to thank Mr. Myles Murphy and Ms. Jeannie Leonard, from the Newfoundland Coordinating Council on Deafness, for their active participation in testing the system developed.

I would especially like to thank the Dr. John A. Robinson, the School of Graduate Studies and the Faculty of Engineering and Applied Science at Memorial University of Newfoundland for providing financial assistance for my master's program.

I thank my wife, Vidya, for her love, patience, and support. I would like to thank Vidya's parents for their support and help. I especially thank my family in India for their love, encouragement and moral support.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Interest in video telephony for the deaf and hard of hearing has been increasing recently due to technological advancements in the field of computer communications, and the desire to break social barriers. The deaf have developed a complex and ingenious language of signs and fingerspelling together with the ability to infer spoken words from lip movements. The visual language of deaf is made up mainly of signs, which are gestures made primarily with the hands and arms, but also with the face and other parts of the body. Signs can convey a complicated idea for which hearing people use sentences, at a rate that is about the same as for speech. Finger-spelling is used for technical words, place names, people's names, etc., where each letter has to be communicated: the rate at which sentences are delivered slows down when many words have to be spelled in this way. Signing, fingerspelling and lip-reading are all used for communication between deaf people. For many of them sign language communication over telephone lines, even with low visual resolution, is highly desirable.

Dramatic increases in performance of microprocessors have become the norm over the last few years and continued advancements show no end in sight. The turnaround time from microprocessor design to full production has decreased from 2 years in the late

1980s to 9 months today, allowing new designs to take advantage of the latest process technology. Pipelining and superscaling techniques also have produced large gains in performance. The result is a personal computer with enormous computational power.

While demonstrating impressive gains, modem bandwidth still has not kept pace with processor performance. Although computers can be produced with emerging high speed local area networking interfaces such as Asynchronous Transfer Mode (ATM) and Fibre Distributed Data Interface (FDDI), a typical modem provides at the most 28 kilobits per second (kbps) link using V.Fast modem technology. Because the bandwidth available from the telecommunications provider is low, video application users using low data rate telephone lines will continue to be faced with this limited bandwidth for a long time yet. Consequently, their system of computation is greatly slanted toward computation and away from communication. Hence recent developments in image communication are addressing the need for very low bit rate video coding through standards like H.263 and MPEG IV. There are many products available in the market today to make video telephony a reality using the low bit rate video coding standards. However, they suffer performance degradation due to lack of bandwidth, causing images to be produced with low temporal resolution that makes movements of objects very jerky.

Since deaf sign language communication is very sensitive to temporal resolution of images, these products are unusable over telephone lines. This demonstrates the need for a different approach. This approach must provide good representation of objects so that the expressions of faces and hand movements are recognizable. It should also provide low data rate to achieve the high temporal resolution that is very important for sign language

communication. One such approach is to extract two level sketches or cartoons from gray level images and code them efficiently for transmission over low bandwidth telephone lines. A cartoon representation is one in which emphasis is given to the location of object boundaries at the expense of shading and texture information.

Inexpensive desktop cameras can be used to keep the cost of the system low. However, video sequences captured from these video cameras are noisy. They produce a host of irregularities that result in image sequences corrupted by throbbing or pulsing noise. Hence, it is necessary to design an effective and efficient filter to reduce the throbbing noise and to improve signal quality for further image processing tasks involved in video telephony applications.

Considering the factors described above, this research is aimed to meet the following objectives:

- to provide a robust and simple solution for sign language communication using low resolution images and good motion representation.
- to provide a low cost and more efficient solution using inexpensive noisy cameras.
- to use the system on telephone lines which requires the image sequences to be coded at a low bit rate.

To meet the above objectives, an efficient feature extraction algorithm is employed to extract important feature points from the gray level image sequences and to construct recognizable binary cartoon images of the objects. An effective and efficient filter is de-

signed to reduce the throbbing or pulsing noise in the signal from the low cost camera. An efficient compression algorithm is employed on binary cartoon images to achieve low bit rates and hence make the system usable on ordinary telephone lines. A practical system is developed with a simple user interface under the MSWindows95 environment for transmission of binary cartoons over telephone lines, Internet and local area networks.

This thesis includes 8 chapters. Besides the introductory chapter, Chapter 2 develops the necessary background and discusses previous systems designed with the aim of transmitting binary sketches over telephone lines as well as current video telephony products. Chapter 3 explains the sketch generation process and adaptive techniques used for finding thresholds used in the sketch generation process. Chapter 4 explains the low frequency temporal noise from low cost cameras and filtering methods tried to reduce the noise. It also presents an effective temporal filtering algorithm for reducing the noise. Chapter 5 describes the compression methods available for binary images and presents an effective compression scheme. Chapter 6 provides implementation details of the practical system developed. It describes the user interface and discusses the programming issues involved in the system design. Chapter 7 discusses the results of subjective testing of the system developed, and Chapter 8 highlights the salient findings of this study and makes recommendations for future work in this area.

Chapter 2

Background

Sign language is commonly used among hearing impaired people to communicate non-technical subjects at about the same speed as hearing persons communicate with ordinary speech. The aim of the present research is to solve the problem of developing the transmission facilities that have been established for voice communication over the telephone network for the use of the deaf community. Though more emphasis is given to telephone network transmission, the system developed can be used with Local Area Networks and the Internet. There are some previous studies that have been done in the fields of television and picture coding related to deaf sign language communication. Studies in cognitive psychology using line drawings in visual object recognition, though not directly related to sign language communication, are also referenced in this chapter.

2.1 Studies in Television and Picture Coding

Sperling [1981] identified the minimum image spatial resolution necessary for signing and finger spelling. By plotting the percentage of correct transmissions versus the video bandwidth required for progressive reductions in picture size, Sperling concluded that a bandwidth of 21 kHz represented the limiting case. Below 21 kHz performance was found to be unsatisfactory. From the data given by Sperling, it can be estimated that a picture had approximately 16 pels/line and a size of 16×38 pels/frame or about 600 pels/frame. This pointed to the possibility of telephone bandwidth analog transmission,

but indicates that an additional 7:1 bandwidth compression is necessary to achieve this, since the bandwidth of a telephone line is around 3 kHz.

Experiments done by Pearson [1981] identified the minimum temporal resolution and bit rate necessary for sign language communication. From his experiments, it was realized that the possibility of data reduction by simple lowering of frame rate was very limited; since jerkiness introduced by reduced frame rate generated a rather serious form of distortion for the deaf. It was also reported that at 80×80 pels/frame, both signing and finger spelling were satisfactory. At 45×50 pels/frame, communication was reported to be difficult for use in signing. At this size, finger spelling was tested and found to be impossible. From his reports it is identified that 80×60 pels/frame at a rate of 12 frames per second, which is 100 kbps, is a reasonable candidate for the threshold between comfortable and difficult communication. The results obtained by Pearson[1981] and Sperling [1981] provided an estimation of feasible data reductions and acceptable degradation in spatial and temporal domains of the image for sign language communication.

2.2 Psychological Studies

Many studies have been made in the field of psychology to use line drawings in visual object recognition. Davies et al. [1978] stated that detailed line drawings of faces obtained from tracing the contour of photographs of familiar faces are very poorly recognized compared with the photographs from which the drawings were traced. However, recent approaches to visual object recognition emphasize the important role played by

"contours" in specifying an object's identity. Beiderman [1987] studied the theory of human image understanding. He stated that the object need not be presented as a full colored, textured image but instead can be a simplified line drawing. Under restricted viewing and uncertain conditions, such as when an object is partially occluded texture, color and other cues, such as position in the scene, may constitute part or all of the information in recognition. An example is identification of a shirt in the laundry pile from just seeing a bit of fabric. Surface characteristics such as color, brightness and texture will have only secondary roles in primary identification of the objects. Primary identification means the first contact of a perceptual input from an isolated, unanticipated object to recognition in memory.

Beiderman and Ju [1988] conducted experiments for measuring naming reaction times. The subjects identified brief presentations (50-100ms) of slides of common objects. Each object was shown in two versions: professionally photographed in full color and as a simplified line drawing showing only object's major components. In three experiments subjects named the object; in a fourth experiment a yes-no verification task was performed against a target name. Overall performance levels with the two types of stimuli were equivalent. Mean latencies in identifying images presented by color photography were 11 ms shorter than the line drawings. Error rate in identifying the objects is 3.9% higher than that of line drawings. These experiments support the theory that the earliest access to a mental representation of an object can be modeled as a matching of an edge based representation. Therefore such an edge based representation is sufficient for primary identification. The effectiveness of the above theory is further proved by the research done in the field of picture coding for deaf sign language communication.

2.3 Methods Used for Communication

Many methods have been tried and proposed by various researchers for effective sign language communication utilizing the related studies from the fields of television and psychological issues related to vision. Some of the methods are described below.

2.3.1 Television for the Deaf

Broadcast television can be used as a high bandwidth solution for deaf communication. Because of the high bandwidth nature it has more than sufficient temporal and spatial resolution to carry signing, lip reading and finger spelling. Television systems of this standard can be used for two-way communication between two deaf people or between one deaf person and a hearing person. Pearson[1981] describes in his paper an experimental operational in-house system installed at Royal National Institute for the Deaf in London, England. The block diagram of the system is shown in Figure 2.1 where the operator acts as an audio to video converter, repeating the message he/she receives from an incoming telephone call into the camera facing him/her, so that a deaf person can lip read the message.

Though the system was a natural and effective real time system, the drawbacks are:

i.) Since the system requires a high transmission bandwidth, it cannot be used for transmission over public switched telephone network and hence cannot be used for distance deaf communication.

ii.) The in-house video systems for the deaf can be linked over the public switched telephone network only with the assistance of two operators.

iii.) The system needs expensive television cameras and high bandwidth network and expensive maintenance.

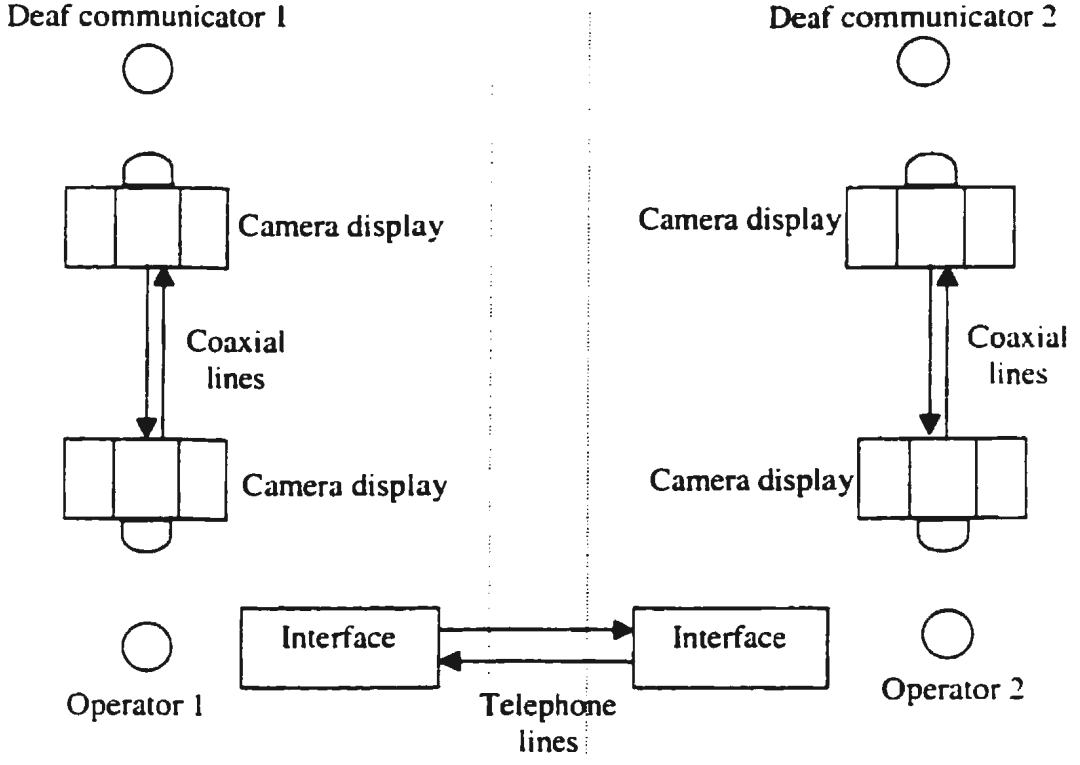


Figure 2.1 Video telephone system used at R.N.I.D

The above mentioned points suggest that, for sign language communication, a low bandwidth transmission system is needed.

2.3.2 Teletypewriters

Currently teletypewriters are the only reduced bandwidth solution for deaf communication. These are devices that enable a sender to transmit a typewritten message to a receiver who sees the characters displayed on a screen or produced on a teletypewriter.

They can be used conveniently for communication between two deaf people or between a deaf and a hearing person. However there are some disadvantages of these systems such as:

i.) The means of communication is not natural. In other words it is not the same as it would be if the two people involved were in the same room and consequently it is slow. Its slowness increases connection times over the switched network and therefore results in substantially higher charges than for spoken language for the same content.

ii.) Both sender and receiver must have necessary reading and typing skills.

This type of system is being used in many places, for example the North Dakota State University, USA uses a similar system to facilitate the learning of students who are hard of hearing.

2.3.3 Palantype

Palantype [Possum Controls Limited web page, 1998] is a method of machine short hand which has been in use for over 50 years in a variety of applications ranging from Court reporting and public inquiries. A highly trained Palantypist (called a speech to text reporter) records speech phonetically using a special keyboard at speeds in excess of 200 words per minute. Several keys are pressed simultaneously to form a "chord" similar to a piano. Each chord represents a syllable or a complete word or a phrase. Thus the palantypist is able to keep up with verbatim speech. The potential of a Palantype can be exploited by linking the keyboard to a computer aided transcription system. The computer can transcribe the shorthand notes simultaneously and display the resulting English tran-

script on a screen for hearing impaired people to read. Downton and Newell [1979] proposed that a single skilled Palantypist could be located at some central exchange serving the needs of several deaf and hearing persons. Communication to the operator is given in speech. The operator then converts the information into Palantype codes for a deaf person. This system is faster than teletypwriting and it is possible to use the telephone network for communication. However, communication between two individual hard of hearing persons is hindered. Reading the English transcriptions requires a level of educational attainment not possessed by all people. Also, privacy is excluded from the system because of the operator involvement

2.3.4 Methods Using Feature Extraction

Attempts have been made to study and extract key features involved in signing, fingerspelling and lip-reading and to code them for transmission at low data rates. Tartter and Knowlton [1981] developed an abstraction technique for hand movements. This involves placing 13 white spots on each hand and a reference white spot on the nose. Feature abstraction and data reduction were obtained in this scheme by lighting the hands in front of a television camera such that only the spots were visible in the image. The coordinates of the 27 spots were transmitted with a precision of approximately 1 percent along each axis, at 15 frames/sec. Communication was possible, with a data rate of 4800 bps. The drawbacks of this system are as follows:

- Though the deaf subjects could conduct a free flowing conversation in sign language, there were difficulties reported in finger spelling tasks.
- Lip reading is not possible.

- This is not a natural system. in that it abstracts hand movements for economy of representation.

Wallis and Pratt[1981] extracted features from images using a non-linear algorithm that can be described as a two dimensional automatic gain control followed by thresholding. This technique measures the local statistics of the image and based on the local activity. either amplifies the detail or suppresses it. The binary images are run-length coded for transmission over a channel at a rate of 9600 bps. They reported a temporal resolution of 8 frames per second to 2 frames per second. The spatial resolution of images tried were 120×120 pels/frame and 240×240 pels/frame. No tests using low data rate channels in a real time environment were reported in their paper. The system that was built used expensive special purpose processors to handle the image processing and communication stages.

In their paper "Telesign Project", Letellier and Nadler [1985] proposed extracting the edges in the images and transmitting only the binary edges. They used a very subjective quality criterion of natural appearance of edges and resemblance to the original scene. To extract the edges, they used a 7×7 Pseudo Laplacian edge detector. Edges are extracted from an original image size of 256×256 pels/frame and subsequently the image is subsampled for a reduced edge image size. With compression techniques such as block coding and run-length coding with ordering, they reported data rates of 63 kbps for an edge image of size 128×128 pels/frame at a frame rate of 25 frames per second. which is almost equivalent to broadcast temporal resolution. User responses were tested for sign-

ing and finger spelling for two edge sequences: one at a speed of 25 frames per second and size of 128 × 128 pels/frame, the other at a speed of 12.5 frames per second and size of 256×256 pels/frame. The recognition rate of the subjects was reported to be 98%. This project achieved promising results for sign language communication in terms of compression and feature recognition. The results also illustrated the importance of temporal resolution for sign language communication. However the disadvantages of the system are:

- It was not tested in a real time environment. Edge sequences were obtained and stored separately in videotapes for the subject evaluation. With the kind of compression rates reported, the authors could have tested it on a telephone line, though the line speeds obtainable at the time of research were low (of the order of 4800 bps).
- The computation time taken for processes like feature extraction and channel coding were not reported making suitability of this project as a real time project deployment difficult to determine.

Pearson and Robinson [1985] built a real time system for deaf sign language communication. Their analyses indicate that valley detection for extracting features from gray level image is superior to edge detection techniques. The authors substantiated this theory further, by comparing their proposed valley operator with seven other feature extraction operators. It was also found in their investigation that black fill-in generated by thresholding improved the representation of the cartoons and compression rates. Two-dimensional and three dimensional compression algorithms were tested. The experiments reported a data rate in the range of 4.8 to 9.6 kbps for images of size 64×64 pels/frame at

a rate of 6 frames per second. Two deaf volunteers were able to communicate for several minutes without strain using the experimental real time system. The advantages of the system were:

- It was a real time system that used very effective feature extraction algorithm using luminance valleys developed by the authors.
- Efficient compression algorithm used.

However, the authors used expensive television cameras and custom processors and the system was not tested on low bandwidth telephone lines. It should also be noted that the tests were done in good lighting, plain background and the subjects wore plain clothing. The research work presented in this thesis can be considered a considerable extension of the previous studies done in mid 1980s and uses current modem and camera technologies to develop an economical system that can be used by the deaf.

2.3.5 Methods using animation images

Xu et al. [1993] proposed a system in which only information concerning the motion of the object is transmitted to provide a model with motion parameters and an animation image is synthesized from the information. This approach consists of three stages:

i.) Hand motion parameters should be extracted using the data glove to represent the motion of the upper limb.

ii.) A sign language word dictionary is used to extract the concepts and semantic information contained in the sign language from the motion parameters.

iii.) The received semantic information is analyzed and converted into motion parameters reflecting the features of the hand motions and used to synthesize animation image.

The paper is concerned primarily with the techniques used in stage iii.). The authors did not develop a system for communication. Sign language recognition including body motion and facial expressions are also not studied in this paper. Though the proposed approach is interesting, it is not a natural system and requires a high-speed computer graphics system to perform image synthesis.

Akoi et al. [1994] transmitted coded text data as the source code of the image data. At the receiving end, the images are synthesized from the text data using computer graphics techniques. In their experimental setup, the authors used two personal computers: one for sending text data and the other for synthesizing images. The transmission speed of the communication system was 64kbits/sec. The authors were able to transmit sign language images in half-duplex mode. The drawbacks of the systems are:

- Designing a complex dictionary to describe all the gestures used in sign language communication.
- Requires a high-speed computer graphics system to decode text data and perform image synthesis.
- The user of the system needs typing skills.
- The system is not a natural system. In other words it is not the same as it would be if the two people involved were in the same room

- It is also not clear how the speed synchronization of hand, arm and fingers movements is achieved during image synthesis to correctly generate the sign language gestures.

2.4 Standards Related to Video Telephony

Recent technological developments in the field of multimedia communications are making desktop video telephony/conferencing a reality today. It should be noted that the systems available currently are aimed at providing multimedia communications for common people, with no disabilities, who can tolerate low temporal resolution of images obtained over low bandwidth telephone lines. However, if the communication bandwidth available to common users increases, for example from low bandwidth analog telephone lines to high bandwidth ISDN (Integrated Switched Digital Network), these technologies can be used for sign language communication.

To avoid compatibility problems between different vendor products, the industry is promoting International Telecommunication Union (ITU) standards. ITU addresses the industrial compatibility problems on multimedia teleconferencing products by H.32x standards. H.324 addresses and specifies a common method for sharing video, data, and voice simultaneously using high-speed (V.34) modem connections over a single analog telephone line. It also specifies interoperability under these conditions, so that videophones, for example, based on H.324 will be able to connect and conduct a multimedia session. Of the three ITU standards that address videoconferencing - H.324, H.323 and H.320, H.324 has the broadest impact in the marketplace. That is because H.324 incorpo-

rates the most pervasive communications facility, analog telephone lines installed today, on a global basis. For reference, H.320 specifies videoconferencing over circuit-switched media like ISDN and Switched 56, while H.323 extends H.320 video to corporate Intranets, LAN's and other packet-switched networks. As a result, H.324 based products are expected to be prominent in the mass market/retail segment, where PC's equipped with this capability are already available. The H.324 suite consists of five recommendations: H.324, H.223, H.245, H.263 and G.723.1 (formally G.723).

H.324	Combination of following standards.
H.263	New video codec (coder-decoder) for analog telephone lines and packet-switched networks.
G.723.1	Low-bandwidth audio codec for voice communications up to 3.1 kHz: can use 5.3 or 6.3 kbps of bandwidth.
H.223	Specifies mechanisms for synchronizing data and bundling it in packets on packet-switched networks.
H.245	Specifies control protocols and signaling procedures for managing audio/video streams.

2.4.1 Commercial Multimedia Teleconferencing Products

There are many popular teleconferencing products available from the market nowadays. To mention a few, Cu-Seeme from WhitePine Software, NetMeeting from Microsoft, Proshare from Intel, LiveLan from PictureTel and many of other vendors. The average system solution from the above vendors costs around $300.00 to $500.00. Apart from software solutions there are standalone hardware solutions too. An example is video-

phone products from C-Phone Corporation. All these products can transmit video, audio and data over telephone lines, local area networks and Internet, using H.324 standards and are interoperable. C-Phone [C-Phone web page, 1998] claims that their videophones can be used for deaf sign language communication over telephone lines. This may be attributed to it using high-speed RISC digital signal processors from Lucent Technologies for video compression and quality cameras. This product costs approximately $1000.00. Though C-Phone claims use of their products over telephone lines, many of the news letters published by the company state that the product can be used more effectively over high bandwidth digital ISDN lines than telephone lines, which are not commonly available.

Though there are many efficient products with software and hardware solutions, the fact remains that the best bandwidth available from analog telephone lines is 28 kbps. H.263 video compression is mainly based on motion compensation and it is difficult to realize the effectiveness of these videoconferencing solutions for deaf sign language communication, where portions of frames change significantly from the preceding frame. In fact, it is gathered from various sources on the net that the overall performance of many of the commercially available products over telephone lines is close to a frame per second, which makes movements of objects jerky. The low temporal resolution obtained from these products over analog telephone lines, makes them unsuitable for deaf sign language communication and motivated this research.

Chapter 3

Sketch Generation

Cartoon or line drawing is a method of portraying a person or a scene. Removal of texture information from the images allows economical transmission of binary sketches over low bandwidth telephone lines, while maintaining a reasonable temporal resolution. Therefore, transmission of line drawings through low bandwidth telephone lines can be used effectively for sign language communication. The line drawings or binary sketches are extracted from gray level images. The feature extraction technique employed for generating binary sketches from gray level images should satisfy two requirements. The first requirement is that it should portray satisfactorily the face and the hands of the person concerned. The second requirement is that it should do so with as few lines as possible. The data to be transmitted increases with the number of lines. Since the research is aimed at using narrow bandwidth analog telephone lines, extreme economy is needed in the cartoon generation. This chapter discusses different sketch extraction algorithms and the reasons for the selection of a particular feature extraction operator [Robinson, 1985], used in this research. Though there is no modification to the basic algorithm of feature extraction employed by Robinson, the present work differs by proposing new methods that make the chosen algorithm adaptive to the statistics of the image, resulting in a simple user interface.

3.1 Feature Extraction Methods

The theories behind various methods employed for meaningful extraction of features from images and their advantages and disadvantages are discussed below. It is noted that two different approaches for extracting features have been taken by researchers. The approach used by Marr and Hildereth [1980] was based on the response of an edge to differential operators. The other approach, by Pearson and Robinson [1985] was to formulate the rule for drawing cartoons in the three dimensional space of the object rather than the two dimensional space of the image. This resulted in an operator whose response is primarily sensitive to image valleys and secondarily sensitive to edges. This section also describes the operator chosen for feature extraction, along with its advantages and disadvantages and methods employed to make the sketch generating system adaptive.

3.1.1 Edge Detection

Edge detection is by far the most common approach for detecting meaningful features from gray level images. An edge is a sudden change in the luminance of the image. The idea underlying edge detection is that a sudden intensity change will give rise to a peak or trough in the first order derivative or equivalently to a zero crossing in the second order derivative. A zero crossing is a place where the value of a function passes from positive to negative. Figure 3.1 shows schematically the luminance profile of an edge and its first and second derivatives.

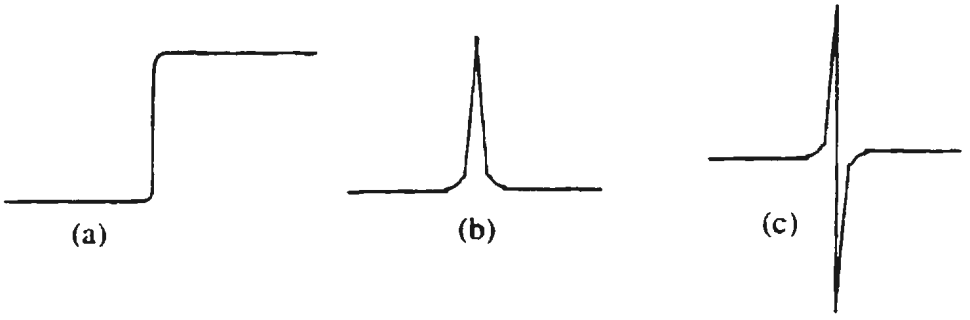


Figure 3.1(a) Luminance profile of an edge. (b) first and (c) second derivatives of an edge.

This idea suggests that in order to detect intensity changes efficiently, the filter should be a differential operator, taking either first or second order spatial derivatives of the luminance distribution of the image.

3.1.1.1 First Order Differential Operators

The most common method of differentiation in image processing is the gradient. For a function $f(x, y)$, the gradient of f at coordinates (x,y) is defined as the vector

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \qquad (1)$$

The magnitude of this vector,

$$\nabla f = mag(\nabla f) = \left[\left(\frac{\partial f}{\partial x} \right)^2 + \left(\frac{\partial f}{\partial y} \right)^2 \right]^{1/2} \qquad (2)$$

is the basis for various approaches to image differentiation. Consider the image region shown in Figure 3.2(a), where the z's denote the values of gray levels. Equation (2) can be approximated at point z_5 in a number of ways. The simplest one is to use the difference

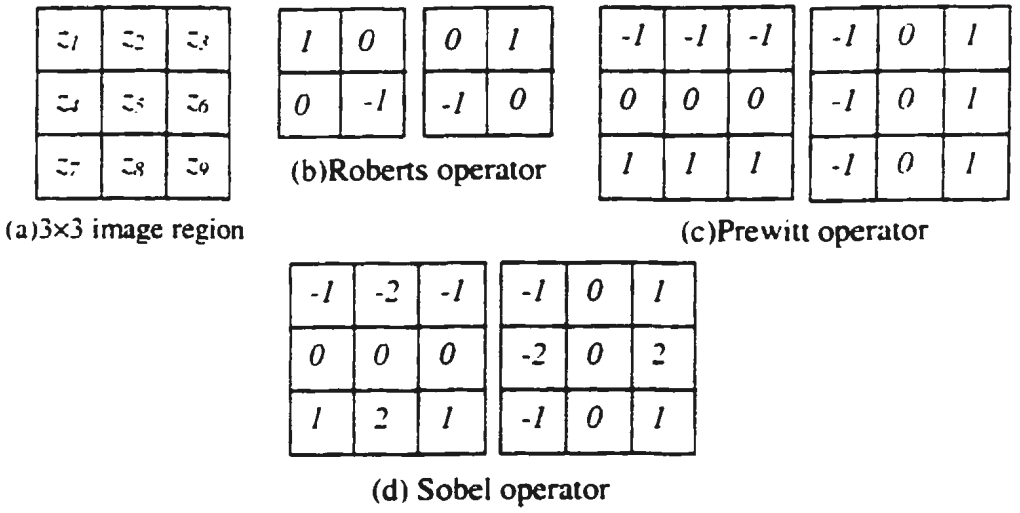


Figure 3.2 A 3×3 image region and various operators used to compute the derivative at a specified point.

(z_5-z_8) in the x direction and $(z_5 - z_6)$ in the y direction combined as

$$\nabla f \approx \left[(z_5-z_8)^2 + (z_5-z_6)^2\right]^{1/2} \qquad (3a)$$

Instead of using squares and square roots, similar results can be obtained by using absolute values:

$$\nabla f \approx |z_5 - z_8| + |z_5 - z_6| \qquad (3b)$$

Another approach for approximating ∇f is to use cross differences:

$$\nabla f \approx \left[(z_5 - z_9)^2 + (z_6 - z_8)^2\right]^{1/2} \qquad (4a)$$

Or using absolute values.

$$\nabla f \approx |z_5 - z_9| + |z_6 - z_8| \qquad (4b)$$

Equations (3) and (4) can be implemented by using masks of size 2×2. For example Equation (4b) can be implemented by taking the absolute value of the response of the two masks shown in Figure 3.2 (b) and summing the results. These masks are called Roberts Cross-Gradient operators.

An approximation to Equation (2), at point z_5, using a 3×3 neighborhood, is obtained as:

$$\nabla f \approx |(z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)| + |(z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)| \qquad (5)$$

The difference between the third and first row of the 3×3 region approximates the derivative in the x direction, and the difference between the third and first column approximates the derivative in the y direction. The masks shown in Figure 3.2(c) are called the Prewitt operators, and can be used to implement Equation (5). Figure 3.2(d) shows yet another pair of masks called the Sobel operators for approximating the magnitude of the gradient. The Sobel operators have the advantage of providing both a differencing and a smoothing effect. Because derivatives enhance noise, the smoothing effect is a particularly attractive feature of the Sobel operators. From Figure 3.2(d), the derivatives based on the Sobel operator masks are

$$f_x = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3) \qquad (6a)$$

$$f_y = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7) \qquad (6b)$$

where z's are the gray levels of the pixels overlapped by the masks at any location in an image. Studies by Letellier [1985] and Robinson [1985] showed the poor performance of first order differential operators in rendering image features.

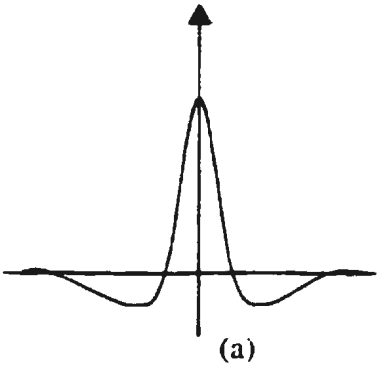
3.1.1.2 Second Order Differential Operators

Marr and Hildereth [1980] suggested an operator $\nabla^2 G$, where ∇^2 is the Laplacian operator and G is the two dimensional Gaussian distribution with standard deviation σ.

Let $r^2 = x^2 + y^2$. Then,

$$\nabla^2 G = \left(\frac{r^2 - \sigma^2}{\sigma^4} \right) \exp\left(-\frac{r^2}{2\sigma^2} \right) \qquad (7)$$

Figure 3.3(a) shows the one-dimensional form of this circularly symmetric function. From the figure, the smoothness of the function, its zero-crossings at $r = \pm \sigma$, and the positive center and negative skirts can be seen. When viewed in 3D perspective with the vertical axis corresponding to intensity, Equation (7) has a Mexican hat shape.



(a)

0	-1	0
-1	4	-1
0	-1	0

(b

Figure 3.3 (a) One-dimensional form of $\nabla^2 G$ (Laplacian of Gaussian). (b) Marr and Hildreth operator for a kernel size of 3×3.

The choice of $\nabla^2 G$ by Marr and Hildereth for effective edge detection is based on two ideas:

- The Gaussian part, G, of the above operator blurs the image, effectively wiping out all of the structure at scales much smaller than the space constant σ of the Gaussian and since the Gaussian distribution is optimally localized in both spatial and frequency domains, blurring is smooth in both the domains.
- The operator is isotropic or orientation independent which requires less computational burden than the directional operators.

For a 3×3 region, the Laplacian of a 2D function $f(x,y)$ with $\sigma = 4$ is defined as

$$\nabla^2 f = 4z_5 - (z_2 + z_4 + z_6 + z_8) \qquad (8)$$

where z's have been defined already. The basic requirements in defining the digital Laplacian are that the coefficient associated with the center pixel should be positive and the coefficients associated with the outer pixels should be negative. Because the Laplacian is a derivative, the sum of the coefficients has to be zero. Hence the response is zero whenever the point in question and the neighbors have the same value. Figure 3.3b shows a spatial mask that can be used to implement Equation (8). To detect zero crossings the image has to be convolved with $\nabla^2 G$, the result has to be converted to binary to simplify detection of zero crossings. These steps increase computational complexity and time, though zero crossings offer an efficient solution for feature extraction.

However, isotropic operators are highly sensitive to low-amplitude high spatial frequency noise. One non-linear operator proposed by Nadler [1985], which reduces the sensitivity to noise and maintains Laplacian characteristics, is the Pseudo Laplacian Operator.

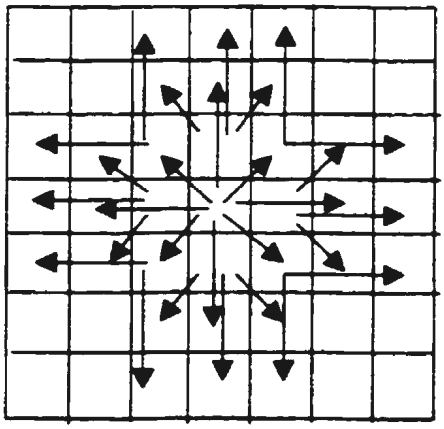


Figure 3.4. Schematic representation of 7×7 Pseudo Laplacian.

The algorithm of the operator is as follows: Consider the 7×7 configuration of the arrows shown in Figure 3.4. Each of these arrows joins two pels and represents the difference

between the pel intensities at the head and the foot of the arrow. Each difference is compared in absolute value to a certain threshold T_1 and if it is greater than that value, the sign of the difference is retained. The positive and negative signs are then counted separately. The edge decision is taken if the following conditions are satisfied:

$$C(+) > T_2$$

$$C(+) - C(-) > T_3$$

where C(+) and C(-) are the counts of positive and negative signs, respectively, and T_2 and T_3 are thresholds. This operator is effective in extracting features. However, due to the window size used by this operator, more processing power is required. Hence it is inefficient for real time sketch extraction.

3.1.2 Valledge Detection

An entirely new approach is used by Pearson and Robinson [1985]. This new approach was to formulate the rule for drawing cartoons in the three-dimensional space of the object rather than the two-dimensional space of the image, constrasting with the ideas of Marr and Hildreth. According to the authors, a cartoon point (usually forming part of a cartoon line) should be drawn in the two dimensional image plane wherever a line grazes the surface of the three dimensional object as shown in Figure 3.5. The basic postulate in the proposed theory of cartoon identifies surfaces that are grazed by straight lines drawn from the camera. When the image of the object is scanned, the computer has to identify points A, B and C, in Figure 3.5, as cartoon points. It has been found that the image feature produced at a surface satisfying the above postulate is a luminance valley. However to accommodate luminance changes in the features like the edges of the noses, they sug-

gested that the operator should have primary response to valleys and secondary response to edges. The one-dimensional spatial response of the suggested operator is shown in Figure 3.6a. The detector responds to an edge with a peak at the foot of the edge as shown in Figure 3.6b. Second derivative operators combine both valley and edge detection that the authors used for successful feature extraction. This contrasts with the

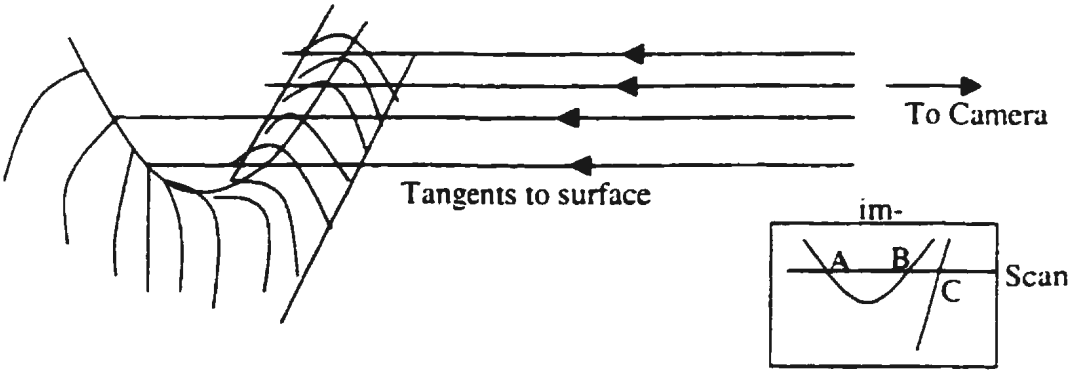


Figure 3.5 Schematic representation of cartoon theory.

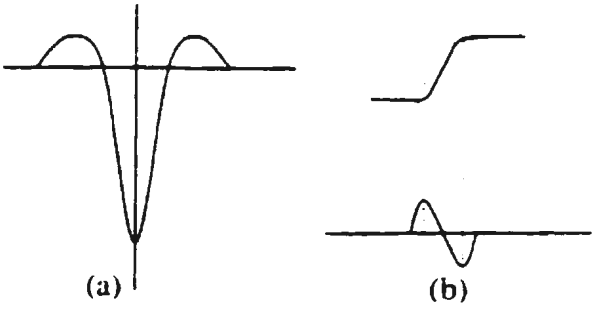


Figure 3.6 (a) Spatial response of a convolutional valley operator (b) Response of a valley operator to an edge.

more conventional use of the directional second derivative and Laplacian in feature extraction, where zero crossings are used to locate the steepest points of edges. Algorithms of three second derivative Laplacian operators suggested by the authors are described below. In all the operators, feature extraction is combined with absolute level thresholding, where all areas below the main threshold are rendered black, which makes significant

areas of darkness (e.g. black hair) filled in, rather than in outline. This approach improved image representation and helped in reducing data rates.

3.1.2.1 5×5 Logical Valley Operator

This operator performs a series of computations in four orientations. If a valley is detected in any of these orientations, a feature point is marked for valley detection. The algorithm begins by checking the simple difference *(l-m)* and *(n-m)* (see Figure 3.7). Provided one of these is over the threshold T_1, the search for a valley proceeds. This initial step enables rapid elimination of unlikely points. Three element blocks parallel to the searched-for valley are then box-filtered *((f+k+p), (g+l+q), etc.)*. Simple valley convolution is applied to these blocks at two different scales. If the output exceeds a threshold (T_2) for the larger scale convolution (involving blocks *(f+k+p), (h+m+r), and (j+o+t)*), small-scale convolutions over the three central blocks are used for peak detection/nonmaximum suppression. The use of three different scales provides protection against spurious detection due to noise. The algorithm to test for a valley through a point in an image region is as given below.

To test for a valley through *m* in Figure 3.7, where *a* to *y* are pels:

if $((l-m) > T_1$ *or* $(n-m) > T_1)$

then

if$(f+k+p+j+o+t - 2(h+m+r)) > T_2$

and $(g+l+q+I+n+s - 2(h+m+r)) > (f+k+p+h+m+r - 2(g+l+q))$

and $(g+l+q+I+n+s -2(h+m+r)) > (h+m+r+j+o+t -2(I+n+s))$

then there is a valley through m.

a	b	c	d	e
f	g	h	i	j
k	l	m	n	o
p	q	r	s	t
u	v	w	x	y

Figure 3.7. Example 5×5 image region

T_1 and T_2 are thresholds. Tests for horizontal valley and diagonal valleys have the same form, but different thresholds. Though this operator is very effective in extracting fine details, the algorithm is computationally intensive which makes it not effective for real time cartoon generation.

3.1.2.2 3 ×3 Logical Laplacian Operator

In this operator nonlinearities are introduced in lookup tables, in order to reject small differences (which might otherwise accumulate) and amplify large differences. The use of two different lookup tables is to afford approximate isotropy (otherwise diagonal, horizontal, and vertical contributions would be differently weighted).

To test for a high Laplacian at e in Figure 3.8a:

value = $L_1(a\text{-}e) + L_2(b\text{-}e)+L_1(c\text{-}e)+L_2(d\text{-}e)+L_2(f\text{-}e)+L_1(g\text{-}e)+L_2(h\text{-}e)+L_1(I\text{-}e)$

If value is greater than the threshold a feature point is detected. $L_1(.)$ and $L_2(.)$ are nonlinear functions of difference implemented as look-up tables; they have shape as shown in Figure 3.8b. This operator is simple and produced effective cartoons.

3.1.2.3 3×3 Edge Counter Operator

To test for a feature at e in Figure 3.8a:

Count = 0

if $(a\text{-}e) > T_1$ Count = count + 1

if$(b\text{-}e) > T_1$ Count = count + 1

•

•

if $(i\text{-}e) > T_1$ Count = count + 1

if(count > T_2) a feature point is detected. T_1 and T_2 are thresholds.

This operator is also simple and used for effective real time sketch extraction.

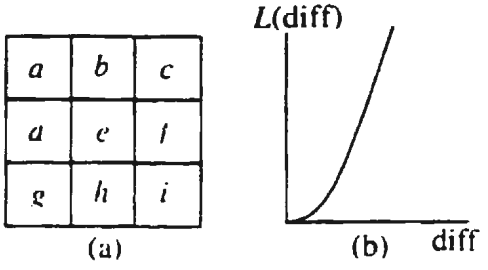


Figure 3.8(a) Example 3×3 image region. (b) Non-linear function used in Logical Laplacian operator.

3.1.3 Efficiency of Valledge Detectors

In their research Pearson and Robinson compared the performances of operators sensitive to edges and operators sensitive to valleys. It is found from their results that operators such as the Pseudo Laplacian operator and valledge operators, which are maximally sensitive to valleys as opposed to step edges, are more successful in extracting the essential

facial features. The 5×5 valledge detector proposed by authors produces thinner lines and is subjectively better at resolving fine details.

In their paper, Hanna and Bruce [1992] reported experiments, which examined how well full cartoons of famous faces could be recognized compared with gray level images and assessed the separate contribution played by each of the valledge components and threshold components. Fifteen famous faces were digitized (518 × 316 pixels at 256 gray levels) and the cartoon algorithm proposed by Pearson and Robinson was applied, with valley width and threshold level adjusted by eye to produce what looked to be the best representation possible from the image. Separate valledge and threshold images were produced using the same parameters for each face as in full cartoon condition. Expressed as a percentage of the faces that were identified from full images, subjects were able to correctly identify 93.3% full cartoons. This result shows that full cartoon drawings produced by valledge operator are identified almost as accurately as gray level images, which substantiates the selection of the valledge operator for extracting features in this research.

For the present work, the 3×3 edge counter operator is chosen for feature extraction. Though inferior to the 5×5 valledge detector, the reasons for choosing this operator are:

- Its primary sensitivity to valleys and secondary sensitivity to edges attributed to its Laplacian structure.
- Simple and easy to implement and requires less processing power.

- It is used in a real system [Pearson and Robinson, 1985], built for testing the efficacy of valley detected cartoons. It is reported that deaf subjects were able to communicate without much strain, which proves the feature extraction efficiency of the operator.
- It is more economical than 3×3 logical Laplacian operator in terms of data rate [Robinson,1986].

Figure 3.9 A sample frame and its binary sketch using manual thresholds.

Figure 3.9 shows a sample frame and its binary sketch generated by the 3×3 edge counter algorithm. Thresholds used in generating the sketch are manually adjusted for better representation. Noise from low cost desktop cameras adds to this operator's sensitivity to noise, which comes from the operator's Laplacian nature and small window size. The next chapter describes the methods tried for reducing the throbbing noise in the original image sequences and provides a solution through a simple recursive temporal filter.

3.2 Threshold Adaptivity

The edge counter algorithm for feature extraction [Robinson, 1986], uses three thresholds T_1, T_2 and an absolute threshold T_3, which makes it difficult for ordinary users to tune the extraction operator for optimum performance. This section explains new methods found

in this research, which are used to make the sketch generation system learn adaptively from the image statistics to decide on the value of thresholds to be used. It should be noted that this new work enables the replacement of image dependent threshold parameters used by Robinson, by image independent threshold parameters, which can be set by the user once and for all. It therefore simplifies the user interface.

3.2.1 Absolute Threshold

Threshold T_3 is used for absolute level thresholding to improve the image representation. The pels that lie below this threshold are made black and those above the threshold are

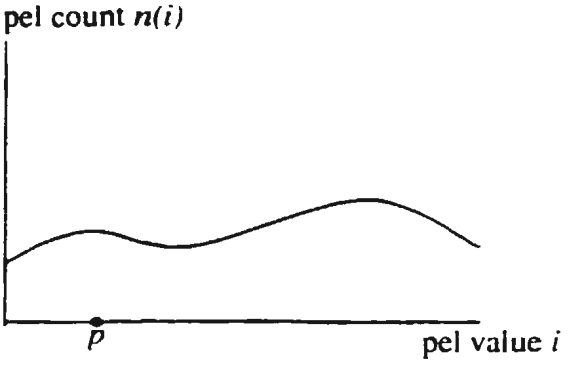


Figure 3.10 A sample image histogram.

made white, resulting in black fill-in areas. Hence, the threshold effectively divides the image into two levels, one above the threshold and one below the threshold. The point of division for good representation depends on brightness and contrast of the image. Since these two characteristics of the image are well represented by histograms, a simple experiment is performed using histograms to study the effect of absolute threshold value. The experiment is explained below.

- Individual histograms are drawn in real time for each image in the sequence as shown in Figure 3.10. The value p of threshold T3, adjusted manually for better representation, is noted.

From the above experiments, it is found that the value of p is the image pixel value on the histogram where the cumulative sum of the count of pixels starting from pixel value zero is a fraction of the total count of pixels in the image. This information defines the algorithm for adaptive calculation of absolute threshold as given below.

1. Total number of pels in the image is counted as T_c.

2. Value p is calculated from T_p, where T_p is defined as,

$$T_p = \sum_{i=0}^{p} n(i) \quad \text{such that . } T_p \cong (1/\alpha)T_c$$

The value of α decides the threshold value p. This method tries to keep the threshold value adaptive to the changes in contrast and brightness. A high p value results in more black- fill in areas and a low p value results in fewer black fill-in areas. For a fixed value of α, an increase in brightness of the image increases the value of p and a decrease in brightness decreases the p value. Changes in contrast produce changes in the range of pixel values present in the image. This results in change of value p. This method will not work well for scenes where there are many moving objects and the amount of light reflected from each object surface is different. This difference in light reflection is due to the distance of the objects from the camera and the amount of light falling on the objects. This kind of an image scene produces multi modes in the shape of the histogram. A value of threshold calculated using this histogram might represent some objects in the scene well but not all the objects. This drawback makes the method unsuitable for adaptive cal-

culation of threshold for all types of scenes. However, this method is found to be good for scenes where there is single moving object with constant background. This is the case for sign language communication where the subject sits in front of the camera without changing the background. This makes this algorithm suitable for the intended application. From the experiments an α value of 3 is found to improve subjectively scenes with single moving objects.

3.2.2 Thresholds for Valledge Detection

Thresholds T_1 and T_2 are used for extracting feature points from the image. To detect a feature point at e (see Figure 3.8(a)), theshold T_1 and T_2 should be set in such a way that there be greater than or equal to T_2 first order directional differences or valleys which are greater than T_1. It is found from the experiments that T_2 can be fixed for a value 1 for all types of scenes. Increasing the value of T_2 to a value greater than 1 resulted in reduced number of feature points and reducing the value from 1 resulted in picking up more noise information because of Laplacian sensitivity to noise. Hence it is decided to keep the value of threshold T_2 as 1. However, it is found from the experiments that a fixed value T_1 is not suitable for varying scene conditions. This prompts the need for finding an algorithm that can adaptively calculate T_1. Since T_1 depends on first order differences or valleys, a simple experiment is performed using difference histograms to study the effect of threshold value T_1. The experiment is explained below.

- For each center pel in the 3×3 window shown in Figure 3.11(a) of the original image, differences or valleys are calculated as a-*e*, *b*-*e*, *c*-*e*, *d*-*e*, *f*-*e*, *g*-*e* and *i*-*e* and a difference histogram is built from the data calculated as shown in Figure 3.11(b).

- The value v of threshold T1, adjusted manually for better representation, is noted.

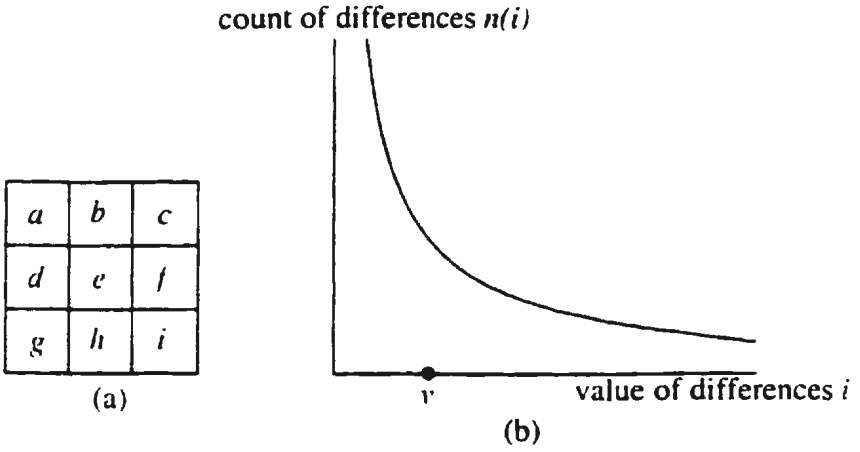


Figure 3.11. (a)Example 3×3 image region. (b)A sample difference histogram

From the above experiments, it is found that the value of v is the image pixel value on the difference histogram at which point the cumulative sum of the count of differences starting from pixel value 255 is a fraction of the total number of differences for all pels in the image. This information defines the algorithm for adaptive calculation of threshold T_1 as given below.

1. Total number of differences for all pels in the image is counted as T_d, where

   T_d = Total number of pels × 8

2. Value v is calculated from T_v, where T_v is defined as

$$T_v = \sum_{i=255}^{v} n(i) \quad \text{such that, } T_v \cong (\beta/100)T_d$$

The value of β decides the threshold value v. An increase in value v results in locating fewer feature points and a low value of v results in more feature points. The value of v depends on the brightness and contrast of the scene. Since the change in brightness does not change the first order differences, this algorithm is resilient to brightness changes.

However, since contrast changes do affect first order differences, this algorithm will not perform well for scenes where there are significant contrast changes. A value of 5 for β is found to improve subjective quality of scenes with single moving objects.

Figure 3.12 shows sample frames and their binary sketches using adaptive and manual thresholds. Figure 3.12(a) shows three frames with different scenes. They are:

(i) Two subjects some distance away from the camera.

(ii) One distant subject in a scene with less overall brightness.

(iii) One distant subject in a scene with more brightness than (ii).

(iv) One subject close to the camera.

Figure 3.12(b) shows the binary sketches of the frames using manual thresholds with the thresholds adjusted for best visual appearance of the subjects. Figure 3.12(c) shows the binary sketches of the frames using a manual absolute threshold adjusted for best appearance and as adaptive feature extraction threshold. Figure 3.12(d) shows the binary sketches of the frames using adaptive absolute and feature extraction thresholds. Figure 3.12(c) shows that the results from using the adaptive feature extraction threshold is comparable with that of the manual absolute threshold adjustments because of its resilience to brightness changes.

As can be seen in Figures 3.12.d (i), 3.12.d (ii) and 3.12.d (iii), the absolute threshold does not adapt for all scenes. This is because of the difference in the light falling on the distant subjects and its limited sensitivity to adapt to the changes in brightness and con-

trast in a scene. However, the adaptive absolute threshold works well for scenes containing objects closer to the camera and under uniform lighting, which can be observed from Figure3.12.d (iv). Since, in sign language communication, the subjects sit close to the camera under uniform lighting, the adaptive method for absolute thresholding works

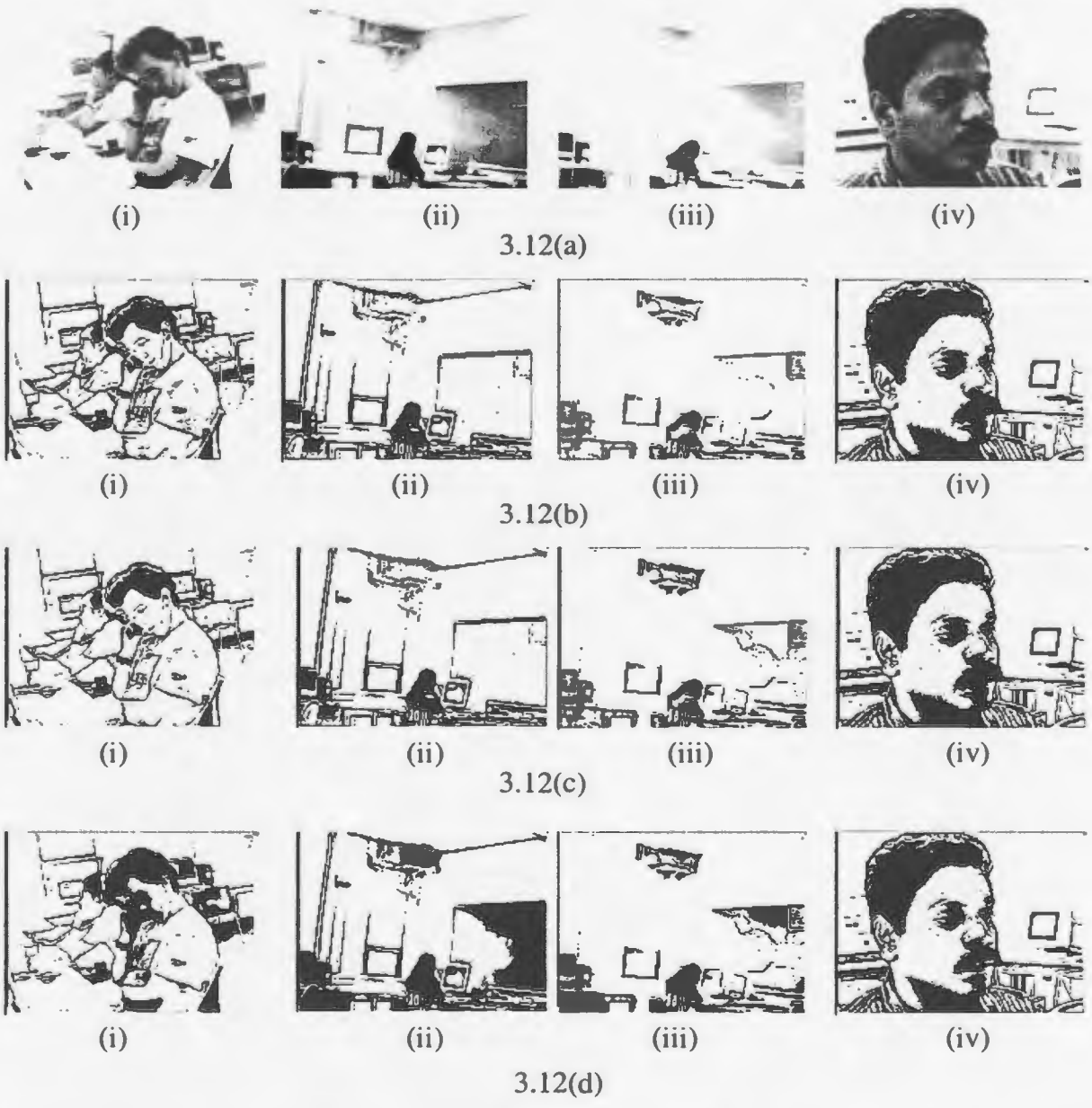


Figure 3.12(a) Sample frames. (b) Binary sketches of the frames using manual thresholds. (c) Binary sketches using manual absolute threshold and adaptive feature extraction threshold. (d) Binary sketches using adaptive absolute and feature extraction thresholds.

effectively. It should be noted that the values selected subjectively for a particular scene condition would not work well if the scene condition changes significantly. For example if the scene changes from one moving object to many moving objects or if the lighting conditions change (thus changing brightness and contrast), the user will need to re-adjust the values of α and β subjectively for adaptive absolute and feature extraction thresholds.

3.3 Summary

This chapter explains the different first and second order differential operators used in feature extraction and identifies a suitable and efficient feature extraction operator. This operator is primarily sensitive to the valleys and has secondary sensitivity to edges in the image and uses three variable thresholds. This chapter also proposes new methods for choosing the thresholds adaptively from the global image statistics. These adaptive methods help in simplifying the user interface.

Chapter 4

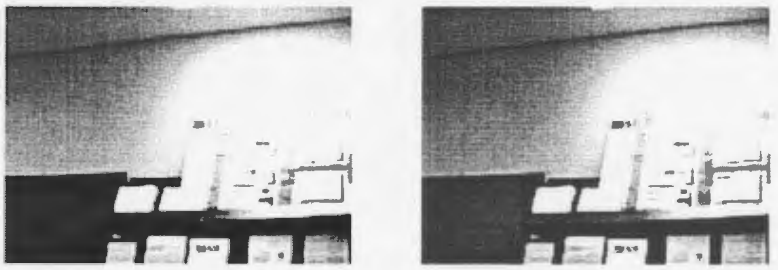
Noise Resilience

One of the limiting factors in image processing is noise. In the design of any image processing system, one must consider the noise performance of the system, especially the manner in which the noise interacts with the visual appearance of the picture and perturbs the subsequent digital processing. Although the visual effects are of importance, they are secondary to later image processing stages like filtering, and compression of images. In particular, inexpensive low cost cameras produce many irregularities that result in image sequences corrupted by throbbing or pulsating noise. This chapter explains the source and property of the noise from low cost desktop video cameras and different filtering methods employed to reduce the effects of noise. An effective, simple recursive temporal filter is designed and compared with the other filters.

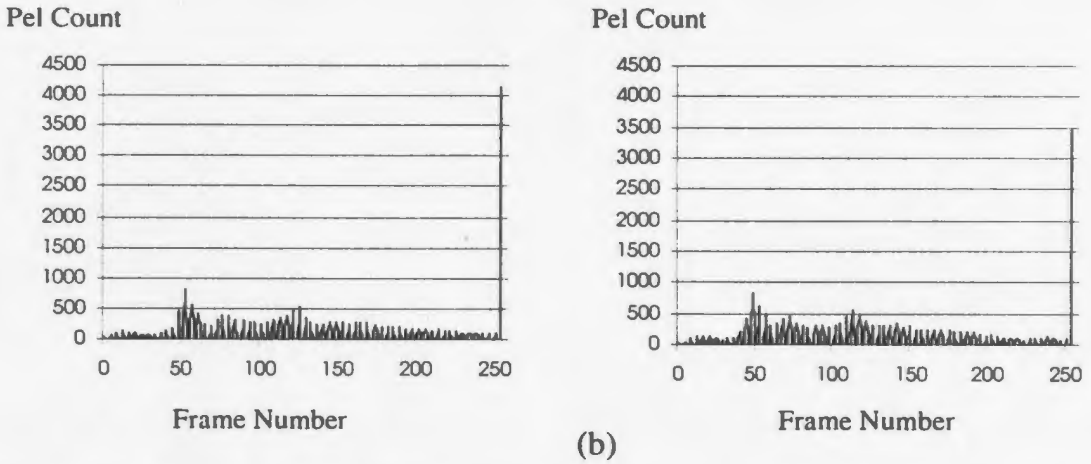
4.1 Sources and Properties of Noise

Low cost video cameras have become widely available for desktop video applications. These have several differences from the more traditional television cameras used in earlier computer applications. These cameras connect to the computer via a serial or parallel port rather than through a video input on a frame grabber. The frame rate depends on the

processing speed and available light rather than on a fixed synchronized clock. These have only simple control of camera parameters like black level and focus. One of the most noticeable properties of such cameras is their tendency to produce a bouncing or throbbing signal, where the overall picture luminance varies at low frequency (of the order of 1 Hz). This comes from a variety of causes like the lack of black level clamping, beating between the camera frequency and the supply frequency and poor gain control. The effect is both multiplicative and additive, as demonstrated in the experiments below, and presents a kind of spatially homogenous low frequency temporal noise.



(a)



(b)

Figure 4.1 (a) Successive frames from a test sequence. (b) Histograms of successive frames of the test sequence

Two successive frames from a test sequence without any motion of the objects are shown in Figure 4.1(a), to give a better idea of property of the throbbing noise. The change in

the size of the half-circled light region in Figure 4.1 shows how the throbbing noise affects the image sequence. Histograms of the frames from the picture sequence are shown in Figure 4.1(b). From these histograms, it can be observed that there is a major shift in pel counts from one value to another as frames are progressively captured from the video device. Figure 4.2(a) shows the changes in the average global intensity of the frames for a test sequence without any moving objects. Figure 4.2(b) shows the variation in standard deviation of average intensity of the frames in the same sequence.

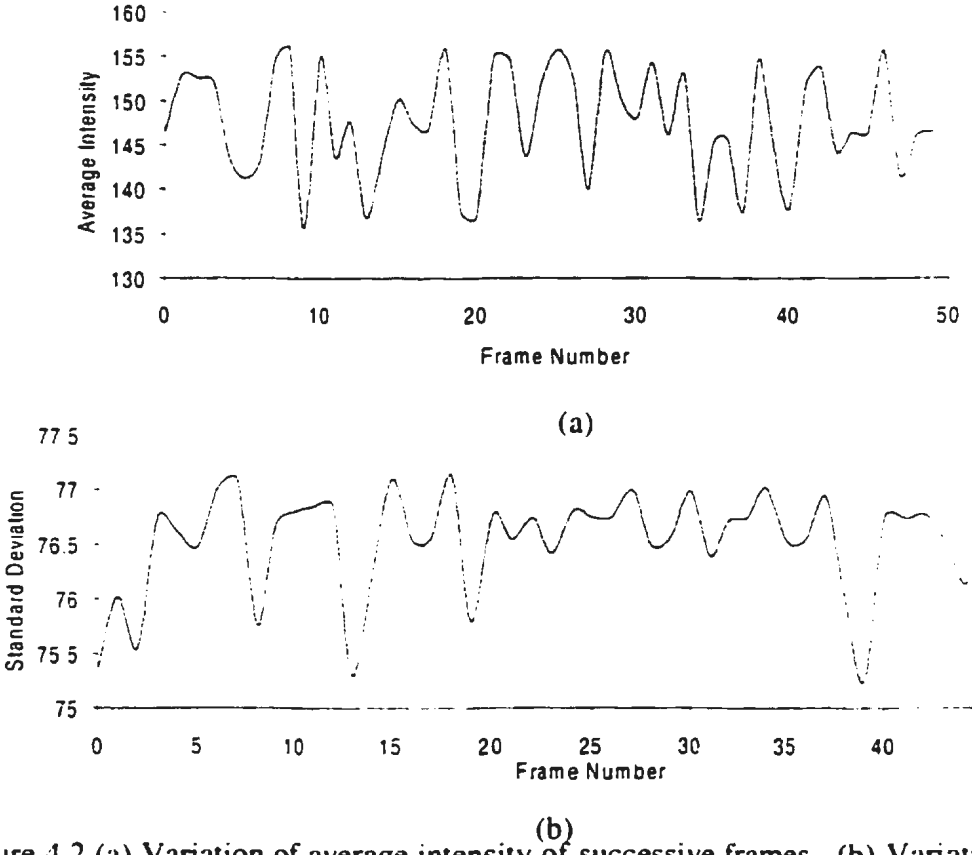


Figure 4.2 (a) Variation of average intensity of successive frames. (b) Variation of standard deviation of average intensity of successive frames

Let $v_t(x,y)$ denote the intensity of a pel at sample point (x,y) and $v_{t-1}(x,y)$ denote the intensity at the same spatial location in the previous frame. If the noise is assumed to be additive and multiplicative, then

$$v_n(x,y) = N_m\, v_{n-1}(x,y) + N_a$$

where N_m is the multiplicative component and N_a is the additive component of the noise. Additive noise mainly changes the average intensity of the image and multiplicative noise changes the contrast or standard deviation of the image. Average intensity is a measure of the overall brightness of a gray scale image and standard deviation is the value that characterizes the extent to which any particular pixel value is likely to vary from the mean value. The larger the standard deviation, the greater the proportion of the pels which lie farther from the mean value. Hence for a gray level image, change in standard deviation implies change in the image contrast. Considering the above equation, if there is no noise N_m should be 1 and N_a should be 0 and average intensity and standard deviation of the image will be constant over time. But as shown in Figure 4.2(a) and Figure 4.2(b), the average intensity changes over time which establishes the presence of an additive component and the changes in standard deviation shows the presence of multiplicative noise.

4.2 Filtering Methods

The aim of this chapter is to find an effective and simple method to reduce the changes in the image characteristics caused by the above additive and multiplicative noise so that the images can be used for further processing. Different filtering methods have been tried to reduce the throbbing noise in the image sequences for the purpose of improving the visual appearance of the sequence and also to facilitate further image processing stages. They can be classified into spatial and temporal filtering methods. The following sections explain the algorithms of different filtering methods used.

4.2.1 Spatial Filter

The use of spatial masks for image processing is called spatial filtering and the masks themselves are called spatial filters. Lowpass filters are used for reducing noise and eliminating high frequency components such as edges and other sharp image details, while leaving low frequencies untouched. Figure 4.3(a) shows the cross section of a circularly symmetric lowpass spatial filter. The shape of the impulse response needed to implement a lowpass spatial filter can be achieved by using a spatial mask shown in Figure 4.3(b). The use of this form of mask is often referred to as neighborhood averaging. The operation of the filter is shown in Figure 4.4(a). Figure 4.4(b) shows successive frames from a test sequence and Figure 4.4(c) shows the result frames after applying lowpass spatial filter.

Figure 4.3 (a) Cross-section of a spatial domain filter. (b) Spatial filter.

This filter is not effective in reducing the noise as can be noted by observing the area of the half-circled light region. The lowpass spatial filter blurs the edges of the objects. This filter also proves that the noise is less correlated spatially which highlights the need for temporal filters that take temporal correlation into consideration for reducing the noise.

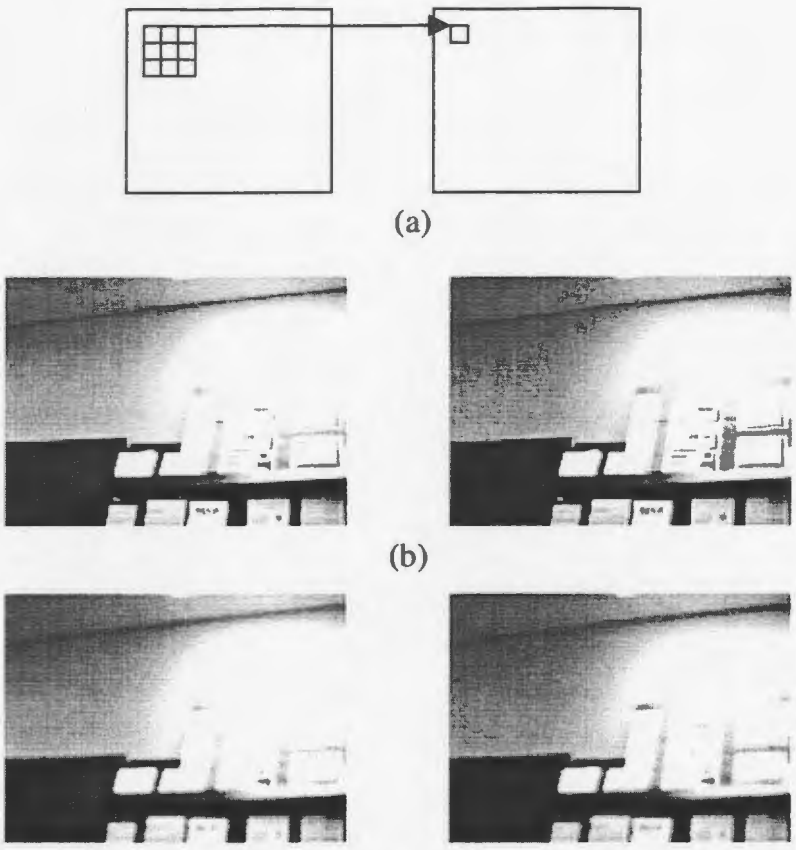


Figure 4.4 (a) Operation of the spatial lowpass filter. (b) Successive frames from a test sequence. (c) Filtered frames.

4.2.2 Temporal Filters

Temporal filtering has its roots in the statistical analysis of stochastic or random processes. The term stochastic process is used to describe the time evolution of a statistical phenomenon according to probabilistic laws. The time evolution of a phenomenon means that the stochastic process is a function of time defined on some observation model. The statistical nature of the phenomenon means that, before conducting the experiment, it is not possible to define exactly the way it evolves in time. The expectations or ensemble averages of a stochastic process are averages "across the process". The time averages can be defined as averages along the process. A random process is said to be ergodic if time averaging is equivalent to ensemble averaging. To be specific, consider a discrete time

stochastic process $v(t)$ that is wide sense stationary. A random process $v(t)$ is said to be wide sense stationary if,

$$E[\, v(t)^2 \,] < \infty \text{, for all } t.$$

Let constant μ denote the mean of the random process. For an estimate of μ, the time average $\mu'(N)$ may be used.

$$\mu'(N) = 1/N \sum_{t=0}^{t=N-1} v(t)$$

where N is the total number of samples used in the estimation. Note that the estimate $\mu'(N)$ is a random variable with a mean and variance of its own. If it can be said that the mean of $\mu'(N)$ is,

$$E[\mu'(N)] = \mu \text{ , for all } N$$

then the time average $\mu'(N)$ is an unbiased estimator of the ensemble average of the process. In the case of physical noise processes, heuristic knowledge of the noise process is often taken as a basis in assuming the property of the process. Therefore, noise processes are not strictly ergotic. The basic block diagram of a temporal filter is shown in Figure 4.5.

Figure 4.5 Block diagram of a linear filter.

The input-output relation for a stochastic process applied to a linear filter can be written as follows:

present values of output	+	linear combinations of past values of output	=	linear combinations of present and past values of input

The structure of the linear filter in Figure 4.4 is determined by the manner in which two linear combinations indicated in the above Equation are formulated. Hence three popular types of models are formulated.

1. Autoregressive model, in which only the present value of the process input is used.
2. Moving averages model, in which no past values of the process output are used.
3. Autoregressive moving averages model, in which the description of the above equation applies in its entire form.

Both Moving averages model (non-recursive) and Auto regression model (recursive) temporal filters have been tried in an effort to reduce the noise from the image sequences. The advantage of temporal filters over spatial filters is that no spatial degradation is introduced if there is little motion, with the additional advantage of having more information in the form of the previous frame store.

Temporal filters are used for television picture noise suppression and image enhancement which helps to bring signals from low quality sources to acceptable standards. An averaging filter in the temporal dimension over N frames reduces the noise variance by a factor of N. But it also blurs the moving parts of the scene. To reduce this effect Huang and Tsu [1981] proposed two methods to estimate motion in the temporal direction and temporal mean low pass filtering over the path of each pixel. In one method, to estimate the motion, they used variances of corresponding pels in the current and previous frames. In another method, they used the method of differentials to relate time differences to spatial differences. These filters are non recursive filters, which require high memory capacity.

Dennis [1980] proposed a recursive temporal filter, which does linear filtering in the non-moving regions and no or less filtering in the moving regions. To segment the image into moving and non-moving parts Dennis used a non linear function. To take care of the artifacts in areas where motion is not detected and in newly exposed areas, Dubois and Sabri [1984][1992], proposed filtering a scheme that takes motion parameters into account. This work can be said to be a combination of Huang's and Dennis's work. In this scheme, in addition to using a non linear function to separate moving and non-moving regions of the image, Dubois used motion displacement parameters to improve the functionality of the recursive filter. Kastseggelos et al.[1989], proposed a method to reduce the noise artifacts in the newly exposed areas and areas where motion is not estimated properly. In their method, the corresponding regions in the current frame replace the regions of high motion estimation errors, after a spatial filter has smoothed them. Kalivas and Sawchuk [1990] proposed a combined algorithm for segmenting moving and non-moving regions and motion estimation. To reduce the noise, spatio-temporal filtering is performed over the motion path of each pixel.

All the methods described above are used effectively for reducing high frequency noise in the image sequences. So they are not useful for reducing the low frequency throbbing noise from the low cost cameras. However, the basic filtering approach of segmenting the image into moving and non-moving regions and applying filtering in only the non-moving regions remains the same. The following sections describe various filtering methods starting from the basic temporal averaging filter to temporal filters using different spatial image characteristics in the local and global domains for moving and static regions segmentation. The effects of these filters on test sequences are also discussed.

4.3 Temporal Filtering Methods

This section describes the temporal filtering methods employed to reduce the low frequency, pulsing or throbbing noise. The non-recursive filters use averaging of corresponding pels in the current and the previous frames to achieve filtering of the noise. It should be noted that the filtering algorithms described below use standard filtering methods with some novel changes to reduce the low frequency noise.

4.3.1 Temporal Filter #1

This filter is the basic temporal averaging filter in which frames are averaged in the temporal dimension over 5 frames to reduce the throbbing noise variance by a factor of 5. The basic operation of the filter is defined as

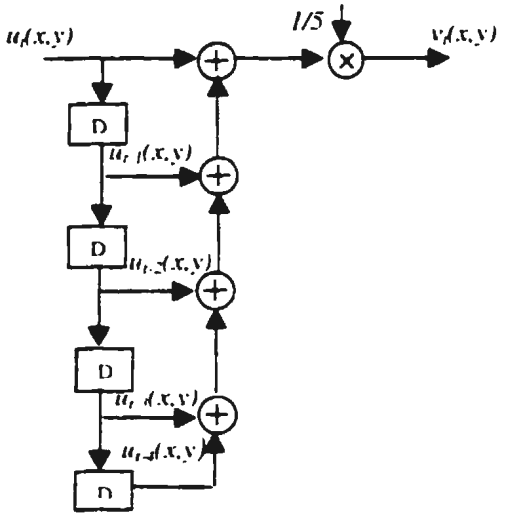


Figure 4.6 Block diagram of temporal filter #1.

$$v_t(x,y) = (u_{t-4}(x,y) + u_{t-3}(x,y) + u_{t-2}(x,y) + u_{t-1}(x,y) + u_t(x,y)) / 5 \qquad (1)$$

The block diagram of the filter is shown in Figure 4.6 where module *D* introduces delay of one frame .

4.3.2 Temporal Filter #2

Temporal filter #2 is an extension of temporal filter #1. Instead of averaging the corresponding pels in the current and the previous frames. pels which belong to moving and static regions are segmented by using the standard deviation σ_t and average A_t of the corresponding pels. The basic idea of this filter is. if there is motion in the current frame pel. then the pel value will be outside the range between $A_t + 2\sigma_t$ and $A_t - 2\sigma_t$ and the pel value is retained in the output frame. which preserves motion. Otherwise. the pel value is replaced by the average value A_t assuming the pel belongs to a static region. thus reducing the noise variance in the static region. The operation of the filter is explained as.

1. The average A_t of corresponding pels in current and four previous frames is calculated as

$$A_t = (u_t(x,y) + u_{t-1}(x,y) + u_{t-2}(x,y) + u_{t-3}(x,y) + u_{t-4}(x,y)) / 5$$

2. The standard deviation σ_t is calculated as

$$\sigma_t = \sqrt{\frac{1}{5}\sum_{i=t}^{t-4}(u_i(x,y) - A_t)^2}$$

3. The value of the pixel in the result frame $v_t(x,y)$ is defined as

$$v_t(x,y) = \{ A_t \text{ if } u_t(x,y) < A_t + 2\sigma_t \text{ and } u_t(x,y) > A_t - 2\sigma_t \}$$

$$= \{ u_t(x,y), \text{otherwise} \}.$$

4.3.3 Temporal Filter #3

This filter is same as temporal filter #1 except for the weights given to the pels in the current and the previous frames. In ordinary temporal averaging all the corresponding pels

are included in the average calculation. In this filter. pels are included in the average calculation depending on whether there is motion between the frames. When motion occurs. there will be a change of value of intensity of the pixel between the previous and the current frames. This property is used to detect motion between frames. Assuming that the value of intensity change due to noise is less than the value of intensity change due to motion. a threshold is used to segment the pels. The average calculation includes pels of frames in the static region where there is no intensity change, and pels of frame in the noise region where there is less intensity change than in the moving region. The algorithm for calculating the average is now modified as follows.

$$A'_t = (\kappa_0 u_t(x,y) + \kappa_1 u_{t-1}(x,y) + \kappa_2 u_{t-2}(x,y) + \kappa_3 u_{t-3}(x,y) + \kappa_4 u_{t-4}(x,y)) / 5 \qquad (2)$$

where the weights κ_t are calculated as

$$\kappa_t = \{ 0 \quad \text{if } \kappa_{t-1} = 0 \}$$

$$= \{ 0 \quad \text{if } | u_{t-i}(x,y) - u_{t-i+1}(x,y) | > T \}$$

$$= \{ 1, \text{ otherwise} \}$$

In plain words this can be explained as, if there is a change in intensities greater than threshold T from the previous frame pel $u_{t-1}(x,y)$ to the current frame pel $u_t(x,y)$, then the current frame pel is discarded from the average calculation. It is also discarded if the change in intensity is greater than threshold T from pel $u_{t-1}(x,y)$ to pel $u_{t-2}(x,y)$. A threshold can be chosen in such a way that it can segment motion and noise variations. Adjusting the threshold manually for better subjective output does this. Basically this filter reduces the noise variance in the static regions by a factor which depends on the motion information between the frames.

4.4 Spatio-temporal (Local Space Domain) Filters

All the temporal filters discussed above exploit the information in the temporal dimension. Considering both temporal and spatial dimensions gives an advantage of more information at hand to improve the function of temporal filters. In the temporal filters discussed below, spatial information between the current and the previous frames is used to segment the moving and non-moving regions and low pass mean temporal filter is applied over non-moving regions. Temporal filter #3 is chosen as the low pass mean temporal filter as it removes moving pels in the averaging function.

4.4.1 Spatio-temporal Filter #1

This filter uses the spatial information in a local region and extends it in the temporal dimension. A spatial window of size 4×4 is used in the filter. The average intensity of the local region is used as the spatial information to segment the moving and non-moving regions. The difference of average intensities of corresponding blocks of current and previous frames is checked against a threshold to detect the regions of motion. If the difference is greater than a threshold, then motion is detected in the region and the block of the current pels are retained. Otherwise each pel in the current frame block is replaced by a temporal mean value.

The operation of the filter is described as,

1. Divide the frames into non-overlapping 4×4 blocks of pixels.
2. For each corresponding pair of blocks in the previous and current frames, average intensities of the blocks are calculated in the spatial domain as B_{t-1} and B_t respectively.

3. If ($| B_t - B_{t-1} | > T$)

   Replace the block in the output frame v_t by the corresponding current frame u_t block.

   Else

   Replace pel value p of each pel $v_t(x,y)$ in the corresponding block of the output frame v_t , by temporal average A'_t calculated using equation (2).

4.4.2 Spatio-temporal Filter #2

In this filter, the ratio of intensities of pels in the current frame $u_t(x,y)$ and the previous frame $u_{t-1}(x,y)$ is used to detect motion between corresponding blocks of the previous and the current frames. The value α is defined as:

$$\alpha = u_t(x,y)/u_{t-1}(x,y)$$

Variations of the α values for pels in a 2×2 block is shown in Figure 4.7. There are no moving objects in this 2×2 block. The variations are purely due to the throbbing noise introduced by the low-cost camera. If there is motion in the block then changes in pel values from $u_{t-1}(x,y)$ to $u_t(x,y)$ will be dissimilar. Since this noise is assumed to be spatially homogeneous the changes in the pel values will be uniform. This will make all the α values for the respective pels in the block change uniformly in one direction in the presence of noise. As illustrated in Figure 4.7 the presence of noise makes the α values change in a uniform manner. In this filter this property is used to segment motion changes and noise changes and mean temporal filtering is applied to blocks where motion is not detected. α values are used to segment the changes due to noise and motion.

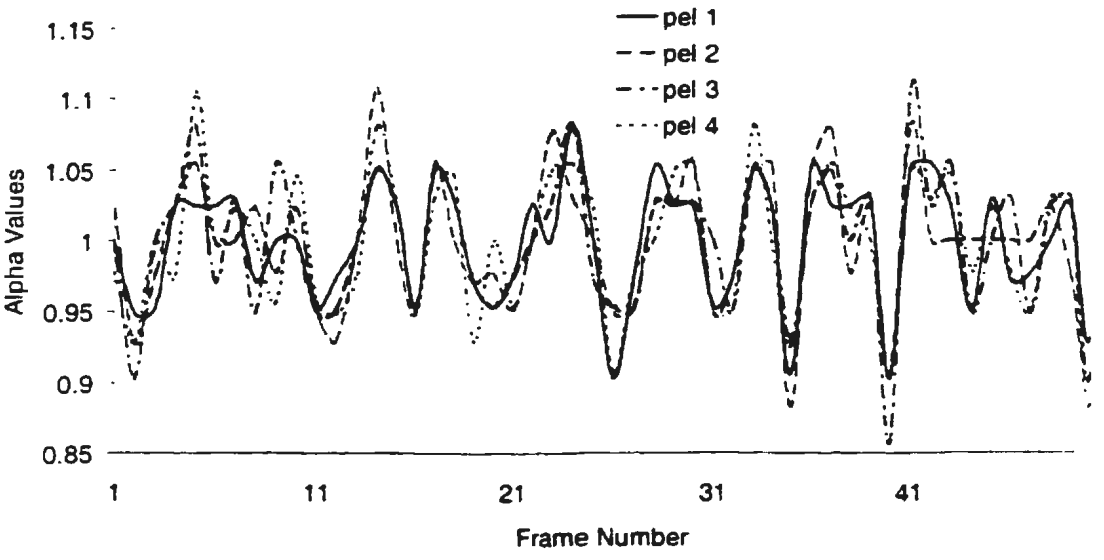


Figure 4.7 Variation of α values for each pel of a sample 2×2 block after applying spatio-temporal filter #2.

A reasonable block size of 4×4 is chosen for effective detection. N_G is number of uniform changes in the increasing direction from the previous frame to the current frame and N_L is uniform changes in the decreasing direction. N_α is the number of uniform changes in the pel values from the previous frame to the current frame in either increasing or decreasing directions and hence represents uniform changes. The value of N_α is checked against a threshold. If N_α is greater than the threshold, then the changes are due to noise and hence that particular block is temporally averaged to remove noise. Checking the value of N_α against a threshold improves the accuracy of this algorithm in detecting changes due to noise and motion. The operation of the filter is explained below:

1. Divide the frame into non-overlapping 4×4 blocks of pels.
2. For each pel in the block, the value α is calculated. α can be either greater than 1 or equal to 1 or less than 1.

The number of α values greater than 1 is counted as N_G

The number of α values less than 1 are counted as N_L

The number of α values equal to 1 is counted as N_E

The value N_α is calculated as follows:

$$N_\alpha = \{ N_G + N_E, \text{ if } N_G >= N_L \}$$

$$= \{ N_L + N_E, \text{ if } N_L > N_G \}$$

3. If ($N_\alpha < T$)

   Replace the block in the output frame v_t by the corresponding current frame u_t block.

   Else

   Replace pel value of each pel $v_t(x,y)$ in the corresponding block of the output frame v_t by temporal average A'_t calculated by using Equation (2).

4.5 Spatio-temporal (Global) Filters

These filters use global spatial properties in segmenting the moving and non-moving regions and effectively filter the noise. These filters can be considered as an extension of spatio temporal filters, which use local spatial domain properties. Since greater selectivity can be obtained by using recursive filters, recursive filters are chosen for this application. This is especially important in temporal filtering where each increase by one in the filter order requires an additional frame memory.

4.5.1 Spatio-temporal filter #3

This filter uses the average brightness of the frame as a spatial property. It assumes that the noise affects the image frame homogeneously by an error E and corrects the frame for that error. In other words, this filter is an additive noise removing recursive filter. The error E is the difference between the current and the previous frames as shown in the block diagram of the filter (Figure 4.8). The algorithm of the filter is described below:

1. Global average intensities of the previous and current frames are calculated as A_{t-1} and A_t respectively and the error in intensity E is obtained as $A_t - A_{t-1}$.
2. If |E| is less than a threshold T, then the intensity of each pel in the current frame is increased by an amount E.

The objective of threshold T is to take into account the change in average intensity of successive frames due to motion of objects in the frame sequence. Variations in the

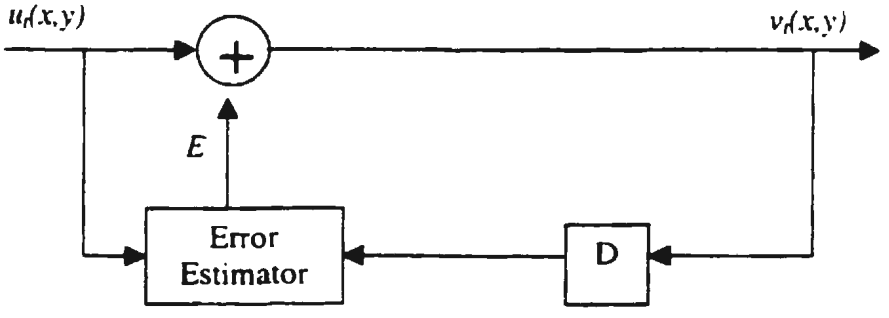


Figure 4.8 Block diagram of spatio-temporal filter #3

average intensity of pixels of block size 4×4, before and after filtering for a test sequence is shown in Figure 4.9. It should be noted that there is no movement of object in the block chosen.

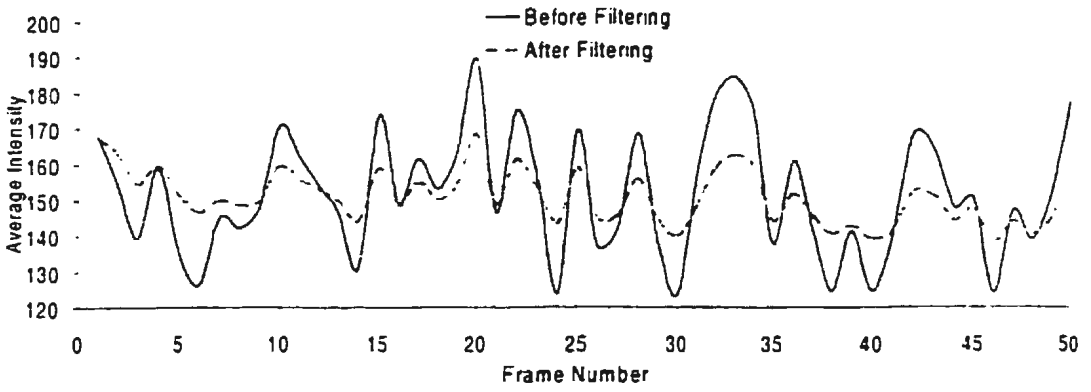


Figure 4.9 Variation in the intensity before and after applying spatio-temporal filter #3.

4.5.2 Spatio-temporal Filter #4

This filter is as an extension to spatio-temporal filter #3. In this filter, the quotient of averages of the current and the previous frames is used as a parameter to define the global property of the image. However not all the pels in a frame are used for calculating the average intensity of a frame. Only pels in the non-moving region are used in the average calculation. From the quotient value an intermediate frame is predicted. If the predicted error is high, then the output frame value is replaced by the predicted value assuming that there is motion at this position. Otherwise the output value is replaced by a temporally averaged value. The block diagram of the filter is shown in Figure 4.10, where $v_t(x,y)$ is the output frame and $u_t(x,y)$ is the input frame, $v'_t(x,y)$ is the processed intermediate frame, T_1, T_2 are threshold values and module D introduces one frame delay.

The basic operation of the filter is described as:

$$v_t(x,y) = \{ v'_t(x,y) \text{ , if } |v'_t(x,y) - v_{t-1}(x,y)| > T_2 \}$$

$$= \{ (v'_t(x,y) + v_{t-1}(x,y)) / 2 \text{ , otherwise} \}$$

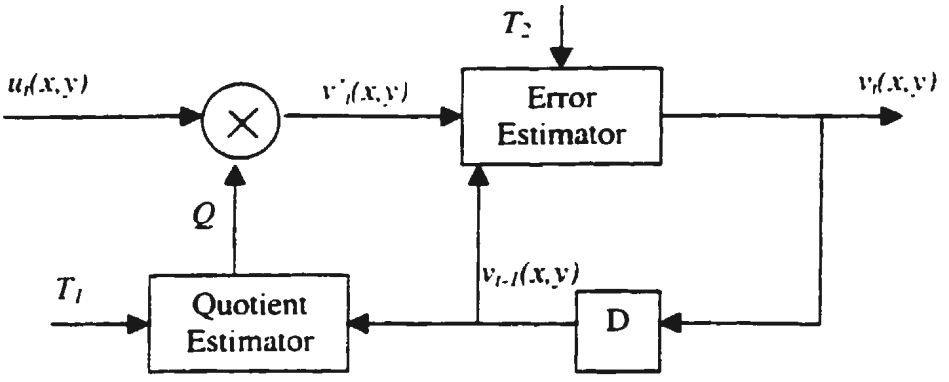


Figure 4.10 Block diagram of spatio-temporal filter #4.

The function of the "Quotient estimator " module is described as follows:

1. The points in the current frame $u_t(x,y)$ and the points in the previous frame $v_{t-1}(x,y)$ are identified as sets $S_t(p)$ and $S_{t-1}(p)$ respectively where

$S_t(p) = \{ u_t(x,y)$, such that $|u_t(x,y) - v_{t-1}(x,y)| < T_1 \}$

$S_{t-1}(p) = \{ v_{t-1}(x,y)$, such that $|u_t(x,y) - v_{t-1}(x,y)| < T_1 \}$,

where T_1 is the threshold.

2. The averages of all the points in the sets $S_t(p)$ and $S_{t-1}(p)$ are calculated as $A_t(p)$ and $A_{t-1}(p)$ respectively. Q value is calculated as:

$$Q = A_t(p)/A_{t-1}(p)$$

3. $v'_t(x,y)$ is calculated using equation $v'_t(x,y) = Q\, u_t(x,y)$.

4. Error E is calculated in the " Error Estimator" module as $|v'_t(x,y) - v_{t-1}(x,y)|$.

5. If ($E < T_2$)

   $v_t(x,y) = (v'_t(x,y) + v_{t-1}(x,y))/2$

   Else

   $v_t(x,y) = v'_t(x,y)$

Threshold T_1 is used to segment the pels in the current frame that are changed due to motion and noise assuming that intensity change due to noise will be less than T_1. Those pels that are changed due to motion are not taken into consideration when calculating averages. Threshold T_2 is used to check the prediction errors.

4.5.3 Spatio-temporal filter #5

In this algorithm, the spatial homogeneity of the noise is exploited to estimate the global parameters for correcting the pel values in the current frame, then apply these with a local temporal filter. This filter uses the histograms of successive image frames.

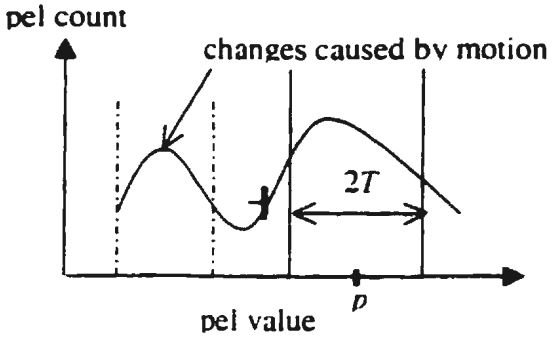


Figure 4.11 Histogram of the previous frame $v_{t-1}(x,y)$ for pel value p in the current frame $u_t(x,y)$ used in spatio-temporal filter #5.

As shown in Figure 4.11, for each pel value p in the current frame, the corresponding values in the previous frame and the number of occurrences of these values are noted. Assuming that the noise causes a variation in the range of $p+T$ and $p-T$ in the pixel value, only pixels within this range are averaged. Variation of the values of the pel outside this range is assumed to be caused by motion in the image sequences. If there is no average value for any p, the value is linearly interpolated from the nearest averages. Thus a

monotonic transformation function is obtained. This function is inverted to get the necessary filtering operation. A block diagram of the filter is shown in Figure 4.12. The basic operation of the first order recursive filter can be described as follows:

$$v_t(x,y) = \beta\ u_t(x,y) + (1-\beta)\, A_t(\ u_t(x,y)\ ) \qquad (3)$$

where $v_t(x,y)$ is the output of the filter in frame t at position (x,y), $u_t(x,y)$ is the input at that position, β is a constant and $A_t()$ is the output from the module "*Global Estimator*". Module D introduces a delay of one frame. The global estimator uses $u_t(x,y)$, $v_{t-1}(x,y)$ and a threshold T for estimating parameters. The operation of the *global estimator* module for linear mapping of picture values is explained below.

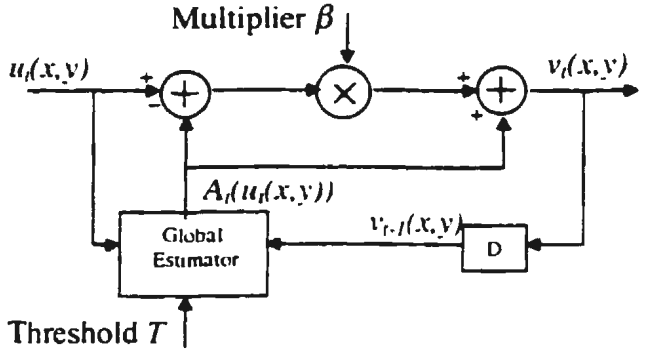


Figure 4.12 Block diagram of spatio-temporal filter #5.

1. For each possible pel value p (e.g. 0 to 255), the pels in the current input frame u_t with that value are located and the corresponding pels in the previous output frame v_{t-1} are identified as set $S_{t-1}(p)$.

$$S_{t-1}(p) = \{\ v_{t-1}(x,y),\ \text{such that}\quad u_t(x,y) = p\ \}$$

2. Points in $S_{t-1}(p)$ whose values are more than a distance T from p are removed to yield set $S'_{t-1}(p)$

$$S'_{t-1}(p) = \{\ v_{t-1}(x,y),\ \text{such that } ut(x,y) = p \ \text{ and }\ |\ v_{t-1}(x,y) - u_t(x,y)|\ <\ T\ \}$$

3. The average of $S'_{t-1}(p)$ is calculated for all p, yielding $A_t(p)$.

4. Zero average values are linearly interpolated. (End zero average values are extrapolated). The data obtained after interpolation are fitted to a straight-line model $y = ax + b$. (End zero values are not taken into consideration when calculating the coefficients a and b). The average values $A_t(p)$ are computed for all the 256 pixel values.

5. $v_t(x,y)$ is calculated from Equation (3).

Figure 4.13 shows the value $A_t(p)$ for a sample frame.

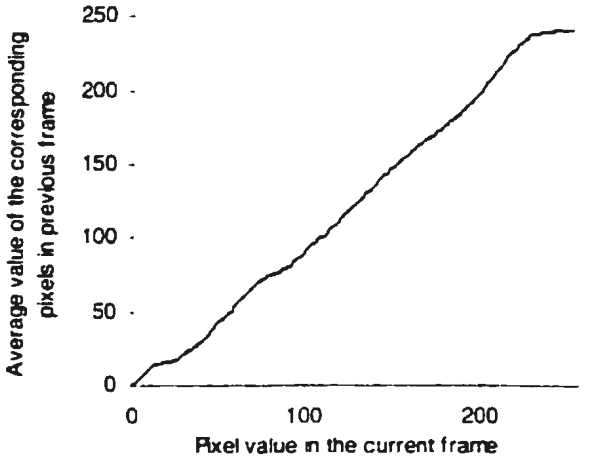


Figure 4.13 Sample computed averages from spatio-temporal filter #5.

The multiplier β is used to control the amount of filtering that can be permitted. The value of β ranges from 0 to 1. It should be noted that the value of β represents the amount of signal of the current frame $u_t(x,y)$ and hence the noise too. Increase in the value of β increases the amount of noise permitted in the output frame and a decreased value reduces the noise in the output frame. However, experiments suggested a value of $\beta = 0.1$ is appropriate for most of the lighting conditions. Threshold T is used to select points in the current frame that changed only by noise from the previous frame. The value of T ranges from 0 to 255. A value of 20 for threshold T is found to improve the

subjective quality of the output frames. Increasing the value of the threshold greater than 50, increases the chance of selecting changed points due to motion and hence introduces dirty windows artifacts. When the values of multiplier β and threshold T are set at 0.1 and 20 respectively during experiments, it is observed that the filter is quite robust to small incremental changes in the values of β and T. Figure 4.14 shows the variation of the average intensity of a sample 4×4 block of pixels of a frame, where there is no motion before and after filtering.

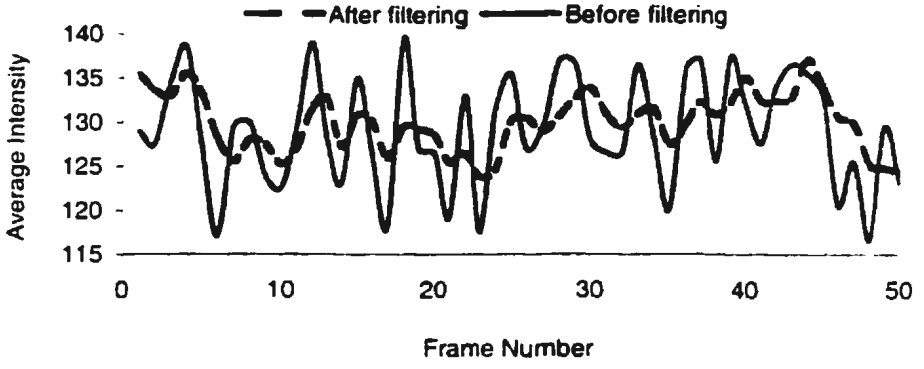


Figure 4.14 Variation in the average intensity of a sample 4×4 block of pels before and after applying spatio-temporal filter #5.

4.6 Dennis Filter

For comparison purposes the recursive temporal filter proposed by Dennis [1980] is implemented. Dennis filter uses a non-linear function for segmenting the moving and non-moving regions of an image and applies this function to the entire image. The block diagram of the filter is shown in Figure 4.15.

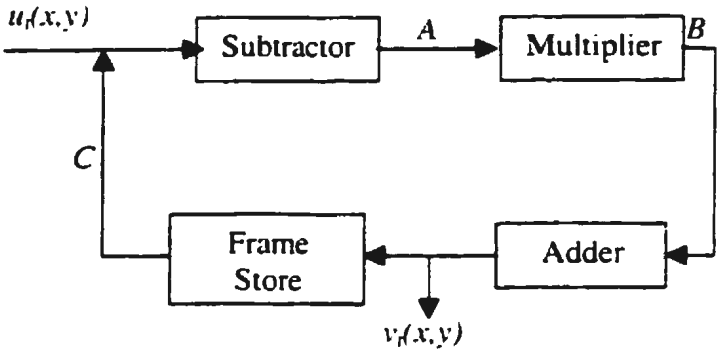


Figure 4.15 Block diagram of Dennis filter.

The non-linear function used in the filter is,

$$Y = X\left\{1 - a\exp-\left(\frac{|X|-p}{d}\right)^2\right\}$$

where a controls the amplitude of the modification, d is the decay parameter. p is an off-set which moves the non-linear region up or down the vertical axis. Y is the multiplier output and X is the input to the multiplier. The basic technique is to minimize the degradation of movement by the filter by an adaptation such that the multiplier is dependent on the amount of motion at each point in the picture. In implementing this filter, the values of a and p are kept at 1 and 0 respectively which is the same as in the experiments conducted by Dennis.

4.7 Motion Estimation

Many researchers have described the advantage of using motion compensation for noise reduction. The block matching method [Jain and Jain, 1981] is implemented to estimate the motion vectors in the presence of noise. The procedure of operation is as follows:

1. Matching element intensity values in a given block of size $M \times N$ in the present frame $u_t(x,y)$ with those of a similar block of size $M \times N$ in the previous frame $u_{t-1}(x,y)$ which is part of a reference block of dimension ($M+2p$)×($N+2p$) , where p is the allowed maximum displacement.
2. The matching criterion is mean square error $D(i,j)$ where $D(i,j)$ is defined as

$$D(i,j) = \frac{1}{M \times N} \sum_{x=1}^{M} \sum_{y=1}^{N} (u_t(x,y) - u_{t-1}(x,y))^2, -p \le i, j \le p$$

Full search motion estimation is performed. After getting the motion vectors, the output frame is formed using these vectors and the previous frame $u_{t-1}(x,y)$ as reference. Block matched frames for two test sequences are shown in Figure 4.18(k) and 4.19(k). As can be seen from the results shown in Figure 4.18(k), even approximate estimation of motion using this method is difficult in the presence of noise. However, there are better motion estimation techniques available that are not tried in this research. J.D.Robbins and A.N.Netravali[1983] describe a pel recursive motion estimation technique. Shaker Sabri[1983] describes a technique that combines both block matching and pel recursive techniques within the block. There are methods available in which the two dimensional Fourier transform of each frame is calculated and phase information from the transforms is used to determine the relative displacement between successive fields [Graham Tho-

mas, 1991]. However the computational complexity of these methods is high. Since the present research is aimed at real time image processing, filtering methods based on motion compensation become unsuitable for reducing the throbbing noise.

4.8 Experiments

Experiments were conducted to study and compare the performances of the filters that are implemented, subjectively and quantitatively. Since the problem is specific to the type of camera being used, it is not possible to use any standard picture sequences in the experiments. Sequences are captured from a low cost camera and used for experimental purposes. The computer system used, has a Pentium-S processor, 32 MB RAM and running at a clock speed of 166MHz. The camera used is a QuickCam gray level camera from Connectix Corporation set to capture images at 64 shades of gray level. Appendix A provides complete camera specifications and controls. Two test sequences are taken for test results. Test sequence #1 is used for subjective tests and test sequence #2 is used for quantitative tests. The following sections describe the test sequences and discuss results.

4.8.1 Test sequences

Test sequence #1, in which an object is moving at a moderate speed, is captured using the camera and is used to compare the subjective quality of the results of different types of filters implemented. It should be noted that each frame of test sequence #1 is corrupted by noise. In the results, only four successive frames of the test sequence #1 are shown. The size of the captured image is set as160x120 pels/frame. Four successive frames from

the captured sequences showing both throbbing noise and minimal motion are taken for testing purposes. Throbbing noise can be noticed by comparing the size of the light oval region in frame1 and frame2 of test sequence #1. It can be noted that the size of the oval region changed drastically from frame1 to frame2, indicating the presence of the pulsating noise. These frames are shown in Figure 4.18(a). The global statistics of the frames are shown in Table 4.1 and the histograms are shown in Figure 4.16.

Test sequence #2, in which an object is moving at a moderate speed, is captured and is used to compare quantitatively the results of different types of filters implemented. In test sequence #2 noise in each frame is removed. This is done by capturing 10 frames of the scene of a fixed object. All the frames are temporally averaged to form a result frame in which the noise variance is reduced by a factor of 10. Then the scene is changed by moving the object to represent motion.

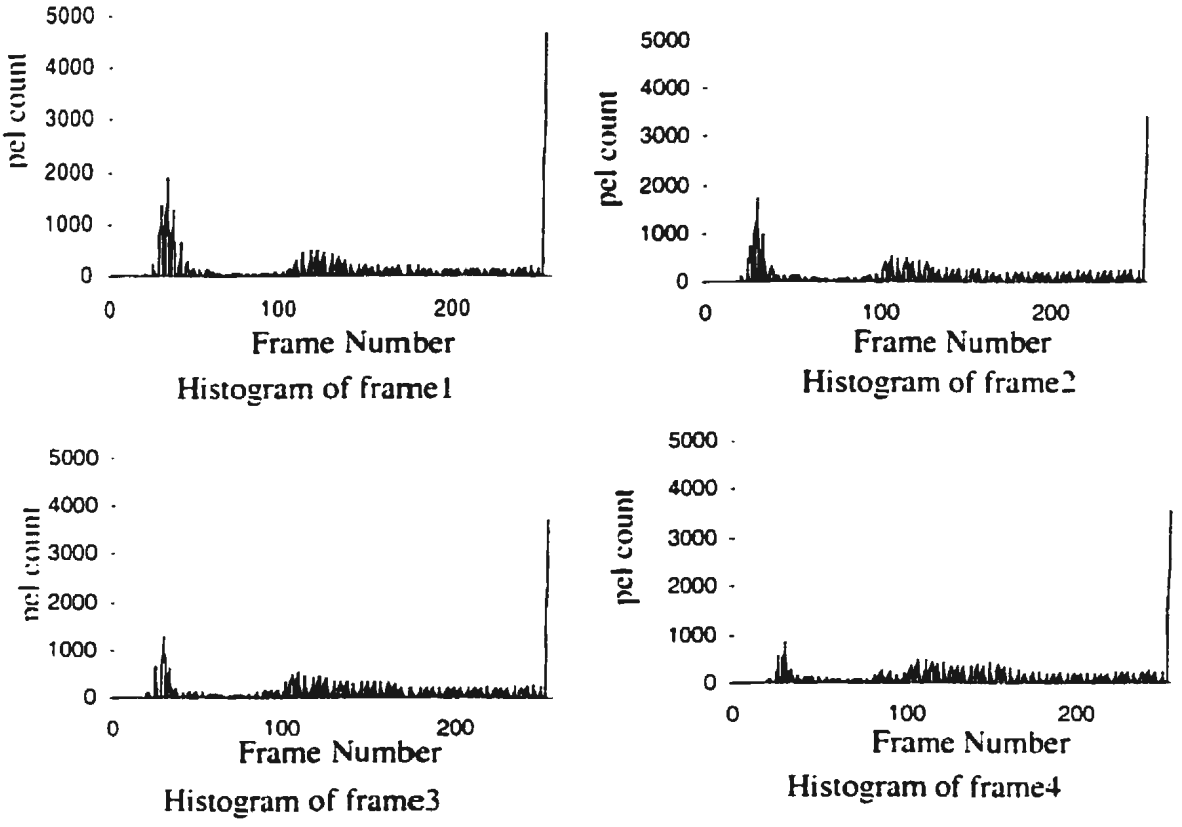


Figure 4.16 Histograms of 4 successive frames in test sequence #1.

Frame Number	Average intensity	Minimum count of any pel value	Maximum count of any pel value
Frame1	141	0	4656(24%)
Frame2	146	0	3428(17%)
Frame3	154	0	3709(19%)
Frame4	157	0	3551(18%)

Table 4.1. Global statistics of 4 successive frames in test sequence #1.

Temporal averaging is done again for the frames captured with this scene to form a result frame. Hence, each frame in test sequence #2 is a temporally averaged frame with reduced noise variance.

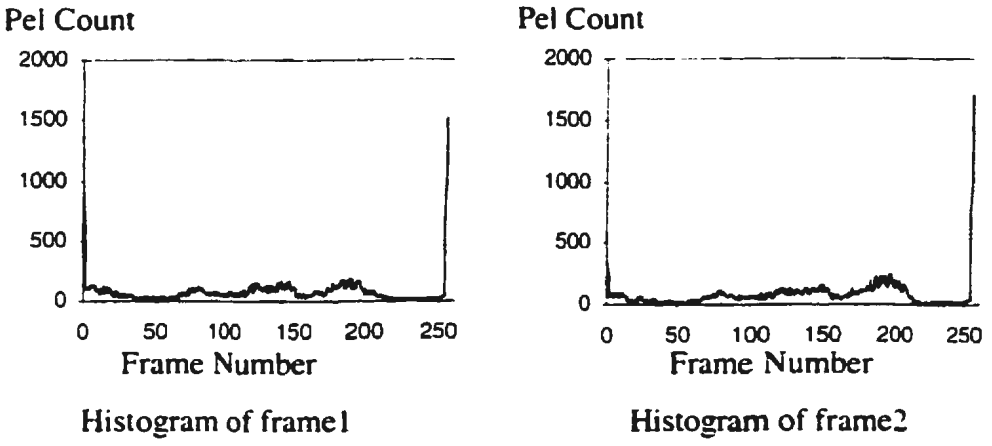


Figure 4.17 Histograms of 2 successive frames in test sequence #2.

Frame Number	Average intensity	Minimum count of any pel value	Maximum count of any pel value
Frame1	125	2	1525(7%)
Frame2	145	5	1707(8%)

Table 4.2 Global statistics of 2 successive frames in test sequence #2.

Two successive frames of the test sequence are used in the quantitative tests. The width and height of the image captured are set to 160x120 pels/frame respectively. Because of temporal averaging the throbbing noise is very minimal and it is barely noticeable in the test frames. These frames are shown in Figure 4.19(a). The global statistics of the frames are shown in Table 4.2 and the histograms are shown in Figure 4.17.

4.8.2 Results and Discussion

Efforts have been made to improve the photographic illustrations of the results to the quality of the image as seen on the monitor. However still one has to keep in mind that

degradation or improvement of a more subtle nature is easily visible on a quality monitor and may not appear on a printed page. Though the nature of the noise is different from earlier noises treated by many authors. one of the filters used by a previous researcher [Dennis. 1980] is implemented for comparison purposes along with some of the filters discussed earlier. for filtering temporal noise. Table 4.3 gives a summary of the properties of the filters implemented. The performance of the filters is discussed and comparison statements are made where relevant.

To evaluate the subjective quality of the filtering results for test sequence #1. the result frames and the original frames are shown in Figure 4.18. If the size of the oval shaped light region is maintained between successive frames. then the filter can be said to be effective in reducing throbbing noise. Quantitative results of the filters are provided in the table 4.4. The parameters measured are mean square error (MSE) and peak signal to noise ratio (PSNR). These two parameters are measured by using frame2 of the test sequence #2 as the input frame and filtered frame2 as the output frame. Test frames and result frames are shown in Figure 4.19.

Temporal filter #1 reduces throbbing noise effectively in the stationary areas if the frame storage capacity of the filter is increased. But it degrades or blurs the moving objects. This can be seen from the results provided in Figure 4.18(b) and 4.19(b). This prompts the need to find a way to segment the regions into moving and non-moving and apply the filter only in the non-moving regions.

Filter Name	Spatial Property	Temporal Property	Filter Type	Filtering Method
Temporal Filter #1	- Nil -	Pels in the current and 4 previous frames	Non recursive	Temporal averaging
Temporal Filter #2	- Nil -	Pels in the current and 4 previous frames	Non recursive	Standard deviation of the corresponding pels in all the frames is used for motion detection. Temporal averages in the static region.
Temporal Filter #3	- Nil -	Pels in the current and 4 previous frames	Non recursive	Temporal averaging filter. Pels are included in the average calculation depending on motion between the frames.
Spatio Temporal Filter #2	Change in the average intensity of block of pels between the current and the previous frames is used as a property for motion detection	Pels in the current and 4 previous frames	Non recursive	Standard deviation of the corresponding pels in all the frames is used for motion detection. Temporal averages in the static region.
Spatio Temporal Filter #2	Number of intensity of pels changing in the same direction is used as a property for motion detection. Pels in corresponding blocks of the current and the previous frames are used.	Pels in the current and 4 previous frames	Non recursive	Standard deviation of the corresponding pels in all the frames is used for motion detection. Temporal averages in the static region.

Table 4.3 Summary of filtering methods used. (Contd...)

Filter Name	Spatial Property	Temporal Property	Filter Type	Filtering Method
Spatio Temporal Filter #3	Change in the average intensity of the current and previous frames is used for motion detection	Pels in the current and the previous frames	Recursive	Adds the prediction error, calculated from using the spatial property, to all the pels in the current frame if motion is not detected.
Temporal Filter #2	Quotient of the average intensities of the current and the previous frames is used for motion detection.	Pels in the current and the previous frames	Recursive	Filtering is applied where motion is not detected.
Temporal Filter #3	Non-linear transfer function is used to segment moving and non-moving regions.	Pels in the current and the previous frames	Recursive	Filtering is performed by applying the transfer function to all the pels in the current frame.
Spatio Temporal Filter #2	Histograms of successive frames is used to segment moving and non-moving regions and to obtain a transfer function.	Pels in the current and the previous frames	Recursive	Filtering is performed by applying the transfer function to all the pels in the current frame.

Table 4.3 Summary of filtering methods used.

Figure 4.18(a) Successive frames from the test sequence #1

Figure 4.18(b) Frames after using temporal filter #1

Figure 4.18(c) Frames after using temporal filter #2

Figure 4.18(d) Frames after using temporal filter #3

Figure 4.18(e) Frames after using spatio-temporal filter #1

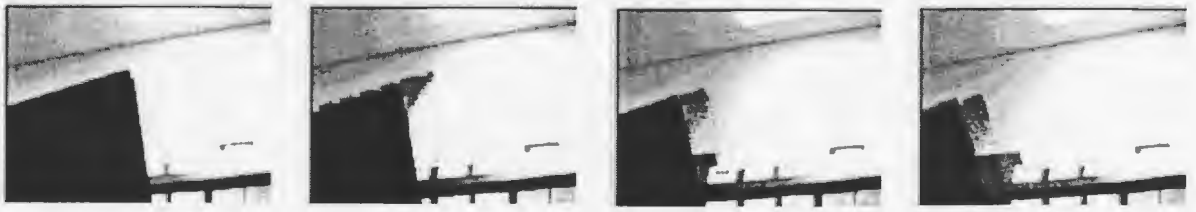


Figure 4.18(f) Frames after using spatio-temporal filter #2

Figure 4.18(g) Frames after using spatio-temporal filter #3

Figure 4.18(h) Frames after using spatio-temporal filter #4

Figure 4.18(i) Frames after using Dennis Filter

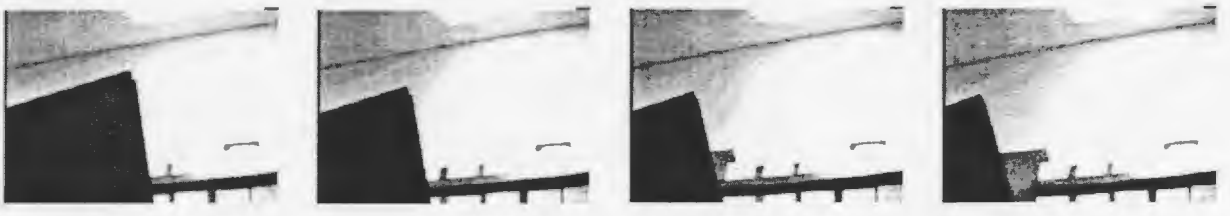


Figure 4.18(j) Frames after using spatio-temporal filter #5

Figure 4.18(k) Frames after using block matching

Figure 4.18 Result frames after applying temporal and spatio-temporal filters on test sequence #1 for subjective quality analysis.

Figure 4.19(a) Successive frames from test sequence #2

Figure 4.19(b) Frame after using temporal filter #1

Figure 4.19(c) Frame after using temporal filter #2

Figure 4.19(d) Frame after using temporal filter #3

Figure 4.19(e) Frame after using spatio temporal filter #1

Figure 4.19(f) Frame after using spatio temporal filter #2

Figure 4.19(g) Frame after using spatio temporal filter #3

Figure 4.19(h) Frame after using spatio temporal filter #4

Figure 4.19(i) Frame after using Dennis filter

Figure 4.19(j) Frame after using spatio temporal filter #5

Figure 4.19(k) Frame after using block matching

Figure 4.19 Result frames after applying temporal and spatio-temporal filters on test sequence #2 for quantitative analysis.

Temporal filter #2 is based on the standard deviation of values of pixels in the temporal dimension. The standard deviation will be large if there is lot of variation in the pixel values of successive frames that may be due to noise or due to moving objects. The standard deviation will be less if there is less variation that results in no averaging of corresponding pixels. This filter is an averaging filter that improves performance by adapting to the conditions in the temporal dimension. This filter however cannot separate or distinguish the noise or motion variations. In the experiments conducted, it is seen that averaging takes place on both noise and motion changes, which results in better noise reduction performance and causes dirty window artifacts in moving areas, as can be seen in Figures 4.18(c) and 4.19(c).

Temporal filter #3 modifies the average calculation depending on intensity variations between each pel of the current and previous frames using a threshold. From the results shown in Figures 4.18(d) and 4.19(d), and other experiments, it has been observed that this filter performs better than the filter using ordinary average calculation. The drawback with this filter however is, if a large threshold value is chosen, assuming that changes below the threshold represent noise, it results in dirty window artifacts in the moving regions and better averaging of the noise signal. But a small threshold allows noise in the output frames. However, this filter performance is better than the ordinary averaging filter.

Spatio-temporal filter #1 uses information of the local spatial region and in the temporal dimension. The average intensity of a block is used for providing spatial information. But it is found that a change in the average intensity of block of pels is not effective as a

measure for segmenting motion and noise variations. It has the same drawback of either allowing the noise as seen from the results presented in Figures 4.18(e) and 4.19(e) or blurring of the moving objects.

Spatio-temporal filter #2 uses the ratio of intensities of individual pels in the blocks of the current and previous frames to provide the spatial information. However, from the results shown in Figures 4.18(f) and 4.19(f), it is seen that this filter causes dirty windows artifacts in regions where there is motion. The noise may contain motion-like property of causing dissimilar α changes in a block which causes this filter to segment motion changes and noise changes improperly.

Spatio-temporal filter #3 uses the global spatial property of average intensity of the frame. It is found from the results shown in Figures 4.18(g) and 4.19(g) that this property is not effective in segmenting moving and non-moving regions. This filter is not at all effective in reducing the noise and acts as a simple additive noise removing filter with no motion segmentation property. From Figure 4.19, it is seen that the filter reduces only the amplitude of the image signal keeping the frequency of variations the same as the noise without reducing the throbbing noise.

Spatio-temporal filter #4 uses a global spatial property of the image in the temporal dimension. As can be seen from the results shown in Figures 4.18(h) and 4.19(h), this filter is also not effective in reducing the throbbing noise. This may be due to the fact that the quotient factor Q is not effectively representing the global property of the noise. This

filter also produces similar artifacts, which is clearly shown in the results where artifacts are seen in the moving region. A property that can represent the noise globally and yet at the same time identifies the noise and motion variations is needed.

The Dennis filter uses a non-linear transfer function to segment moving and non-moving regions. It is found that Dennis filter is effective in reducing the throbbing noise. However that the transfer function is not effective in distinguishing motion from noise. Hence the filter introduces dirty window artifacts, though these are very small compared to other filters, particularly in the edges of the moving object as shown in Figures 4.18(i) and 4.19(i).

Spatio-temporal filter #5, the proposed filter, uses successive histograms of frames for filtering operation. Since histograms represent global characteristics of an image, pel intensity changes due to motion and noise can be segmented more effectively than the previous filtering methods. From the results (Figure 4.18(j) and 4.19(j)), it can be seen that in stationary areas the performance of this filter is superior to the other temporal filters discussed earlier for controlling the throbbing noise. The improvement in performance is also observed in moving areas, where the previous temporal filters introduce "dirty window" artifacts. From Figure 4.14 it is seen that this filter reduces the frequency of intensity variations due to noise, making it effective in attenuating the throbbing noise.
The effectiveness of this filter can also be seen from the quantitative test results provided in Table 4.4. From the PSNR figures provided in column 4 of the table, it can be observed that spatio temporal filters perform better than temporal filters, with the best figure obtained from using spatio-temporal filter #5.

Filter Name	Subjective test results	Quantitative test results	
		MSE	PSNR
Temporal Filter #1	Produces dirty windows artifacts in the moving areas	1684.4694	15.86617
Temporal Filter #2	Produces dirty windows artifacts in the moving areas	1683.2844	15.86923
Temporal Filter #3	Produces dirty windows artifacts in the moving areas, but better than Temporal filter #1.	171.2103	25.79550
Spatio Temporal filter #1	Performance is similar to Temporal filter #3.	196.0810	25.20645
Spatio Temporal Filter #2	Produces dirty windows artifacts. Performance is better than Spatio-temporal filter #1.	93.7020	28.41331
Spatio Temporal Filter #3	No dirty windows artifacts and reduction of noise. Worse performance than the above spatio-temporal filters.	373.1845	22.41157
Spatio Temporal Filter #4	Produces fewer dirty windows artifacts. But performance is better than the above spatio-temporal filters.	44.2075	31.67588
Dennis filter	Produces fewer dirty windows artifacts. Performance is better than all the above filters.	40.5527	32.05061
Spatio Temporal filter #5	No artifacts. Reduces noise. Better performance than all the above filters.	13.6163	36.79023

MSE - Mean square error
PSNR - Peak signal to noise ratio

Table 4.4 Results of subjective and quantitative tests.

4.9 Summary

Digital filters with frame storage have been used for television picture noise reduction to bring signals from low quality sources to acceptable standards and to improve image processing stages like compression. Many recursive and non-recursive schemes with and

without motion estimation/compensation have been proposed by many researchers. to reduce the noise. But these schemes are applicable only for high frequency noise and can not be used for the low frequency throbbing noise as demonstrated in the experiments. But the fundamental problem of segmenting the image into moving and non-moving re-

Filter name	No. of operations per pel	Memory store (number of frames)
Temporal filter 1	5	5
Temporal filter 2	16	5
Temporal filter 3	13	5
Spatio temporal filter 1	max: 15 min: 2	5
Spatio temporal filter 2	max: 15 min: 2	5
Spatio temporal filter 3	max: 3 min: 2	2
Spatio temporal filter 4	max: 14 min: 12	2
Dennis filter	2	2
Spatio temporal filter 5	10	2

Table 4.5 Complexity analysis of filtering algorithms.

gions is the same for both types of noise. For the low frequency noise an alternative method was found to segment the image into moving and stationary regions. The global characteristics of the image from the histograms are used for segmenting moving and non-moving regions, after trying many filtering schemes including both recursive spatio-temporal and non-recursive temporal schemes.

Table 4.5 provides the processing overhead in units of number of operations per pel and the memory requirements in units of number of frames for each filter. It should be noted that for spatio temporal filters, depending on whether motion is detected at a pel or not, the number of operations performed on that pel will vary. For spatio temporal filters the table provides the maximum and minimum number of operations that can be performed on a pel. It should be noted that filter #5 involves a processing overhead of approximately 10 operations per pel. The processing time for a frame achieved on a 166MHZ Pentium machine is close to 170 milliseconds. This includes the frame capturing time of 110 milliseconds from the low cost camera. The algorithm is simple and effective for removal of low frequency temporal noise in the video sequences from a low cost camera. This method effectively reduces noise in stationary areas without affecting motion details of the image sequences and makes it an attractive solution as a preprocessing technique for improving the efficiency of subsequent image coding operations. Filter functionality can be improved if methods can be found for adaptive choice of the multiplier β and threshold T.

Chapter 5

Binary Image Compression

The different compression methods used for two level pictures and a proposed method for compressing the binary images to transmit them over low bandwidth telephone lines are discussed in this chapter. It is found that the proposed compression method is effective in compressing binary images. A simple compression algorithm for a real time application is required for transmitting images over telephone lines without losing temporal resolution, which is an important criterion for deaf sign language communication. Though sign language communication can be performed at a spatial resolution of 16×38 pels/frame image [Sperling, 1981], the experimental results are reported with 160×120 and 90×120 pels/frame images with the intention of providing better spatial as well as temporal resolution. Irreversible preprocessing techniques to remove noise-like points in the binary image and reduce the number of fluctuating points between successive binary image frames are discussed. A new preprocessing predictive filter, along with the techniques used in previous studies, and the effects of the combination of preprocessing stages to achieve better compression efficiencies are also presented.

5.1 Compression Schemes

Two level picture coding can be normally categorized into three types: one, two-dimensional and three-dimensional coding. Both the one and two dimensional schemes exploit spatial redundancy of the picture information. The three dimensional schemes utilize both spatial and temporal redundancies of moving image sequences. Various compression schemes used for binary images are discussed in the following sections.

5.1.1 One Dimensional Run-length Coding

In this scheme the distances between adjacent transitions in the picture level are transmitted. Figure 5.1(a) shows part of a single line. Starting at the beginning of the line,

(a)

00000011 00000010 00000001 00000101

(b)

Figure 5.1(a)Example line from a cartoon frame. (b)Run-length coded words.

runs of 3, 2, 1 and 5 pels are encountered between changes in the picture level. If fixed codewords are assigned, these values would be coded as shown in Figure 5.1(b). Having converted the picture information into series of run lengths, the statistical redundancy of these run lengths may be exploited. The statistical redundancy can be exploited by using variable word length coding [Huffman, 1952], wherein frequently occurring runs are given short codewords and infrequently occurring runs long codewords.

Huffman devised a method for constructing an optimal variable word length code, given an input message ensemble with associated probabilities of occurrence. The codewords allocated according to Huffman's algorithm yield lowest average data rate for the given input codeword statistics and represent a lower limit on the data rate. Huffman coding is dynamic in the sense that codewords and allocations are worked out afresh for each new set of input data. This dynamism increases the complexity that often results in modified versions of Huffman Codes. An example is the Modified Huffman Code used in the CCITT standard. This fixed coding scheme provides a set of agreed codewords that have been allocated according to a Huffman statistical analysis of several test documents. It uses two types of codewords: terminators and make-up words. Terminators are used for all run lengths up to N pels (N=64 in the CCITT standard). The make-up code words represent multiples of N. If a run length greater than N is encountered, it is coded using a make-up codeword plus a terminator.

Though this scheme is used in facsimile transmission of text documents, it cannot be used for low-resolution binary images. The reasons are:

1. Low-resolution binary images contain noise that comes from both the sketch extraction operator and the image source. This results in very short runs.
2. Implementing dynamic Huffman coding to take advantage of its good compression efficiency is computationally intensive and is unsuitable for real time applications.
3. Since the proposed system is scene independent, static Huffman Codes will not provide good compression efficiencies.
4. This coding scheme takes advantage of statistical redundancy only in one spatial direction.

5.1.2 Two Dimensional Coding Schemes

There are many two dimensional coding schemes available for binary images. Some of the coding schemes are discussed below. These coding schemes take advantage of two-dimensional spatial redundancy in a frame for compression.

5.1.2.1 Relative Address Coding

The principle of relative address coding is shown Figure 5.2. Transitions on the current line most often occur close to related transitions on the previous line. By coding the displacements between the current line's transitions and their neighbors on the previous

Figure 5.2 Example pair of lines from a binary frame.

line, this closeness is exploited. The result is that most transitions are represented by codewords denoting displacements, or relative addresses of 0, ± 1, or ± 2. When this method is combined with variable word length Huffman coding, significant compression is achieved.

Various researchers have proposed variants of enhanced relative address coding schemes. Edge Difference Coding [Yamada, 1979] groups transitions into pairs and segments the groups into three classes. The class to which a pair belongs determines its codeword. Another efficient scheme based on relative address coding is Modified READ (*Relative*

Element Address Designate). The position of the transition is classified into three modes depending on its relative position to four surrounding transitions and codewords are determined accordingly. This scheme has been adopted for both the CCITT Group 3 and 4 two-dimensional facsimile coding standards. Though relative address coding is effectively being used in facsimile transmission, the complexity of the algorithm and its use of only spatial information makes it unsuitable for the proposed real time application consisting of moving binary images.

5.1.2.2 JBIG (Joint Bi-level Image Group)

JBIG [JBIG web site, 1998] is an advanced compression scheme utilizing lossless predictive methods. Although designed for bi-level images, JBIG is also capable of compressing grayscale and color images. The JBIG algorithm [PixelMagic web site, 1998] is based on Arithmetic Coding. For each pixel in an image a "context" is derived from a specific fixed pattern of surrounding pixels. For a given context there is a statistical likelihood as to whether the current pixel is white or black. If the pixel agrees with this prediction, it can be represented in fewer bits of information than if it does not agree with what the context predicts. As each pixel is processed, the associated context is updated. The bits produced by this process are then coded using a technique known as "interval subdivision", all of which provides a significant compression improvement over Huffman Coding based algorithms. However the arithmetic coding used by JBIG is considerably more complex than the Huffman and run-length coding used by Group 4 compression, so it generally takes longer to decompress a JBIG image, which requires special purpose processors to compress and decompress the binary images. Since the aim of the present

research is to use existing computers to compress moving image sequences. simple schemes that exploit both temporal and spatial redundancies would be more suitable than the two dimensional relative address schemes and complex schemes like JBIG.

5.1.2.3 Vector Run-length Coding (VRC)

Wang and Wu [1992] reported algorithms that exploit the two dimensional redundancy in a more general way by coding vector or block patterns and run-lengths of column vectors. The basic idea of vector run length coding is to code the run length of a vector instead of just one pixel. where a vector is composed of n samples in the same column. It segments an image into separate regions. each starting with a varying block containing a set of continuously changing columns and followed by a constant block containing vectors with the same pattern as the last one in the varying block. The varying block can contain a fixed or variable column of vectors. The constant block is represented by its length. while the varying block can be represented in various ways. Depending on the coding method for the latter. various versions of VRC (Single Run-Length VRC. Double Run-Length VRC and Block Run-Length VRC) have been developed which have

Figure 5.3 Coding format of Single Run-Length VRC.

different trade-offs between compression gain and complexity. The coding format of Single Run-Length VRC is shown in Figure 5.3. R and V represent the codewords for run-length. and vector pattern. respectively.

Though the proposed algorithms are lossless, in applications where certain degrees of distortion are accepted, the vector pattern can be coded by a codebook containing only typical patterns. The generalized Lloyd algorithm, an adaptive technique for generating pattern code books for vector quantisation [Gray, 1984] can be employed to design pattern codebooks that minimize a given distortion criterion. Gray reports compression results for text documents. It has been found that binary images generated from low-cost cameras provide short runs of vectors, which makes VRC unsuitable for compressing binary images, since it utilizes only the spatial information of the image. Since the aim of the present work is to compress moving binary images, a compression scheme that uses temporal redundancy of moving images would be more suitable for this application.

5.1.2.4 Block Oriented Coding

In this scheme [Robinson, 1985], a two level picture is divided into $m \times n$ pel blocks, each having $2^{(m \times n)}$ pel configurations. These blocks are coded using a look-up table of variable-length codewords, designed from previously measured picture statistics that determine relative frequencies of occurrence. The result is that flat areas may be compressed into a few bits of information, while 'busy' areas require many bits.

Block location coding (or hierarchical coding) is a modification of the above method that overcomes the look-up table memory requirement problem and allows some exploitation of interblock correlation. A square block of size $N \times N$ is progressively subdivided until each sub-block is codable. The block may be halved or quartered at each subdivision, the

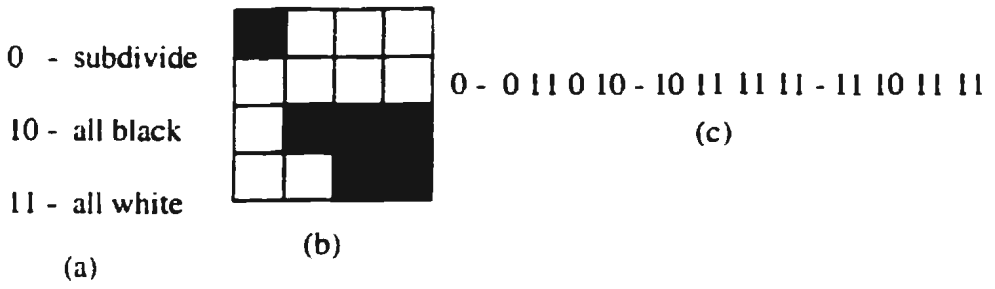


Figure 5.4(a) Sample codebook (b)Sample block (c) Block location coded bit sequences.

former case leading to a binary tree, the latter to a quad tree. This process leads to the generation of primitive blocks of small size and these blocks are coded using specific code words. Figure 5.4 shows a codebook, an example block and its coded bit sequences using quadtree coding and the codebook. The block is first subdivided to allow coding of the top-right block with an all-white code and bottom-right block with an all-black code. The top-left and bottom-left blocks are subdivided until each of their subblocks is individually coded. Though block oriented coding is simpler in implementation, a scheme that utilizes both spatial and temporal dimensions is needed.

5.1.3 Three Dimensional Coding

In these coding schemes the temporal dimension is considered, which makes them efficient for coding moving image sequences. Three-dimensional relative address coding and three dimensional block oriented coding are presented here.

5.1.3.1 Three Dimensional Relative Address Coding

In this scheme the position of the current transition is coded as a distance D relative to either the position of transition in the previous line or the previous frame. The decision is made using a threshold T. The value of D is then coded using a variable-length Huffman code. The codebook is derived from statistical analysis of the distribution of D over a typical test sequence. Though high compression efficiencies were achieved using this scheme [Robinson, 1986] the complexity of this scheme makes it unsuitable for the real time applications.

5.1.3.2 Three Dimensional Block-oriented Coding

Two dimensional block oriented coding can be extended to the third dimension simply by providing the blocks temporal 'depth'. This is achieved by taking a difference frame of the current and the previous frames and using spatial block oriented coding on the difference frame[Robinson, 1986]. This scheme is comparatively simpler than the three-dimensional relative address coding which makes it suitable for the real time application.

5.2 Proposed Compression Scheme

Haskell [1979] reported a technique called *Conditional Replenishment* for coding moving image sequences. In this technique, successive frame detail is not retransmitted if it is judged not to have changed significantly since the last frame, but simply redisplayed from the buffer at the receiver that holds that frame. Changed areas are retransmitted but with a resolution dependent upon subjective considerations and also on the available channel

bandwidth. Conditional replenishment is widely adopted for compressing image sequences. Cu-SeeMe, a videoconferencing application, also uses conditional replenishment for coding gray level images [Cornell University web site, 1998]. The first step in the Cu-SeeMe video encoding is to represent the video image using 4 bits/pixel (16 shades of gray). The image is then subdivided into 8x8 pel squares, which are treated as independent units. When a new frame of video is acquired, a square is transmitted if it differs sufficiently from the version of that square that was most recently transmitted i.e., if it differs from what the recipients are currently displaying (assuming no packet loss). The index used to determine how different a square must be in order to trigger updating is roughly the sum of the absolute values of all 64 pel differences in an 8x8 square.

In the proposed scheme, the data is coded in two stages: a temporal stage and a spatial stage. In the temporal stage, the conditional replenishment technique [Haskell, 1979] is used for binary images. The operations in the temporal stage are explained as follows:

1. The current frame and the previous frame in the image sequence are divided into $M \times M$ pel blocks.
2. For each block, the number N of pels that are changed in the current frame block from the corresponding block of the previous frame is noted.
3. The number N is checked against a threshold T.

   if ($N > T$) then the block is marked as achanged block and coded spatially as explained below.

It should be noted that if the value of threshold T is kept at 0, then the scheme becomes lossless. A high value of the threshold reduces the data rate and subjective quality of the

images at the receiver and a low value improves the subjective quality and increases data rate.

Each changed block is further coded in the two dimensional spatial stage using the hierarchical coding described in the previous section which is a recursive method. However, in this case an alternative approach is taken which can be said to be partially recursive. Initially the block is quartered, depending on whether it is individually coded or not. Each sub block is then progressively subdivided or quartered until it is completely codable leading to a quadtree configuration. The order of processing of the subblocks is top left square first, top-right second, bottom-left third and bottom-right fourth. This variation of hierarchical coding proved to be simpler to implement and faster than the original truly recursive hierarchical coding and hence used for spatial coding of changed blocks.

Coding of an example block shown in Figure 5.5(b) is explained as follows. A fixed codebook is used for coding which is shown in Figure 5.5 (a). The block is first subdivided to allow coding of the sub blocks. The example block is coded as explained below

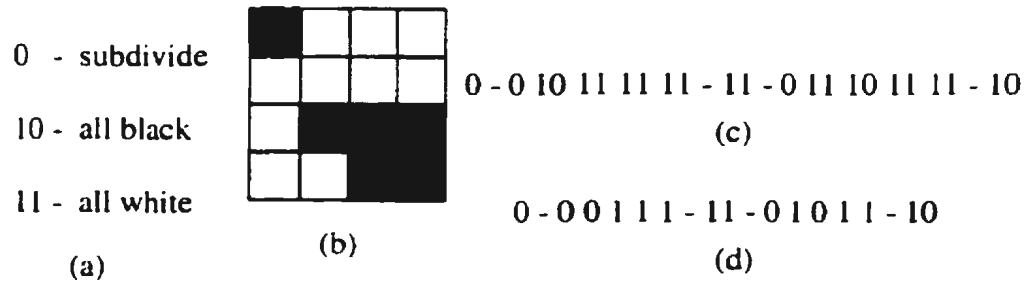


Figure 5.5(a) Sample codebook (b) Sample block. (c) and (d) Quad tree coded bit sequences.

1. The block is subdivided. This generates bit sequence '0'.
2. The top-left subblock is subdivided leading to four primitive blocks and each block is coded with all white or all black code. This process generates the bit sequence '0 10 11 11 11'.
3. The top-right subblock is completely codable with an all white code word. This generates the bit sequence '11'.
4. The bottom-left subblock is subdivided leading to four primitive blocks and each block is coded with an all white or all black code. This process generates bit sequence '0 11 10 11 11'.
5. The bottom-right subblock is completely codable with an all black code word. This generates code '10'.

The complete bit sequence generated for this example block is shown in Figure 5.5(c). To further compress the data, the bottom level of the quadtree structure is represented by only one bit: '1' for an all white block and '0' for an all black block. This results in the bit sequence as shown in Figure 5.5(d), which provides further economy of bits in representing the block. The result of application of this compression method is discussed in a later section.

5.3 Irreversible Preprocessing Techniques

Although a noise filter is introduced in the preprocessing stages of the gray level image sequences for reducing the throbbing noise, it is found during experiments that the binary

sketches generated contain many spurious noise-like points. The reasons for these points are :

1. Infiltration of part of the throbbing noise through the noise filter.
2. The sketch operator's sensitivity to noise due to its Laplacian nature.
3. Detection threshold values of the sketch operator not completely adapting to the time varying images from the camera.

These spurious points lead to transition of points between the previous and the current sketch frame. Since the compression scheme employed is a temporal based scheme, these fluctuations result in a reduction of compression efficiency. Hence, removal and reduction of these noise-like points improves both image quality in space and compression efficiency. Three preprocessing techniques are investigated that operate as part of the sketch generation system. They are isolated point removal [Robinson, 1986], threshold hysteresis [Robinson, 1986] and predictive filtering.

5.3.1 Isolated Point Removal

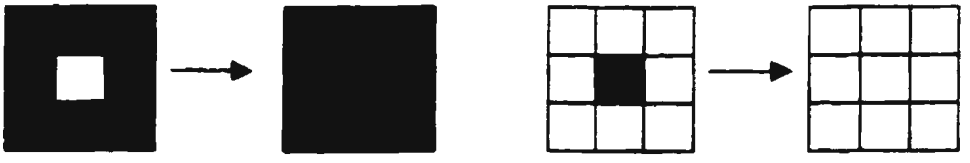


Figure 5.6 Isolated point removal scheme.

In each frame, any black point surrounded by white points is made white and any white point surrounded by black point is made black as shown in Figure 5.6. Isolated point re-

moval is built as a separate stage after feature extraction. It should be noted that isolated point removal works on only the spatial information of a frame.

5.3.2 Threshold Hysteresis

Since the absolute threshold value determines a high percentage of feature points in a binary frame. threshold hysteresis is used to alter only the value of the absolute threshold. The basic purpose of threshold hysteresis is to maximize the possibility of keeping a feature point the same as the frames progresses in time. This results in a reduced number of fluctuations. It should be noted that if a pel is black in a binary image plane. it lies below the absolute threshold value. If the pel is white. it lies above the absolute threshold value. The operation of threshold hysteresis for an absolute threshold value T is described as given below.

1. If a pel is black in the previous sketch frame. corresponding pel in the current frame will be thresholded at a slightly higher value $T + \delta$ where δ is a small number.
2. If a pel is white in the previous sketch frame. thecorresponding pel in the current frame will be thresholded at a slightly lower value $T - \delta$.

The above operations increase the possibility of a pel remaining the same in time. Threshold hysteresis is built into the feature extraction algorithm and it should be noted that threshold hysteresis requires temporal information.

5.3.3 Predictive Filter

This new work can be considered as an extension of the threshold hysteresis method. This method uses temporal information to reduce the temporal fluctuations of successive frames and is based on prediction. The algorithm of the filter is given as follows.

1. Each frame is divided into $M \times M$ pel blocks.
2. For each block in the current frame and the corresponding block in the predicted frame, prediction error E is calculated as follows:

$$E = \sum_{x=1}^{M} \sum_{y=1}^{M} | u(x, y) - p(x, y) |$$

   where $|u(x,y) - p(x,y)|$ is the absolute difference, $u(x,y)$ is the pel in the current frame at position x,y and $p(x,y)$ is the value of the pel in the predicted frame at position (x,y).
3. The value E is then compared with a threshold T. If the value of E is greater than T, then the feature extraction operator is applied to the current frame block and the predicted frame block is updated using the current frame block.

It is found that a value of 4 for T produced subjectively best results. If increased more than this value, the predictive filter introduces spatial noise and if decreased, the filter is found to be less effective in reducing changing points between successive frames. The Predictive filter technique is built into the feature extraction algorithm. It also reduces the number of feature points to be found in the current binary frame, though this property depends on the amount of motion between frames, which makes it work faster than the op-

erator with only feature extraction. The effect of application of this simple filter is discussed in a later section.

5.4 Tests on Preprocessing Stages

Since the selected compression scheme is based on temporal information, compression efficiency depends heavily on the fluctuating points between successive frames. Fewer fluctuating points results in better compression efficiency and more

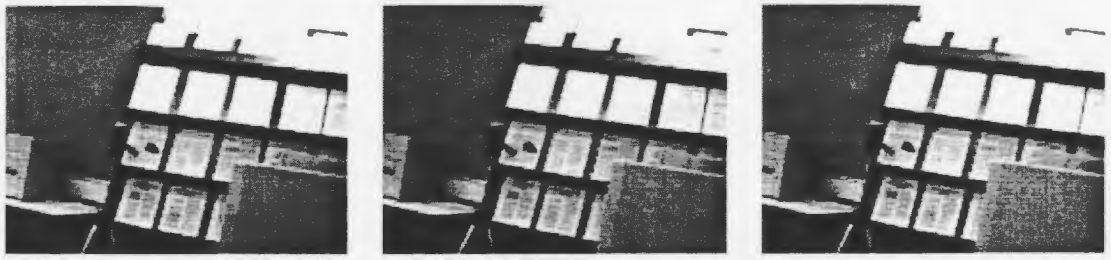


Figure 5.7 Successive frames from a static scene.

Figure 5.8 Successive frames from a sequence with motion.

fluctuating points reduce the compression efficiency. In the tests, two sequences are used to study the effectiveness of the preprocessing techniques in reducing changing points between two successive frames. One sequence contains 50 frames of a static scene with throbbing noise and another sequence contains 50 frames of a moving object. Figure 5.7 shows successive frames from the static test sequence. Throbbing noise in the sequence

can be observed by looking at the light region in the right topmost corner of the frames. The change in the area of this light region indicates the presence of throbbing noise. Figure 5.8 shows successive frames in a sequence with a moving object with negligible throbbing noise and motion in all the 50 frames.

The number of pel changes between successive frames represents the number of fluctuating points between successive frames. To improve the compression efficiency, preprocessing stages should try to keep the number of fluctuating points fewer. To estimate the efficiency of the preprocessing stages in reducing the fluctuating points, the number of pel changes between successive frames is plotted over time for both the test sequences and shown in Figures 5.9 and 5.10. All combinations of preprocessing stages are tested on these sequences.

It should be noted that the predictive filter and threshold hysteresis are built into the feature extraction algorithm and turned on/off when required while taking data. Isolated point removal technique follows the feature extraction operation. Figure 5.9 shows the results of an application of combinations of preprocessing stages on the static scene test sequence and Figure 5.10 shows the results on the moving object test sequence.

5.4.1 Results Discussion (Tests on Preprocessing Stages)

Figure 5.9(a) shows the effectiveness of the noise filtering algorithm in reducing the throbbing noise resulting in a reduced number of pel changes between two successive frames. In an ideal case, the values of the y coordinate should be zero for a static scene.

However the throbbing noise from the camera and the high frequency feature extraction operator makes the number of pel changes vary a lot between successive frames. The temporal filter, though not completely eliminating the throbbing noise, does reduce much of the effects of the noise can be seen from Figure 5.9(a).

Figure 5.9(b) and Figure 5.9(c) show the combinations of preprocessing stages with and without the temporal noise filter respectively. Almost identical results are shown for the combinations NF+PF, NF+IPR+PF and NF+IPR+PF+TH. From these results it can be concluded that the inclusion of the predictive filter and the temporal noise filter as pre-processing stages is effective in reducing the number of pel changes between successive frames.

Figure 5.10(a) shows the results with and without the temporal noise filter on a sequence with negligible throbbing noise where most of the pel changes between successive frames are due to actual motion of the object. As expected the temporal noise filter has no effect on the number of pel changes between successive frames. Figure 5.10(b) and Figure 5.10(c) show the combinations of preprocessing stages with and without the temporal noise filter respectively. It can be seen that almost all combinations of preprocessing stages produce the same results.

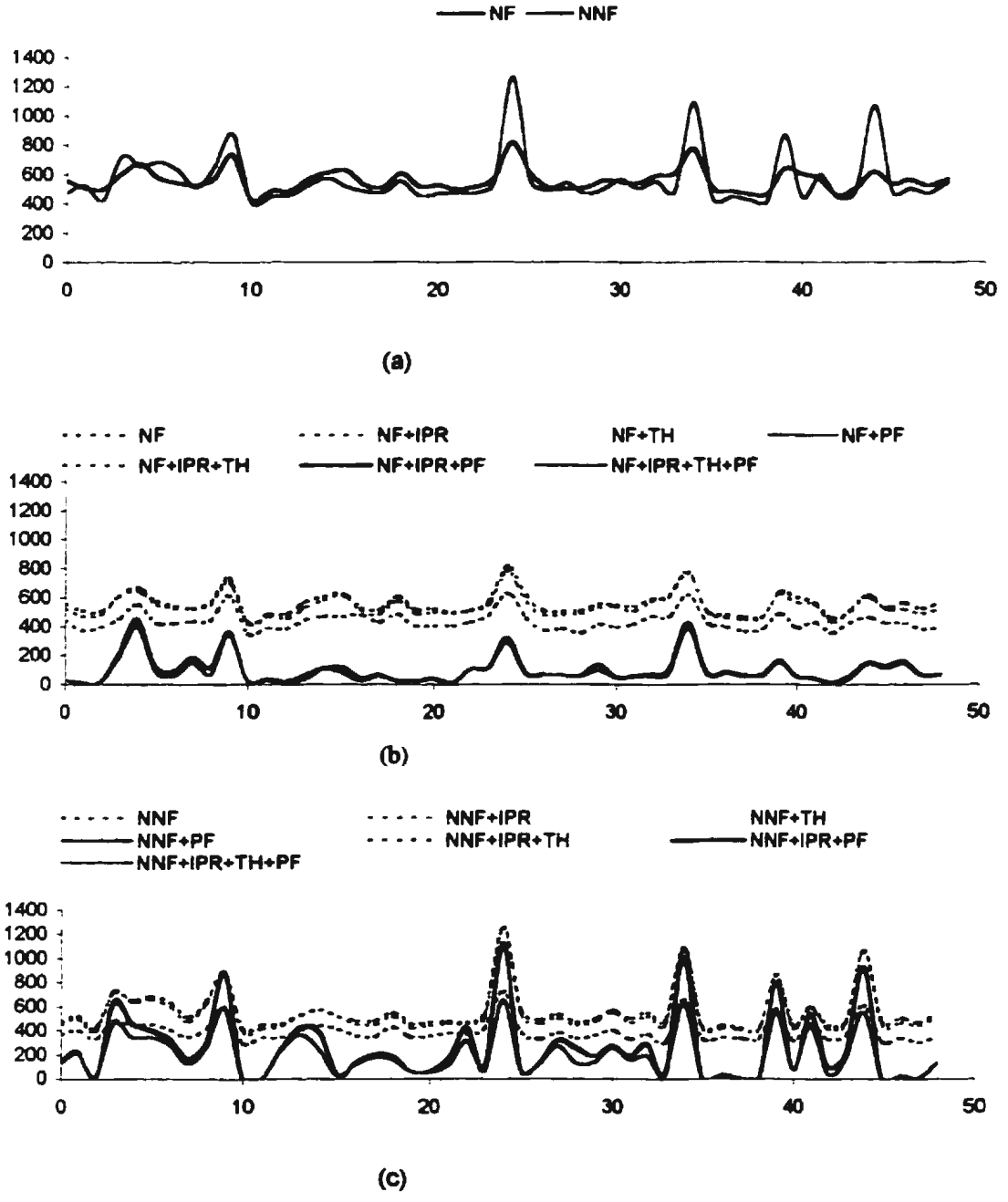


Figure 5.9 Number of pel changes between successive frames after applying various combinations of preprocessing steps in a static sequence.

NNF - Only feature exraction operator
NF - Temporal noise filter and feature extraction operator
IPR - Isolated point removal TH - Threshold hysteresis
PF - Predictive filter
X axis - Frame number Y axis - Number of pel changes

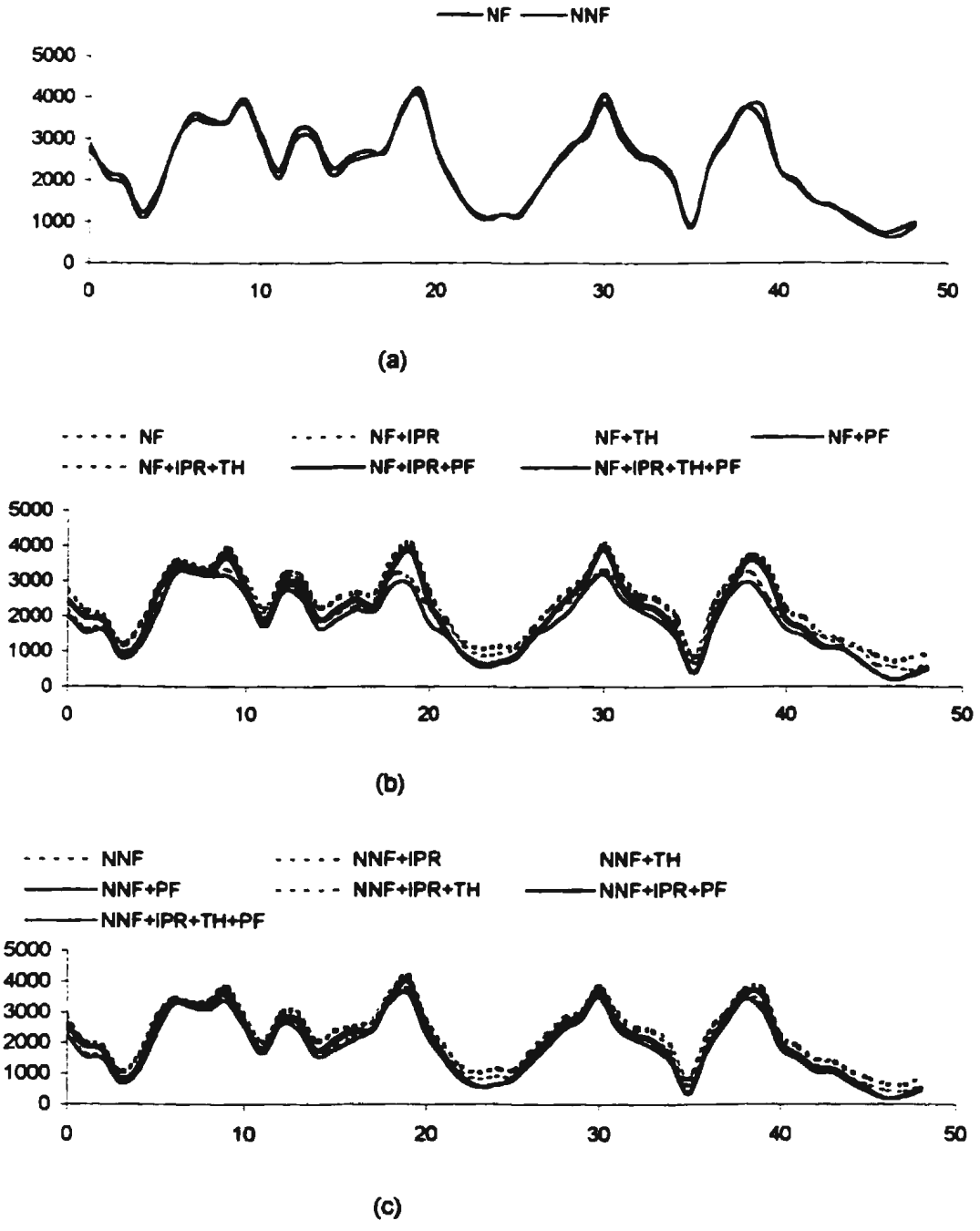


Figure 5.10 Number of pel changes between successive frames after applying various combinations of preprocessing steps in a moving sequence.

NNF - Only feature extraction operator

NF - Temporal noise filter and feature extraction operator

IPR - Isolated point removal TH - Threshold hysteresis

PF - Predictive filter

X axis - Frame number Y axis - Number of pel changes

However, during the experiments it is noted that both the predictive filter and threshold hysteresis produce spurious points as the frames progress in time for a moving scene because of their temporal nature which degrades spatial resolution. This effect makes the isolated point removal technique important as a preprocessing stage since it improves the spatial resolution. Though both threshold hysteresis and the predictive filter operate on the temporal information between successive frames, the predictive filter provides much better results than threshold hysteresis.

Process	Time
Feature extraction	O(MxN)
Isolated point removal	O(N)
Threshold hysteresis	O(N)
Predictive filtering	O(N)
Compression	O(NlogN)

Table 5.1 Order of running times of preprocessing stages

Table 5.1 shows the order of running time of each preprocessing stages and the compression algorithm used. In the table, M is the number of pels in the extraction operator and N is the number of pels in the frame. It should be noted that the time units presented in the table are the time taken if each process acts upon the picture individually. To measure the actual computation time for each preprocessing stage, a more informal test setup is used. A fixed number of frames are captured from the camera and preprocessing is applied to the frames. The average processing time for a single frame is calculated from

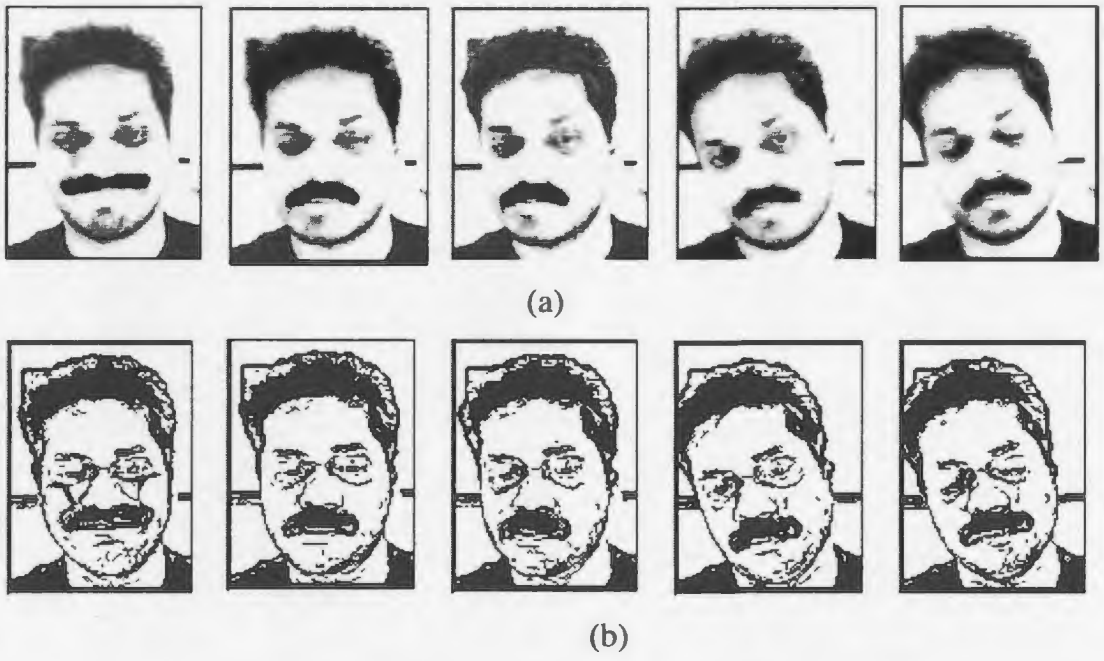
the total time taken for the fixed number of frames. From the tests, it is found that the feature extraction operation takes 40 milliseconds, isolated point removal method takes 20 milliseconds and the temporal noise filter takes 30 milliseconds. If threshold hysteresis is turned on feature extraction takes 45 milliseconds and if the predictive filter is on, it takes 30 milliseconds, again depending on the motion information between successive frames. It should be noted that all these measurements are taken on a 166MHZ Pentium machine and are approximate. Considering all the points discussed above, it is decided to keep the temporal noise filter, predictive filter and isolated point removal as the preprocessing stages which result in better temporal and spatial resolution of the binary frames.

5.5 Compression Tests

For compression tests two sequences are taken. Sequence #1 contains 50 frames in which a face covers almost complete the frame and the image information changes in all the frames. In other words each frame shows a change in the facial features of the subject. Sequence #2 contains 50 frames in which a subject, with no knowledge of deaf signs, moves his hands and fingers similar to the movement caused in the case of sign language communication. To bring the facial and hand features clear, both sequences are taken with light from two table lamps on both sides of the subject and focussed on the subject. It should be noted under this condition there is negligible throbbing noise and the feature extraction algorithm brings out features better than in ordinary overhead ceiling lighting. The frames are captured from the video camera at a size of 160×120 pels/frame and reduced to 90×120 pels/frame. The reduced frame size is used for further processing.

5.5.1 Discussion of Results

Successive sample frames from sequence #1 are shown in Figure 5.11(a) and sample frames from sequence #2 are shown in Figure 5.12(a). Figure 5.11(b) and Figure 5.12(b) show the sketches of the frames with the temporal noise filter, predictive filter and isolated point removal as the preprocessing stages. For comparison purposes single run-length VRC is implemented with a vector size of 4×1 pels and, results of a previous similar experiment is considered[Robinson, 1986]. The data rates for the proposed three dimensional compression scheme are shown in Table 5.2 and Table 5.3. The data rates for the implemented single run-length VRC are shown in Table 5.4. These rates are calculated by averaging data rates over 50 frames.



(a)

(b)

Figure 5.11 (a) Sample frames from sequence #1 (face sequence). (b) Sketches of sample frames.

(a)

(b)

Figure 5.12 (a) Sample frames from sequence #2 (signing sequence). (b) Sketches of sample frames.

block size	bits/pic	bits/pel
4	5000	0.462
8	5248	0.485
16	5240	0.485

Table 5.2 Data rates for sequence #1 (signing sequence) using the proposed compression scheme.

block size	bits/pic	bits/pel
4	3328	0.308
8	3456	0.320
16	3472	0.321

Table 5.3 Data rates for sequence #2 (face sequence) using the proposed compression scheme.

Sequence Number	Bits/pic	Bits/pel
1 (signing)	10168	0.942
2 (face)	12016	1.113

Table 5.4 Data rates for the test sequences using Single run-length VRC.

Coding scheme	3-D RAC	BLC
Best Data Rate	0.201 bits/pel	0.229 bits/pel

Table 5.5 Results for Robinson's sequence #2 (signing sequence).

Coding scheme	3-D RAC	BLC
Best Data Rate	0.201 bits/pel	0.229 bits/pel

Table 5.6 Results for Robinson's sequence #3 (face sequence).

Sequence #1 has motion in 75% of the frames. sequence #2 has motion in almost all frames and sequence #3 has high-resolution facial features. Sequence #2 and sequence #3 in Robinson's research is similar to sequence #2 and sequence #1 of the current research respectively. Table 5.5 and Table 5.6 shows the results of Robinson's research.

As seen from Table 5.4, SVRC provides higher data rates. This can be explained by the fact that SVRC is not using the additional temporal information from the moving cartoon sequences and it only removes the two dimensional spatial redundancy. Since the run-length of each vector from the test sequences is very short, it is not effectively represented by SVRC. The results reported by Robinson are better than the results achieved in this current research. For the signing sequence Robinson's results are 50% better and for the-face sequence they are about 10% better than the current research. However, it should be noted that Robinson's results were obtained after each scheme's output was fed to a statistical analysis program to determine optimum Huffman codeword assignments based on the probability of occurrence of the input. This process enabled the author to achieve the minimum possible data rate for each scheme.

Though the data rates achieved in the present research is higher than the past research, the present compression scheme proposed is simple and faster which makes it suitable for real time applications. On a 166Mhz Pentium machine the encoder takes approximately 25 milliseconds to compress a binary frame of size 90×120 pels/frame and 8 milliseconds to decompress it, which makes the proposed compression scheme suitable for deaf sign language communication, which is sensitive to temporal resolution. The time measurements are taken by averaging the total time over a fixed number of frames. The compression scheme also provides less error propagation in the data than the variable length coding schemes used by Robinson. The proposed scheme is block oriented. Propagation of errors is limited to individual blocks in the frame. In a way, this scheme provides limited error resilience to the data because of its block oriented nature.

5.6 Summary

This chapter has reviewed various compression schemes used in binary image processing and proposed a simple three-dimensional compression method that effectively removes the spatial and temporal information redundancies of moving binary frames. The proposed method is simple and achieves good compression efficiency. This makes it suitable for real time applications. Three irreversible preprocessing methods, which include a new predictive filter, are also investigated to remove noise like points and to reduce the number of changing points between successive frames. From the results reported, it is concluded that the combination of the temporal filter discussed in Chapter 4, a predictive filter, isolated point removal and the proposed compression scheme is effective in achieving good compression results.

Chapter 6

User Interface Design and Programming Issues

User interface design is becoming increasingly important and complex. Graphical interfaces dominate user interface design more than command line interfaces. They are easy to learn and use. With graphical interfaces full-screen interaction is possible rather than line-oriented interaction with command line interfaces. The interface developed for the complete system along with some suggestions for improvements to the present interface are discussed in this chapter, along with the programming issues involved in the system design. This chapter explains the modular object oriented approach used and functions of various classes used in the system.

6.1 User Interface of the System

The user interface design of the system developed for deaf sign language communication is oriented towards a research environment. Though the system built is mainly used for research purposes, many of the controls are understandable by ordinary user and the sys-

tem has a Graphical User Interface (GUI). The interface model used for the system is a control panel model and user interface objects are provided as buttons, switches, menus, displays and sliders. The explanation for each of these objects is as follows.

(1) *Buttons* Picking a button always causes a single action to be initiated.

(2) *Switches* Switches may be set at a number of positions to configure a system or move a system from one state to another.

(3) *Menus* Menus are collections of buttons or switches which may be made visible and selected. Picking a menu title causes a pull-down menu to appear.

(4) *Displays* Displays are areas where graphical or textual information may be displayed. The user interface designer controls whether or not a display is editable.

(5) *Sliders* Sliders are input devices to set a specific inputvalue. The user drags the slider along the scale to set the required value.

(6) *Up Downs* Displays a pair of arrow buttons that the user can click. Up-down objects are usually used with a companion object, called a buddy. When the user clicks on the up-down object, a value in the buddy is modified. For example, a up-down object can be used to scroll tabs in a tab control object into view, or change a number displayed in a text box object.

6.1.1 Controls Used in the System

The system is used to capture grey level images from the camera and extracts binary images. To improve the subjective quality of the images, pre-processing can be performed on the binary images. The images are then compressed and sent over Internet Protocol

(IP) networks or telephone networks depending on the user's choice. The complete controls of the program are shown in Figure 6.1, and the whole diagram is divided into parts.

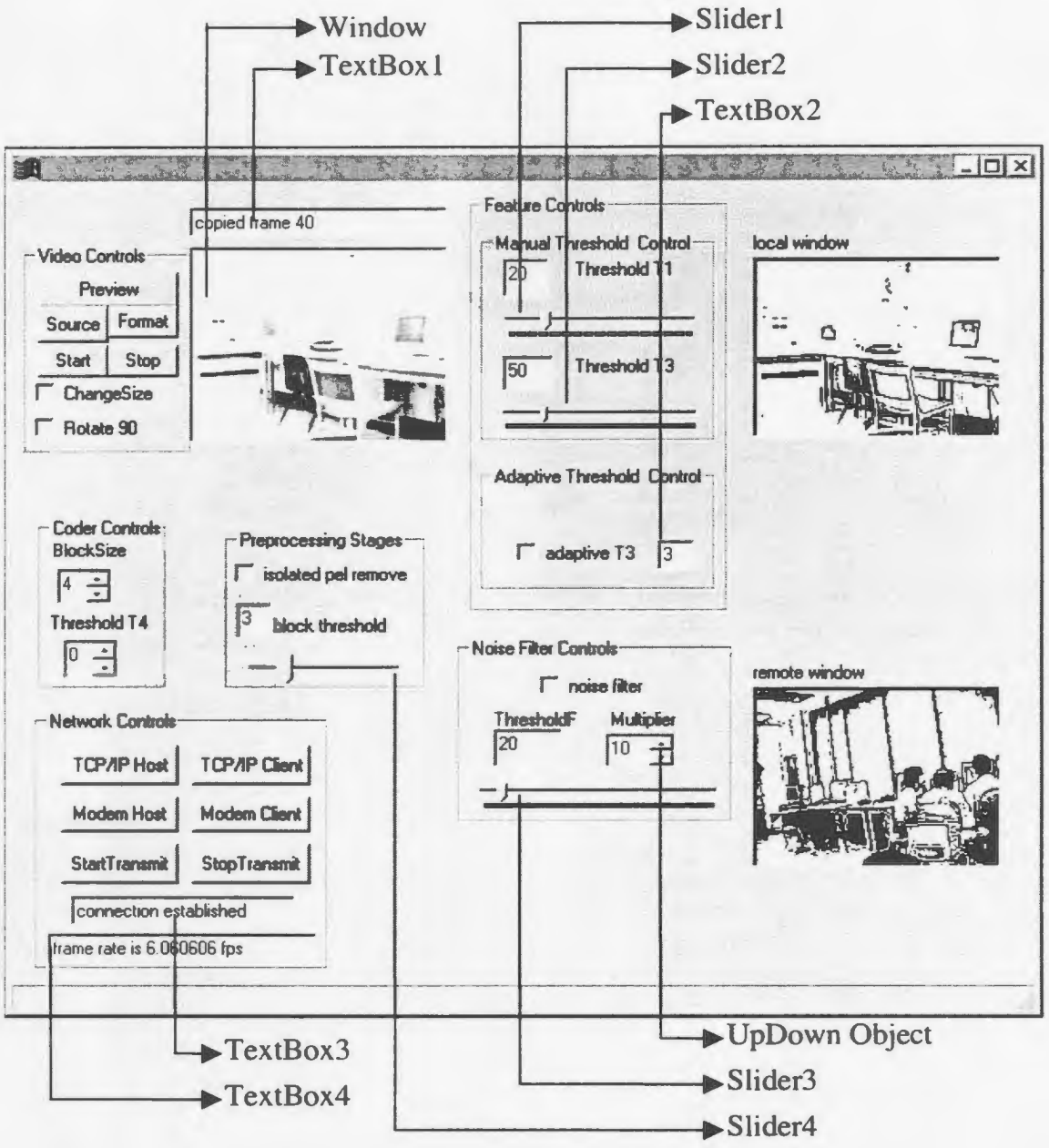


Figure 6.1 Program window showing all the controls

The parts are as follows.

(1) Video Controls

(2) Feature controls

(3) Noise filter controls

(4) Binary image pre-processing controls

(5) Binary image coder(compression) controls

(6) Network transmission controls

Video Controls are used to change the image characteristics of the gray level images including camera controls. The *Preview* button is used for viewing the images captured from the camera. Clicking the *Format* button opens a dialog box which allows the user to change the size of the image and number of gray levels in an image frame. The *Source* button opens a dialog box through which user can change the camera parameters such as brightness, contrast and white balance. The *Change Size* switch is used to crop the picture of size 160×120 pels/frame to 90×120 pels/frame. This allows capture of only the middle portion of a 160×120 pels/frame. The *Rotate 90* switch is used to change the landscape image format captured from the camera to portrait format. It effectively rotates the image right hand side by 90 degrees. The *Start* button is used to start the capture of images and *Stop* is used to stop the capture. The *Window* is the preview window. The *TextBox1* shown is used to display text information on the number of frames captured and start and stop button actions.

Feature controls are used for controlling the thresholds of the feature extraction operator explained in Chapter 3. From a user's point of view they are used to change the subjec-

tive quality of the cartoon images. Moving *Slider1* along the scale from 0 to 100 controls *Threshold T1*. *Slider2* is used to manually control *Threshold T3*. The scale used for this slider is from 0 to 250. *Threshold T3* can be adjusted adaptively by selecting the *AdaptiveT3* switch and inputting a value from 1 to 10 in the *Text Box* next to the switch. Selection of the adaptive switch disables manual control of *Threshold T3*.

Noise processing controls are used to control the filter used for reducing low frequency throbbing noise explained in Chapter 4. Noise filter switch is used to select the filtering operation. *Slider3* is used to adjust the threshold parameter of the filter which can be varied from 0 to 250 and *up down* object is used to select the multiplier parameter of the filter which can be varied from 0 to 100.

Binary image pre-processing controls are used to control the pre-processing stages of isolated point removal and predictive filter. *Isolated pel remove* switch is used to remove noise like points in the cartoon. *Slider4* is used to control the parameter for the predictive filter explained in Chapter 5. The scale can be changed from 1 to 16. Binary image coder controls are used to control the coder that compresses the binary images. The *Block Size* up down object, which can be adjusted from 1 to 64, is used to decide on the size of the fixed size blocks each frame has to be divided into and the *ThresholdT4* up down object which can be adjusted from 0 to 16 is used decide the threshold parameter for the compression system. It should be noted that the block size values should always be progressively divisible by 4.

Network transmission controls are used to establish a connection between two computers either through telephone lines or through IP networks. To initiate the connection one machine has to act as a host and other machine as a client. *TextBox3* informs the connection status and number of frames transmitted from the computer. Clicking the *TCP/IP Host* button makes a machine the host and informs the status in the *TextBox3*. Clicking the *TCP/IP Client* button opens a dialog text box. The user enters the IP address of the machine. The status of this operation is indicated through *TextBox3*. Clicking *Modem Host* button opens a dialog box through which the user puts the modem in the answering mode. The *Modem Client* button click opens a dialog box in which the user supplies the telephone number of the host. The modem in the client machine dials out the number to establish the modem connection through telephone lines. The connection status is displayed in the *TextBox3* of the respective machines. *TextBox4* is used to display the frame rate information which depends on the processing power of the computers and the bandwidth of the channels used. Clicking the *Start Transmit* button transmits binary images across the network. A *Stop Transmit* button click stops the transmission. The *Local Side* window displays the binary images transmitted from the local computer system and the *Remote Side* window displays the images received from the remote computer system.

6.1.2 Operating the System

The steps involved to use the system for sign language communication between two machines are as follows :

(1) Run the program on both the machines in the windows 95 environment, opening up the window shown in Figure 6.1.

(2) Establish connections on either IP network or telephone lines. This is done by clicking the host button for the network on one machine, then clicking the client button for the same network option on the client machine. Supplying necessary information to the dialog box establishes the connection between two machines.

(3) To start transmission, click the Start button in the video control and the Start Transmit button in the network control. This initiates full duplex transmission of binary images between the two machines. Transmission can be stopped, by clicking the Stop Transmit button in the network controls.

(4) The controls explained above can be used to tailor the quality of images and the rate of transmission.

(5) To exit the program, click the Close button of the program window.

6.1.3 Improvements to User Interface Design

As mentioned previously the user interface design of the system is not optimised for ordinary users. The system still needs input from the users. Even from the designer's view point, some of the improvements needed in the interface are observed and they are discussed in this section.

- The names of the buttons can be changed to be more usable. For example, in the feature controls instead of using names threshold1 and threshold3, the names can be changed to reflect their functionality from an ordinary user's point of view. However, since the project is in research phase and current names are suitable for the research environment, they are kept as they are.

- Currently to establish a connection, the machines have to be designatedas host and client. This requires communication between end users who are not in the same premises. This problem is not addressed by the present system.
- The client has to type in the name of the host machine in the network controls. This can be eliminated, if the system can have a directory structure which can be pulled down by a menu system. This avoids typing mistakes performed by the user and the user can directly select the name using the mouse as the pointing device.
- A text chat mode can provide feedback between end users on the quality of transmission. Currently the remote user has no way of knowing how his/her binary image is being received at the other end.
- Though the steps involved are simple, an help manual with simple and understandable terms can be provided.

Though the system is found to be effective in an operational sense for full duplex transmission of binary images, it needs many improvements and feedbacks from ordinary users, which will be addressed in the next versions of the system.

6.2 Programming Issues

This section describes high level design issues of the system and describes the functionality of the different classes that are parts of the program . The system uses some of the classes from a library called MCLGallery [Cheng, 1998]. MCLGallery is a software library and framework designed by Cheng and Robinson[1998], to meet the various research needs of the Multimedia Communications Laboratory(MCL). It mainly targets development work in the Windows95 computer platform. PowerSoft's Power++ is cho-

sen for use with MCLGallery and the designed system. Power++ has a strong optimising C++ compiler (based on Watcom C++) and provides a comprehensive, easy to use Rapid Application Development Environment(RAD). The following sections describe the classes used in the system and their functions.

6.2.1 Functional Description of the System

Grey level images are progressively captured from the camera. These grey level frames are then filtered using the temporal filter discussed in Chapter 4. The filtered frames are fed to the cartoon extraction function to obtain binary cartoon frames. The binary frames are further processed for better subjective quality using pre-processing stages implemented as functions. The pre-processed binary images are fed to the encoder object, which compresses them into a structure, which can be transmitted over the network. The received object is then decoded and painted in the remote window of the computer system. The functional diagram of the system is shown in Figures 6.2(a) and 6.2(b).

6.2.2 Description of Classes and Their Functions

The whole system is implemented as classes and objects of the classes are instantiated in the main class to do the proper functions. Appendix B lists all the classes used in the system design and important methods of some of the classes. The functionality of the classes are explained below:

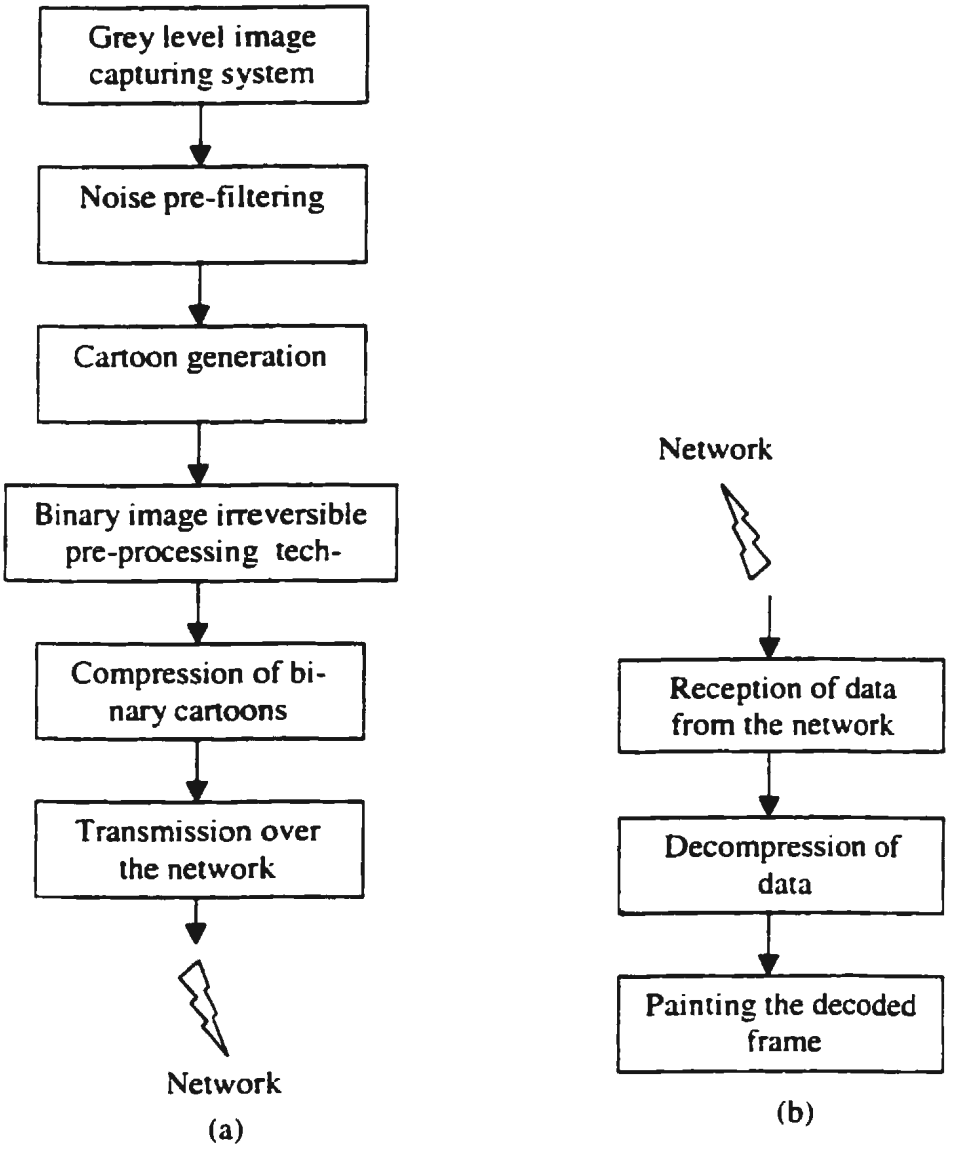


Figure 6.2 (a) Transmission process. (b) Reception process.

(1) *Form* is the main class. An object of this class instantiates the objects of other classes and controls all the functions of the system.

(2) *MCLVideo* class captures video frames from a video device. This class is defined by MCLGallery.

(3) *MCLBitmap* class is used to present and manipulate the image information. This class is defined by MCLGallery.

(4) *MCLData* class is used to represent the image data as a byte buffer for further manipulations. This class is defined by MCLGallery.

(5) *MCLNet* class is used to provide network services over different types of networks. This class is defined by MCLGallery.

(6) *Encoder* class is used to compress the image frames.

(7) *Decoder* class is used to decompress the data.

(8) *NoiseFilter* class is used to filter the noisy images.

Figure 6.3 shows the main class, form, of the program. As illustrated in the Figure 6.3 all the objects of the classes mentioned above are instantiated in the form object. Grey level images are captured from the camera using the MCLVideo object, which presents the images in MCLBitmap form to suit the manipulation of frames by further processes. The noise filter object takes the noisy frame and produces filtered frame. The function *F(Sketch Extraction)* extracts the feature points and outputs the binary cartoon frame as a MCLBitmap structure. *F(Pre-processing)* processes the input binary frame and produces an output frame as a binary frame. The processed MCLBitmap frame is compressed using the Encoder object as a MCLData object, which is then transmitted across the network using MCLNet object.

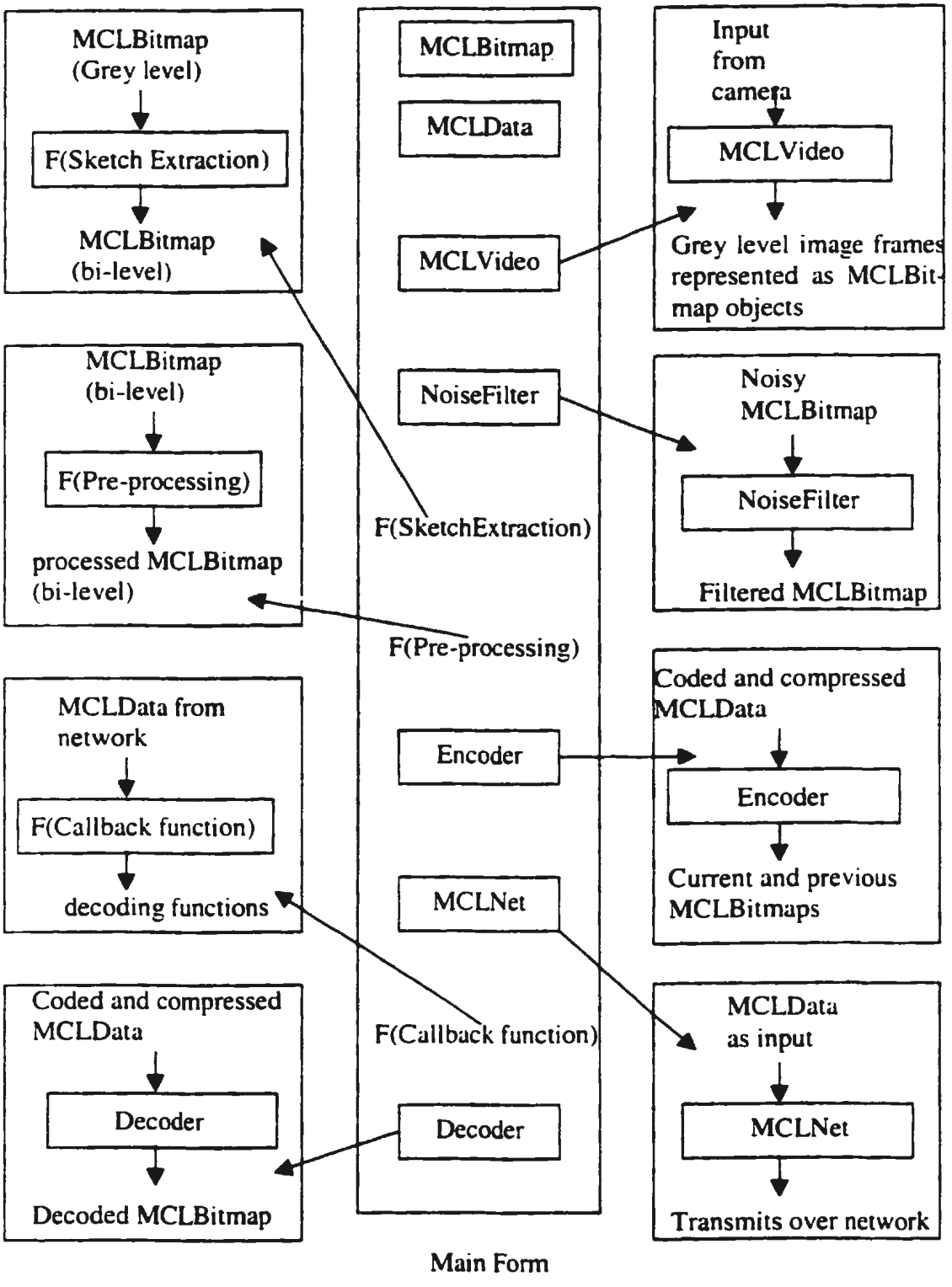


Figure 6.3 Classes used in the design of the system.

When the transmitted data reaches the remote machine, it invokes the network call-back function *F(Call-back function)*. This function in turn initiates the decoding process by calling the decoder object. The decoder object takes the MCLData structure, decompresses the data and produces the decoded MCLBitmap binary frame. The decoded frame is then displayed in the remote window of the machine. Currently the system operates in such a way that, when there is communication between machines of mismatched processing powers, the faster machine operates at the same speed as the slower machine. Appendix C describe, the flow control problem and a solution to this problem. It also explains the methods tried to improve the timing of the system processes, which exploit the multithreaded property of the system.

6.3 Summary

This chapter explained the graphical user interface design used for the system. Though the interface design mainly concentrates on the interests of a research oriented environment, suggestions are provided along with the drawbacks, to improve the interface, so the system can be used by ordinary users. The programming issues involved in the system design are briefly explained, in addition to the functionality of the classes used in the system.

Chapter 7

Subjective Tests

Deaf people use three modes for communication between themselves and hearing people. *Signing* is made up of signs, which are gestures made primarily with the hands, arms, face and other parts of the body. *Finger spelling* is used for technical words, place names, etc., where each letter has to be communicated. Some deaf people also use *lip reading* to communicate with hearing people. In order for a system to be used for deaf sign language communication, it should support all the three modes. The frame rates, or temporal resolution, achieved by using the current system over phone lines are 7 to 8 frames per second for 80×60 pels/frame pictures, 4 to 5 frames per second for 160×120 pels/frame pictures and less than a frame per second for 320×240 pels/frame pictures. It should be noted that the rates mentioned above are for full duplex, i.e. bi-directional, communication. This chapter explains the experiments conducted on the developed system to test its effectiveness for deaf sign language communication. Two deaf subjects were involved in the experiments. The following sections explain the experimental setup and discuss the results of the experiments.

7.1 Test Setup

The system was tested to find its efficiency by changing three parameters: lighting environment, size of the picture and communication channel. Each parameter was varied in the following manner:

- Two types of lighting environments were used in the testing. They were the *normal environment* and the *enhanced environment*. In the normal environment, the lighting is as expected in a working environment. In other words this lighting is a combination of light from over head fluorescent lamps and natural lighting. In the enhanced environment, two table lamps were focussed in such a way as to bring more facial features from the subjects. In both the environments, a dark background was used.
- Four sizes of pictures were used. They were *80×60, 160×120, 120×160* and *320×240 pels/frame*.
- Two types of communication channels were used in the testing. They were *low bandwidth telephone channels* and *high bandwidth local area IP (internet protocol) networks*.

The conditions under which the system was tested are provided in the Table 7.1.

Condi-tionNum-ber	Environment	Communication Channel	Picture size Width×Height Pels/frame
1.	Normal	Local Area Network	80×60
2.	Normal	Local Area Network	160×120
3.	Normal	Local Area Network	120×160
4.	Normal	Phone lines	80×60
5.	Normal	Phone lines	160×120
6.	Normal	Phone lines	320×240
7.	Enhanced	Phone lines	80×60
8.	Enhanced	Phone lines	160×120

Table 7.1 Test Conditions.

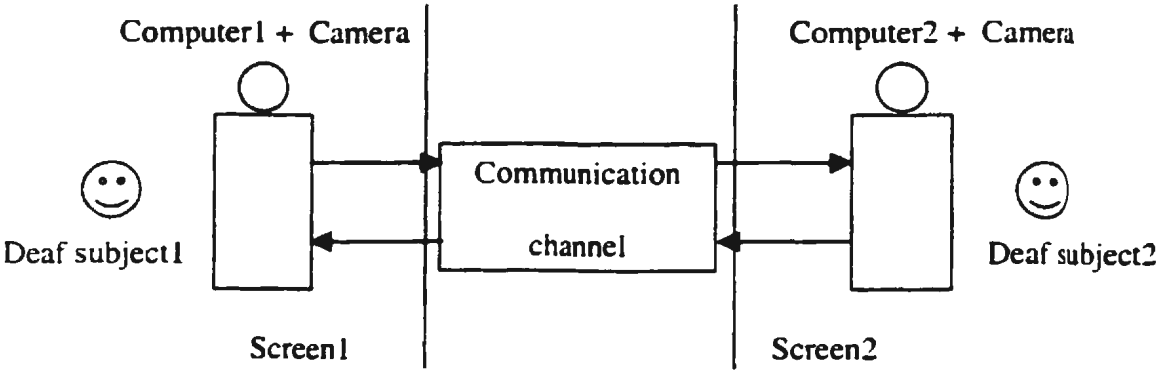


Figure 7.1 Experimental setup for deaf communication.

The testing arrangement is shown in Figure 7.1. Two computers in the same room, but separated by screens, were used in the testing. Computer1 is a Pentium based machine with a processor speed of 166 MHz with 32 Mbytes of RAM(Random Access Memory) and is fitted with a modem connected to a phone line. Computer 2 is a Pentium machine with a processor speed of 200 MHz and 64 Mbytes of RAM and is fitted with a modem connected to a phone line. The cameras used in the test environment are gray level low cost cameras from Connectix Corporation.

Two signers, Mr. Myles Murphy and Ms. Jeannie Leonard, from the Newfoundland Co-ordinating Council on Deafness, agreed to be the subjects for testing the system and are familiar with American Sign Language. They will be referred to as subjects in the following sections. Each subject was provided with eight sentences for signing tasks and eight city name - number pairs for finger spelling tasks, to be used in eight different test conditions listed in Table 7.1. Test sentences and city-number pairs provided to subject1 and subject2, for different test conditions, are listed in column 2 and column 4 of Table 7.2, respectively. Before starting the experiments, the subjects were informed about the procedure of testing and asked to familiarize themselves with the binary sketches they would expect during the testing procedure. There was no restriction on the color or texture of the fabric worn by the subjects. One subject was wearing a white T-shirt and other subject was wearing a patterned shirt. The testing procedure was as follows.

a) Establish the communication channel with the parameters specified in the test condition.

b) Subject1 signs the sentence and pauses for subject2 to recognize and write the sentence. If there is a problem in recognizing the sentence, Subject2 signs for repeats. Otherwise subject1 proceeds with finger spelling the city-number pair. Subject2 recognizes and writes the pair recognized.

c) Subject2 signs the sentence and city-number pair and subject1 recognizes them and writes. The same procedure in step (b) is followed.

d) The condition is changed by changing the parameters and steps(a),(b) and (c) are followed. All the test conditions listed in the Table 7.1 are tested in the above manner.

The results of the above performed tests are discussed in the next section.

Condition Number	Subject1 Signs	Responses of Subject 2	Subject1 Signs	Responses of Subject 2
1.	I am going swimming tomorrow. London - 1123	Did you go swimming? London - 1123	I love playing darts. Boston - 2675	I love playing darts. Boston - 2475
2.	Please walk along the road. Paris - 4762	I enjoy walking winding roads. Paris - 4762	Is that a boy or girl? Chicago - 4029	Which one boy or girl that person? Chicago - 4029
3	I was walking, there was a lot of traffic. Tokyo - 5289	I was walking ina huge crowd of people. Tokyo 5289	That boy likes nut chocolate. Calcutta - 1532	That boy likes that peanut chocolate. Caultte - 1532
4.	Tomorrow I am going on the boat. New York - 7261	I went boating. New York - 7261	French red wine tastes sour. Vancouver - 7395	French red wine is bitter. Vancouver - 7395
5.	Can you come skating with me? Delhi - 9682	Can both of us go skating tomorrow? Delhi - 9682	Orange is sweet - lemon sour. Madras - 1989	Orange is sweet & lemon is sour. Madras - 1989
6.	Will you go fishing? Bombay - 3941	Will you go fishing? Bombay - ??	The soup was only warm. Moscow - 7575	Can not perform communication.
7.	He ran far and fast. Washington - 8236	The young boy was running. Washington - 8236	Men shave every day. Pune - 9289	Man shaves every day. Pue - 9289
8.	I don't know which way to go. Toronto - 4159	I don't know which way to go. Toronto - 4159	I heard a big noise. Bangalore - 6819	I hear a noise. Bangalor - 6819

Table 7.2 Test sentences and results.

7.2 Discussion of Results

Results of the tests are provided in column 3 and column 5 of the Table 7.2. The results provide some interesting findings. Some of the results are discussed below. For condition number1, 80×60 pels/frame over LAN, the sentence *"I am going swimming tomorrow"* was understood as *"Did you go swimming?"*. Facial expression is a major factor in deciding whether the sentence is a question or a statement. Since the spatial resolution of the images was 80×60 pels/frame which was too small to get the facial expressions clearly, the subject had misunderstood the statement sentence as a question sentence. For condition number2, 160×120 pels/frame over LAN, the sentence *"Is that a boy or girl?"* was understood as *"Which one boy or girl that person?"*. The context of the sentence was understood clearly though the words were different from the test sentence to the response. This result shows that 160×120 pels/frame images provided good spatial resolution to understand the facial expressions more clearly for sign language communication.

For condition number4 , 80×60 pels/frame over phone lines, the test sentence *"Tomorrow I am going on the boat"* was interpreted as *"I went boating"*. The tense of the response was wrong. Normally in sign language past, present or future tenses of the sentences are interpreted by the context of the sentence and an extended sign for representing the verb tense. In this case the word *"tomorrow"* is the key in deciding the tense of the sentence. The misinterpretation was the result of the subject missing the word *"tomorrow"* in the sentence. In condition number8, for 160×120 pels/frame over phone lines, the sentence *"I heard a big noise"* was understood as *"I hear a noise"*. In this case also

tense was misinterpreted. This misinterpretation was the result of sentence being short and also missing the extra sign for representing the tense. However, it can be said that in a conversing mode where day to day issues are communicated, the tense of the sentences can be understood clearly from the context of the conversation.

From the experiments, it also had been observed that, over Local Area Networks, the system worked at slightly faster rates than over telephone channels. But this speed increase was not significant. This is due to the overhead involved in the process of image capturing by the camera. For example on a system with 166 MHz Pentium processor, it takes 150 milliseconds just to capture a frame of size 160×120 pels/frame, which is significant.

It was also observed during the tests, that the dark background with normal lighting brought out features well. The enhanced environment with extra lighting did not help much in the testing for the subjects and the subjects felt uncomfortable with the lighting directed towards their faces. In one of the test conditions, the mode of the picture was changed from landscape, 160×120 pels/frame, to portrait mode, 120×160 pels/frame. Both the subjects preferred the landscape mode to the portrait mode, since landscape mode gave them lot of room to sign. This is a significant finding from the research which differs from the tests done by Pearson et al.[1985], in which portrait mode of images was used.

The communication was tested for different picture sizes over phone lines. Both the subjects felt and reported that the test condition with 80×60 pels/frame pictures at a rate of 8 to 9 frames per second was the best. They were able to communicate and finish the tasks with the test condition of 160×120 pels/frame pictures at a rate of 4 to 5 frames per second. They could not complete the tasks with the test condition of 320×240 pels/frame pictures at a frame rate of less than a frame per second, though one subject partially completed a task with many retries, which proved the necessity of good temporal resolution for effective sign language communication.

It took some time for the subjects to adjust to the space provided by camera view angle. Since some of the names of the cities were unfamiliar to them, for example Bangalore and Pune, they had to repeat twice to reconfirm the names of the cities. The subjects were able to get the context of the signs in all the environments with limited grammatical mistakes in the sentences, which can be observed from the results provided in the Table 7.2. Though the subjects were able to complete the tasks given to them, they felt that the system was slow compared to face to face communication. In the case of 80×60 pels/frame image size facial expressions were not very clear which gave the subjects some difficulty in recognizing the signs. However, the higher frame rate achieved at this size helped to complete the tasks faster than the environment with 160×120 pels/frame pictures. The subjects are currently using the teletype writing system for distance communication. They commented that this system would provide a more natural interface for communication than the teletype writing system. They also said that the test condition with 80×60 pels/frame pictures was the best for communication. They also suggested that it would be

better if the system could provide a spatial resolution of 160×120 pels/frame picture and the current temporal resolution of 80×60 pels/frame, which is 8 to 9 frames. The results of the subjective tests show that:

- Performance of the system over phone lines is almost similar to the performance on local area networks because of the significant overhead involved in the process of image capturing by the camera.
- Landscape mode of images is better than the portrait mode.
- Special lighting is not needed if a dark back ground is used.
- Good temporal resolution is very important for sign language communication.

Since the system provides good temporal resolution for small pictures of size 80 ×60 pels/frame the subjective test results prove that it can be used for deaf sign language communication.

7.3 Summary

This chapter explains the subjective tests performed on the system under different environments. The parameters changed in the environments are communication channel, picture size and lighting conditions. The results shows the effectiveness of the system for deaf sign language communication.

Chapter 8

Conclusion

Components of a system designed for deaf sign language communication have been presented. Deaf sign language requires a temporal resolution of 8 to 10 frames per second for effective communication. The system differs from the commercial video conferencing applications in the following ways.

- The system is designed specifically for deaf sign language communication with the aim of getting reasonable temporal resolution of small pictures.
- Texture and color information are traded to keep movement details of the image sequence. This is done by extracting only the main feature points of the objects and representing them as binary points. In essence the system moves cartoon or binary sketches of the objects over communication channels.

The following sections explain and list the major findings from the research and propose recommendations on future research directions.

8.1 Major findings

The main design of the system includes noise filtering of the image sequences from a low cost camera, feature extraction, irreversible preprocessing steps and compression. The major findings during the different phases of the system design are listed below.

- Noise filtering is performed to reduce the low frequency noise from the low cost cameras. The noise is found to be additive and multiplicative which produces a throbbing or pulsing effect on the image sequences. Many variations of temporal and spatio-temporal filtering methods are tried to reduce the noise. A method which uses histograms of successive images for segmenting moving and non-moving regions is found to be effective in reducing the noise without producing any dirty windows motion artifacts. This method is also found to be simple, taking less processing power which makes it useable in a real time application.
- A 3×3 valley detector is used for feature extraction. This extractor was used in a previous experiment to extract features from a television quality camera. It is found that the extractor is equally good in extracting features from the low quality cheap cameras. The extractor uses three thresholds, to be set by the user, for effective feature extraction. A study made on the thresholds and the binary sketches helped to develop new adaptive mechanisms to calculate the thresholds. The thresholds are calculated adaptively by using the global image statistics, which simplifies the user interface.
- Irreversible preprocessing techniques were studied to improve the subjective quality of the binary sketches and to improve compression efficiency. They are isolated point removal, threshold hysteresis and predictive filtering. It is found that the combina-

tions of isolated point removal and the new predictive filtering is very effective in achieving good results.

- A three dimensional compression algorithm is used for binary sketches. It uses the conditional replenishment scheme in the temporal stage and a variation of quadtree coding in the spatial stage. This algorithm is found to be simple, fast and provides reasonable compression efficiency for moving cartoons. It also provides limited error resilience to the data because of its block oriented nature.
- The system designed and developed is subjectively tested formally by deaf subjects. From the results of the subjective tests, it is found that the system can be used for deaf sign language communication. The subjective tests also confirm the fact that deaf sign language communication requires good temporal resolution of frames.

8.2 Recommendations

Considering the developments happening in low bit rate video communications and the present drawbacks of the system, the following recommendations can make the system a helpful tool for the deaf people.

- Though the current feature extraction algorithm is effective in extracting main features, it uses three thresholds. An operator adaptive to the global image characteristics with fewer or no thresholds would simplify the feature extraction process and the user interface.
- Efficient predictive video coding schemes may be used to extract the moving parts of the scene which form a low percentage of the total image detail of the complete scene. Moving areas may be segmented by using the color information and motion compen-

sation algorithms. Since the amount of information is less, the information can be transmitted over low bandwidth telephone channels.

- Resolution of the binary images can be improved by using valley information in the generated binary sketches. The valley information may be used as a foundation or primitive upon which shading or color information can be built. Since only the image primitives are transmitted, it will not bottleneck the low bandwidth telephone channels.
- Though reasonable compression is achieved by using the present compression algorithm, compression efficiency may be improved further by using variable length coding schemes like Huffman coding and Arithmetic coding as a final stage in the compression process. To reduce the processing time, the present algorithm may be modified in such a way as to build the variable length coding scheme into the basic compression algorithm.
- Telephone channels normally introduce channel errors to the data. The current system has no error correction facilities. If variable length coding is used as a final stage in the compression process, error resilient entropy coding schemes which work on variable length codes, may be considered. This will provide more resilience to the data for channel errors without affecting the compression efficiency.
- To improve the processing speeds, multi-threading features of the Windows operating system can be used. Time can be saved by parallel processing other image processes such as sketch extraction, compression and image capturing.
- The user interface of the system may be improved by taking feedback and suggestions from ordinary users and implementing them in the design. Chapter 6 provides a sec-

tion on the improvements to the user interface of the system from the designer's point of view.

References

George Sperling (1981), "Video transmission of American sign language and finger spelling: Present and projected bandwidth requirements", *IEEE Transactions on Communications*, vol.com-29, no-12, pp.1993-1002.

Don Pearson (1981), "Visual communication systems for the deaf", *IEEE Transactions on Communications*, vol.com-29, no-12, pp.1986-1992.

Davies G.M., Ellis H.D. and Shepherd J.W. (1978), "Face recognition accuracy as a function of mode of representation", *Journal of Applied Psychology*, 63, pp.183-187.

Beiderman I. (1987), "Recognition by components: A theory of human image understanding", *Psychological Review*, 94, pp.115-145.

Beiderman I. and Ju G. (1988), "Surface versus edge based determinants of visual recognition", *Cognitive Psychology*, 20, pp.38-64.

North Dakota State University web page (1998), *http://www.ndsu.nodak.edu*, North Dakota State University, Fargo, North Dakota, USA.

Possum Controls Limited web page (1998). *http://www.possum.co.uk*. Possum Controls Limited. UK.

Downton A.C. and Newell A.F. (1979). "An assessment of Palantype transcription as an aid for the deaf", *International Journal of Man-Machine Studies*. vol. 11. pp.667-680.

Tartter V.C. and Knowlton K. (1981). "Perception of sign language from an array of 27 moving spots", *Nature*, vol. 289, pp.676-678.

Wallis R.H. and Pratt W.K. (1981). "Video conferencing at 9600 baud". *IEEE International Conference on Communications*. Conference record. pp. 22.2.1-3.

Philippe Letellier, Morton Nadler and Jean-Francois Abramatic (1985). "The telesign project", *Proceedings of the IEEE*. vol. 73. no. 4. pp.813-827.

Don E. Pearson and John A. Robinson (1985). "Visual communication at very low data rates". *Proceedings of the IEEE*. vol. 73. no. 4. pp.795-812.

Jun Xu. Yoshinao Akoi and Zhong Zhang (1993). "A Method for Synthesizing Animation to Transmit Sign Language by Intelligent Communication", *Electronics and Communications in Japan*, Part 3, Vol. 76. no. 2. pp. 108 – 116.

Yoshinao Akoi. Shin Tanahashi and Jun Xu (1994). "Sign Language Image Processing for Intelligent Communication by a Communications Satellite", *Proceedings, IEEE Inter-*

national Conference on Acoustics, Speech and Signal Processing, Piscataway, NJ, USA, Vol. 5, pp. v.197 – v.200.

C-Phone Corporation web page (1998). *http://www.cphone.com*, C-Phone Corporation, NC, USA.

Marr D. and Hildereth E. (1980),"Theory of edge detection", *Proceedings of the Royal Society of London*, B-207, pp.187-217.

Nadler M. (1976),"Effective and cost-effective real time picture operators for medical imagery", *Decision Making and Medical Care*, Edited by F.T. Dombal and F.Gremy, Amsterdam, The Netherlands.

Elias Hanna and Vicki Bruce (1992), "Testing the efficacy of computer generated cartoons", *IEEE, Machine storage and recognition of faces colloquium organized by Professional Groups E4 (Image processing and vision) and E5(Signal processing)*, Savoy Place, London, UK, pp..

John A. Robinson (1986), "Low data rate visual communication using cartoons: A comparison of data compression techniques", *IEE Proceedings*, vol. 133, pt. F, no. 3, pp.236-256.

Huang T.S. and Hsu Y.P. (1981),"Image Sequence Enhancement", *Image Sequence Analysis (Springer Series in Information Sciences 5)*, Edited by T.S. Huang, Springer-Verlag Publications, New York, pp.289-309.

Dennis T.J. (1980),"Nonlinear temporal filter for television picture noise reduction", *IEE Proceedings*, vol. 127, pt. G, no. 2, pp.52-56.

Eric Dubois and Shaker Sabri (1984),"Noise reduction in image sequence using motion-compensated temporal filtering", *IEEE Transactions on Communications*, vol.com-32, no-7, pp.826-831.

Eric Dubois (1992), "Motion-Compensated Filtering of Time-Varying Images", *Multidimensional Systems and Signal Processing*, 3, Kluwer Academic Publishers, 1992.Boston, pp.211-239.

Katsaggelos A.K., Driessen J.N., Efstratiadis S.N. and Lagendijk R.L. (1989),"Spatio-temporal motion compensated noise filtering of image sequences", *Series: The international society for optical engineering (SPIE), Visual communications and Image Processing IV*, vol. 1199, pp.61-70.

Dimitrios S. Kalivas and Alexander A. Sawchuk (1990),"Motion compensated enhancement of noisy image sequences", *International Conference on Acoustics, Speech and Signal Processing , 1990*, Albuquerque, New Mexico, pp. 2121-2124.

Jain J.R. and Jain A.K. (1981),"Displacement measurement and its application in inter-frame image coding", *IEEE Transactions on Communications*, vol. COM-29, pp.1799-1808.

Robbins J.D. and Netravali A.N. (1983),"Recursive Motion Compensation: A Review", *Image Sequence Processing and Dynamic Scene Analysis (Series F: Computer and System Sciences No. 2)*, Edited by T.S.Huang, Springer-Verilag Publications, New York, pp.76-103.

Shaker Sabri (1983), "Movement Compensated Interframe Prediction for NTSC color TV signals", *Image Sequence Processing and Dynamic Scene Analysis (Series F: Computer and System Sciences No. 2)*, Edited by T.S.Huang, Springer-Verilag Publications, New York, pp.156-199.

Graham Thomas (1991),"Motion and Motion Estimation", *Image Processing*, Edited by Don E. Pearson, McGraw-Hill Book Company Limited, UK, pp.40-57.

Huffman D.A. (1952), "A method of construction of minimum redundancy codes", *Data Compression, Bench Mark Papers in Electrical Engineering and Computer Science*, volume 14, Edited by Lee D. Davisson and Robert M. Gray, Dowden, Hutchison and Ross, Inc.,1976, pp. 28-31.

Yamada T. (1979), "Edge difference coding - a new, efficient redundancy reduction technique for facsimile signals", *IEEE Transactions*, 27, pp. 1210-1217.

JBIG web page (1998), *http://www.disc.org.uk/public/jbighomepage.htm*, Joint Bi-Level Image experts Group.

Pixel Magic web page (1998), *http://www.pixelmagic.com*, Pixel Magic, USA.

Yao Wang and Jen-Ming Wu (1992)," Vector run length coding of bi-level images". *Proceedings Data Compression Conference, March 24-27, Snowbird, Utah*, Edited by James A. Storer and Martin Cohn, IEEE Computer Society Press, Los Alamitos, California, USA, pp.289-298.

Gray R.M. (1984)," Vector Quantisation", *IEEE ASSP magazine*, vol. 1, pp.4-29.

Haskell B.G. (1979), "Frame replenishment coding of television", *Image Transmission Techniques (Advances in Electronics and Electron Physics, Supplement 12)*, Edited by Pratt W.K., pp. 189-217, Academic Press, New York.

Cu-seeme homepage (1998), *http://cu-seeme.cornell.edu*, Cornell University, USA.

Li-Te Cheng and John Robinson (1998),"MCLGallery: A framework for Multimedia Communications Research", *Conference Proceedings, IEEE Canadian Conference on Electrical and Computer Engineering*, Waterloo, Canada, pp.413-417.

Li-Te Cheng (1998), *"MCLGallery - Curator's Guide"*.

Don E. Pearson, E. Hanna and K. Martinez (1990), "Computer-Generated Cartoons", *Images and Understanding*, Edited by Horace Barlow, Colin Blakemore and Miranda Weston-Smith, Cambridge University Press, Cambridge.

David Marr (1982), *Vision*, W.H. Freeman and Company, USA.

Clarke R.J. (1996), *Digital Compression of Still Images and Video*, Academic Press, USA.

Rafael C. Gonzalez and Richard E. Woods (1993), *Digital Image Processing*, Addison-Wesley Publishing Company, Inc.

Rhys Lewis (1990), *Practical Digital Image Processing*, Ellis Horwood Limited, England.

Simon Haykin (1994), *Communication Systems (Third Edition)*, John Wiley and Sons, Inc.

Ian Sommerville (1992), *Software Engineering (Fourth edition)*, Addison-Wesley Publishing Company, Inc.

Lon Barfield (1993), *The User Interface Concepts and Design*, Addison-Wesley Publishing Company, Inc.

William Stallings (1997). *Data and Computer Communications (Fifth edition)*. Prentice-Hall. Inc.

Steve Oualline (1995). *Practical C++ Programming*, O'Reilly and Associates. Inc.

William H. Press. Saul A. Teukolsky, William T. Vellerling and Brian P. Flannery (1992). *Numerical Recipes in C (Second Edition)*, Cambridge University Press.

(1997). *Power ++ Programmer's Guide*. Sybase. Inc.

Appendix A
Camera Specifications and Controls

The camera used for the testing and capturing of test sequences is the gray level low cost camera QuickCam from Connectix Corporation, USA. QuickCam uses a Charge Coupled Device to capture the image from the lens. The image is accumulated and then transmitted to the computer as a digital data stream via a parallel port. A Connectix-written driver then converts the bits into a moving image that Video for Windows can understand.

A.1 Specifications

The specifications of the camera are as follows:

- Up to 320×240 pels/frame image capture.
- 4-bit gray scale (16 shades of gray) for movies, 6-bit gray scale (64 shades of gray) for bitmaps and other still graphics.
- Frame rates up to 24 frames per second (fps). 10 fps for a window size of 160×120 pels/frame.
- Powered from keyboard port of a computer, draws less than 350 milliwatts of current.
- Field of view is approximately 65 degrees (equivalent to a 38mm lens on a 35mm camera).
- Focus is fixed from 18 inches to infinity.
- Lens is f1.9.

A.2 Controls

The software driver of the camera provides two windows for controlling the camera parameters. They are Image size and Quality (Figure A.1) and Camera Adjustments shown in (Figure A.2). Controls in both the windows are explained below.

A.2.1 Image Size and Quality

- *Image Size* - Image size is expressed in pixels; the example above shows that the image size will be 160×120 pels/frame. The image size can be set by clicking one of the preset buttons (1/4, 1/2, or Full), or by choosing another size from the drop-down menu.
- *Greys* - The 16 greys option captures the video more quickly, but 64 gray gives sharper image.

- *Zoom* - Zoom In shows a close-up of the image by magnifying the central portion of the image. Zoom Out restores the image to its previous size.

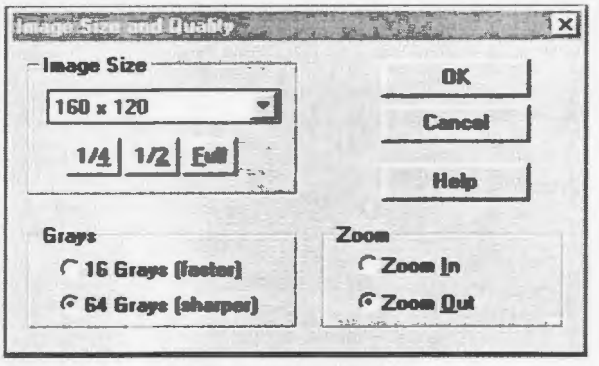


Figure A.1 Image size and quality controls.

- *Help* button provides more help about the options on the dialog box.
- *OK* button is clicked after selecting the options.

A.2.2 Camera Adjustments

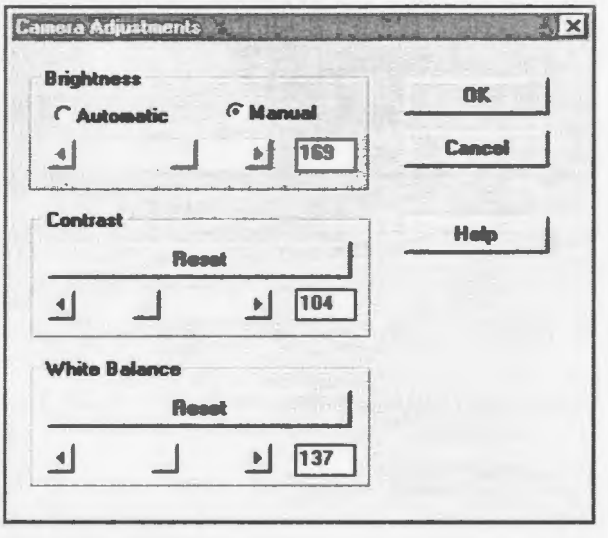


Figure A.2 Camera adjustment controls.

- *Brightness* - The brightness setting on the QuickCam is like shutter speed on a camera, staying open longer to let in more light. It can be set *manual* or *automatic*. The higher the setting, the more light that reaches the camera. But if the setting are too high, the camera "stays open" for so long that moving objects will be blurred in the video. This setting has to be experimented with to find the best one under different lighting conditions.
- *Contrast* - Sets the image to have shades of gray or black and white. *Reset* button is clicked to set the contrast back to the QuickCam's default setting.

- *White Balance* - This setting adjusts the amount of white to black for the camera and surrounding lighting. If the image looks washed out or consists solely of whites and light grays, the white balance slider is moved to the left to lower the amount of light reaching the camera. If the picture looks dark or consists solely of dark greys and blacks, the slider is moved to the right to let more light in. The white balance is to be set correctly when the picture covers the spectrum fully from black to white with a wide range of grays in between. To let the camera decide the best setting, *Reset* button is clicked for White Balance.
- *OK* button is pressed after choosing the options.

Appendix B
Classes used in the system design

The classes used in the design of the deaf sign language communication system can be divided into two types; classes from MCLGallery and *Specific Operation Classes* designed to do specific operations such as compression and noise filtering. For MCLGallery classes, only the purpose of the classes is explained here. Detailed explanation of all the functions of MCLGallery classes are provided in the MCLGallery Curators Guide. For specific operation classes all the functions are listed.

B.1 MCLGallery classes

The MCLGallery classes used in the system design are:

(1) *MCLVideo class* - allows the capture of video frames from a video device such as a video camera. The images captured by this class can then be sent to MCLBitmap structure for actual processing, or saved to a standard Windows 95 .AVI format file.

(2) *MCLBitmap class* - allows the presentation and manipulation of video information. Designed to load and save Windows 95 .BMP format images. This class interfaces with MCLVideo to capture image frames from a camera.

(3) *MCLData class* - is the key fundamental class behind most of the MCLGallery classes. It presents data as a byte buffer, and has functions that support byte manipulation. It is used internally by most MCLGallery classes, but can also be used directly for byte manipulation tasks such as compression.

(4) *MCLNet class* - allows to connect and host network sessions with other MCLGallery applications. It sets up the incoming message callback. Supported network services include TCP/IP (Transmission Control Protocol / Internet Protocol), modem and IPX (Internet Packet Exchange).

B.2 Specific Operation Classes

The classes designed to do specific operations are

(1) Form class
(2) NoiseFilter
(3) Encoder
(4) Decoder

These classes are explained in the following sections. The methods of the classes are also listed.

B.2.1 Form class

The Form class the main class which controls the operations of all the classes . This class has many functions which are related to the components used in the main program window. Since each component has a variety of functions depending on the events generated by the operations performed on the components, providing all the functions will overload this document. Hence only important functions are listed in this section.

This class uses both MCLGallery objects and objects of specific operation classes. It captures video from the camera, reduces throbbing noise from the frames using a filter, and extracts binary sketches. The binary sketches are processed using irreversible techniques such as isolated point removal. The binary frames are then compressed and sent over the network. When data is received from the network the callback function is called which does the decoding operation. Some of the related functions of this class are listed below.

Private Methods

WBool FilterSketch(MCLBitmap & curbitmap, MCLBitmap & refbitmap, MCLBitmap & outbitmap)

This function does both predictive filtering and feature point extraction. For predictive filtering it uses the current frame and a reference frame. The algorithm of predictive filter is provided in Chapter 5. The extracted sketches are stored in a separate bitmap. Returns TRUE on success and FALSE on failure.

WBool CalculateT3(MCLBitmap & bitmap)

Adaptively calculates value of one of the thresholds used in the sketch extraction algorithm. Returns TRUE on success and FALSE on failure.

WBool IsolatedPointRemove(MCLBitmap & bitmap)

This function removes noise like isolated points in the bitmap. Returns TRUE on success and FALSE on failure.

WBool Compress( )

This function calls the encoder object and passes to the object a bitmap, which is defined globally for compression. Returns TRUE on success and FALSE on failure.

WBool Transmit(MCLData & data)

This function transmits the compressed data on the network. Returns TRUE on success and FALSE on failure.

WBool ChangeSize(int width, int height)

This function changes the size of the globally defined bitmap to the sizes passed as parameters. Returns TRUE on success and FALSE on failure.

*WBool rotate(MCLBitmap & original, MCLBitmap & rotated, HWND wnd, MCLBitmap_component * component)*

Rotates the original bitmap by 90° right and stores it in the rotated bitmap. Returns TRUE on success and FALSE on failure.

*int mclnet_callback(MCLData * data , void * extra)*

This is the callback function called when data is received from the network. Extra pointer can be set to any desired data. Returns 1 on success and 0 on failure

B.2.2 NoiseFilter Class

An object of this class is instantiated in the form class. This class filters the noisy frames. It is a recursive filter and uses information of the current frame and the previously filtered frame. Detailed explanation of filtering algorithm is provided in Chapter 4. Public and private methods of the class are listed below.

Public methods

NoiseFilter( )

Default constructor. Initializes NoiseFilter object.

~NoiseFilter( )

Default destructor. Does the cleaning up operations for the decoder object.

WBool Filter(MCLBitmap & CurBitmap, MCLBitmap & PreBitmap, int Threshold, int Multiplier)

This function calls the other private functions for filtering the current frame and uses the information provided by the previous frame. Returns TRUE on success and FALSE on failure.

Private methods

WBool Initialise(MCLBitmap & Bitmap, int thresh, int mul)

Initializes global data members. Returns TRUE on success and FALSE on failure.

WBool FillArray(MCLBitmap & Cur , MCLBitmap & Pre)

This function takes data from the information provided by the current and the previous frames and puts it into the array structures defined globally. The data in the arrays is used for further processing. Returns TRUE on success and FALSE on failure.

WBool GetModAvgArray( )

Fills data in a special array from the information gathered in the previous processes. Returns TRUE on success and FALSE on failure.

WBool CheckModAvgArray( )

Checks a special array for missing elements and interpolates the missing elements. Returns TRUE on success and FALSE on failure.

WBool Interpolate(int index1, int index2)

Linearly interpolates data between two points supplied as indices of an array. Returns TRUE on success and FALSE on failure.

B.2.3 Encoder Class

An object of this class is instantiated in the form class and used for compressing binary frames. It takes the current frame and the previous frame and uses the temporal information thus provided to compress the current binary frame. Detailed explanation of the

compression algorithm is provided in Chapter 5. Public and private methods of this class are listed below.

Public methods

encoder( )

Default constructor. Initializes encoder object.

~encoder( )

Default destructor. Does the cleaning up operations for the encoder object.

*MCLData * encode(MCLBitmap & Current, MCLBitmap & Previous, int Threshold, int block_size)*

This function calls the other private functions for compressing current bitmap into an MCLData object. Returns pointer to the data object on success and NULL pointer on failure.

Private methods

void InitialiseParameters(MCLBitmap & cur, MCLBitmap & pre, int thresh ,int size)

Initializes global data members for further processes.

void CodeCompleteBlocks(MCLBitmap & Cur, MCLBitmap & Pre)

Divides the current frame into blocks and marks the blocks into changed and unchanged blocks with reference to the previous frame and codes the changed blocks only.

*void CopyBlockData(BYTE * ptr , int BlockWidth , int BlockHeight)*

This function is called for coding the changed blocks.

*int enquad(BYTE * ref_ptr, int size)*

This function does quadtree encoding of a block. The position of the block in the form of a pointer and size of the block are supplied as parameters. Returns 1 on success and 0 on failure.

WBool PackToMdata( )

This function packs the encoded bits into an MCLData structure defined globally. Returns TRUE on success and FALSE on failure.

B.2.4 Decoder class

An object of this class is instantiated in the form class. Functions of this object are called in the network callback function. The MCLdata structure received from the network is decoded into an MCLBitmap structure by this object which is then displayed on the remote window. The public and private functions provided by this class are listed below.

Public methods

decoder( )

Default constructor. Initializes the decoder object.

~decoder( )

Default destructor. Does the cleaning up operations for the decoder object.

WBool decode(MCLData Data, MCLBitmap & bitmap)

This function call the other private functions for decoding MCLData object into a MCLBitmap object. Returns TRUE on success and FALSE on failure.

Private methods

void InitialiseParameters(MCLBitmap & bitmap)

Initializes global parameters used in the class.

WBool DecodeCompleteBlocks(MCLBitmap & bitmap)

Decoding operation is done on a block by block basis and as blocks are decoded they are copied to the bitmap. Returns TRUE on success and FALSE on failure.

*int dequad(BYTE * ref_ptr, int size)*

Used for quad tree decoding. The initial position of the block in the form of a pointer and size of the block are provided as parameters. Returns 1 on success and 0 on failure.

Appendix C
Flow Control and Timing Optimization

Flow control is a technique for assuring that a transmitting station or computer does not overwhelm a receiving station or computer with data. Following sections explain the flow control problem when two machines of different speeds communicate with each other and also present a solution to the problem.

Deaf sign language communication system is a multithreaded program. A thread is the fundamental item to which the operating system allocates execution time on the processor. In a simple program, there is a single line or thread of execution. When one routine calls another, the first routine waits for the second to finish before resuming its own execution. This kind of program is called single threaded; execution proceeds in a linear fashion, with only one routine executing at a time.

A multithreaded program has more than one line of execution running concurrently in single process. For example, one routine may call another, then immediately resume its own execution, without waiting for the called routine to finish. In other words, the calling routine starts a new thread of execution which runs concurrently with the first thread. MCLNet uses a separate thread to handle transmission of data over network. To be specific, when the program calls the sending function of the MCLNet object, it provides the data structure to be transmitted to the sending function and resumes the next process without waiting for the sending thread to finish. Following section explains the methods tried to optimize the transmission time and identifies an optimum method which effectively utilizes the multithreaded property of the system.

C.1 Flow control

Figure C.1 shows the process diagram in a machine where module A is responsible for binary cartoon generation and module B is the callback function module. Considering Figure C.1(a), when there is communication between a slow machine and a fast machine, the faster machine processes data faster and sends it over the network. Assuming there is less transmission delay over the network, the slow machine will always be busy processing the received data, sending very few frames occasionally. This affects smooth transmission of frames. In other words, it reduces temporal resolution of the frames received at the faster machine which is not suitable for deaf sign language communication.

To achieve flow control between machines of different speeds, the processes are rearranged to the configuration shown in Figure C.1(b). The callback function now controls entire flow of the program. It decompresses the data when it receives the data from the network and generates binary frames and sends the data to the network. Effectively, a machine transmits a frame only when it receives a frame which make the communication half duplex. In this way, when there is communication between machines of different speeds, the faster machine synchronizes with the speed of the slower machine, enabling smooth transmission of binary frames.

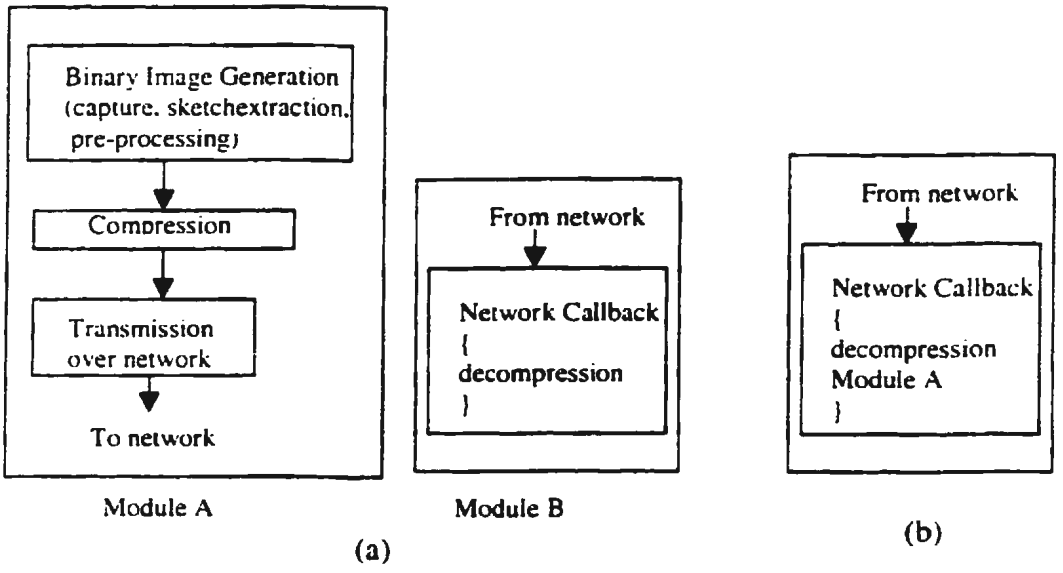


Figure C.1(a) Process diagram in a machine. (b) Changed process diagram.

C.2 Timing optimization

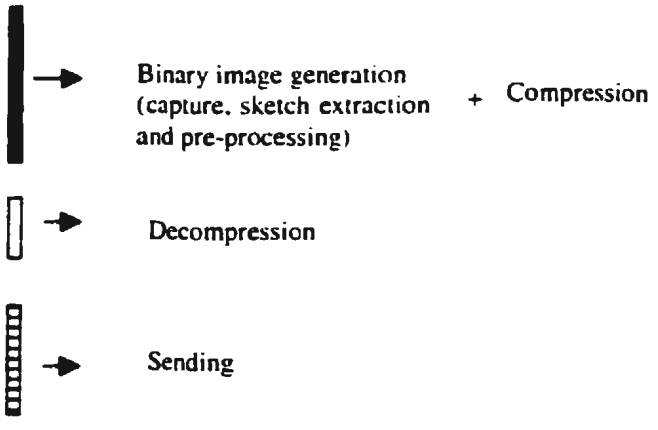


Figure C.2 Time taken for each processes.

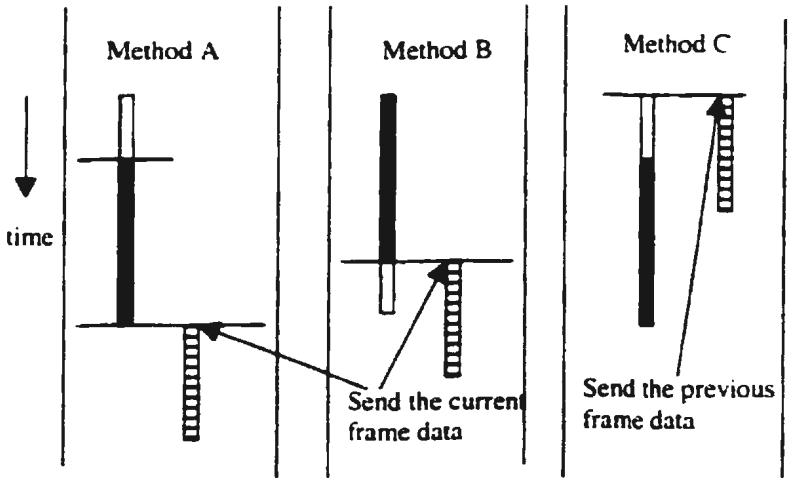


Figure C.3 Timing diagram for timing optimizing methods.

Figure C.2 shows the timing slices for binary frame generation, decompression and the sending thread. Figure C.3 shows the program flow control implemented in the network callback function for three methods. In Method A when a frame is received from the network, it is decoded and the next frame is processed and sent over network. In this method there is no overlapping of the processes which increases the time for sending a frame. In Method B when a frame is received, next frame is processed and sent over the network. While the data is travelling over the network the received frame is decoded. As seen from the Figure C.2 decompression takes less time and hence not much time is gained from this method although this method is better than Method A. In Method C when a frame is received, previously processed frame is transmitted over the network. While this data is travelling over the network, the received frame is decompressed and next frame is processed. As shown in Figure C.3 more time overlapping of processes occurs with this method which makes this method an optimal one and is implemented in the system.