UNIVERSITY OF ALBERTA

Variable Resolution Image Compression

By

Allan Sullivan

A thesis
submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree
of Master of Science

Department of Computing Science

Edmonton, Alberta
Fall 1993

ISBN 0-315-88215-8

UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR: Allan Sullivan

TITLE OF THESIS: Variable Resolution Image Compression

DEGREE: Master of Science

YEAR THIS DEGREE GRANTED: 1993

(Signed) ______________________

Permanent Address:
109 Janice Street,
North Bay, Ontario,
Canada

Date: ______________________

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled *Variable Resolution Image Compression* submitted by *Allan Sullivan* in partial fulfillment of the requirements for the degree of Master of Science.

Supervisor: A. Basu

Examiner: X. Li (Computing Science)

External: W. Grover (Electrical Engineering)

Date: 13 05 96

For friends and for family

Abstract

The large number of bytes required to digitally encode picture images, combined with the limited resources available for storage and transmission, create the need for image compression. Compression methods are very diverse, ranging from entropy encoding techniques to Discrete Cosine Transformation based techniques for still and motion pictures.

This thesis examines an image compression technique built on a variable resolution model. Within a picture, there often exists regions which an observer considers more significant than the remainder. Images can thus be viewed as having a central fovea surrounded by a periphery. Spatial sampling of pixels in the periphery is used to reduce the number of bytes representing the image. In this way, high quality around the region of interest is maintained. The performance of variable resolution is compared to other compression techniques in terms of compression ratio, image quality, and time. A videoconferencing system which uses variable resolution as its compression engine is also described.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

One of the most remarkable technological advances has been in the ability to digitally display, store and manipulate images. Today, it is not uncommon to have workstations capable of displaying over 1 million pixels, with each pixel's colour being defined by 24 bits. Another advancement has been the increasing bandwidth capacity on computer networks. New technology (such as fiber optics) makes gigabit per second networks possible. ISDN (Integrated Services Digital Network), offering combined voice and data on a single carrier, will eventually become commonplace. These technological advances have created a new concept: Multimedia. Sound, images, and text can be manipulated, exchanged between users, and stored for further use.

One major problem when handling digital images is the amount of resources needed for storage and transmission. For example, a 'typical' image with dimensions of 640*480 pixels and 24 bits of colour will require almost 1 megabyte of storage space. Transmission of this same image over a link with 9600 bps (a reasonable speed for widely-available commercial modems) would take over 96 seconds. Clearly, some method of data compression is needed in order to reduce the number of bytes used to represent the image to a more easily managed level.

Data compression schemes have existed since the early days of computing science. However, early schemes were aimed mainly at the compression of text and/or programs [20]. Technology eventually progressed to the stage where elec tronic manipulation of images is possible. This progression has been accompanied by research into the special problems involved with image compression.

The range of uses for image compression schemes is vast. Methods used to compress a single, still image can be used in the publishing industry or in any image archival system. The compression of motion pictures can be used in video-conferencing (videophones), electronic storage of movies, or even television transmission.

1.1 Lossy vs. Lossless Compression

Compression methods can broadly be classified as either lossless or lossy, depending on the method's effect on the data being compressed. In a lossless compression method, data is unaffected by compression. That is, the data after compression is identical to the data before compression. (The data may be stored in a different form, but exact pixel values remain unchanged.)

In contrast, a lossy method alters the data during the compression. Generally, lossy compression methods produce better **compression ratios** (reduction in the number of bytes required to represent data) than lossless methods; however they cannot be used on all forms of data, especially where errors would become obvious. (An example of this would be English text.) They can be used for other forms of data, such as digitized sound or images. Where lossy compression is acceptable, changes to the data are supposed to be imperceptible to humans. Such schemes make use of the limited abilities of the human senses. (For example, adjacent pixels with almost identical colours may be blended together with no visible defect.)

1.2 Variable Resolution Compression

In many images, there often exists an area which is of greater interest than the remainder of the picture. In this area (known as the fovea) more detail is required. The outer regions (known as the periphery) are often of secondary importance, and thus less detail is required. This decrease in detail within the periphery may be obtained by performing spatial sampling. Thus, areas near the fovea will have many points sampled and will be almost identical to the original image. Regions closer to the edge (and presumably of less interest to the observer) will appear distorted, since they will be represented using fewer sample points.

The concept of variable resolution has been applied to many different areas, including stereo matching and character thinning [17], but its application to image compression is new.

A number of distinct advantages exist with variable resolution compression:

- By controlling the sampling, the exact size of the compressed image is guaranteed
- The VR transform can be achieved using a table lookup; thus VR compression can be performed quickly
- Images compressed using the VR transform can be further compressed using other methods

VR cannot be used to compress all data, since in many cases (such as commercial television broadcasts) the distortions caused would be unacceptable. However, there are many applications which would benefit from having a high resolution fovea. Some examples include: videoconferencing (fovea on subject's face) sports events (fovea on the flow of the game), and video lectures (fovea on the speaker and/or teaching material).

1.3 Objective of Thesis

This thesis describes the work performed in developing an image compression system based on variable resolution. Distortions caused by the VR process are examined and compared to other compression methods. In addition, a videoconferencing system based on VR is described. The work should provide an indication of the viability of the VR transform for image compression.

1.4 Description of Thesis

Chapter 2 discusses previous work on data compression, including some state-of-the-art compression standards for image compression. Chapter 3 describes the mathematical foundation of the VR transform. Chapter 4 describes the criteria by which the performance of a compression algorithm can be measured. Chapter 5 contains an examination of the performance of VR compression (including compression ratios and distortion) and compares the results with other compression methods. Chapter 6 describes a video-conferencing system designed using VR as the compression method. Conclusions and suggestions for further research are discussed in Chapter 7.

Chapter 2

Background

The roots of data compression go back to the 1940s, when much of the theory was developed by Claude Shannon at Bell Labs. Compression is related to Information Theory, and its origins precede the widespread use of computers [20].

Data compression depends on developing a model for the data to be compressed, and using some method for removing redundancy based on the model. When compression routines were implemented on the earliest computers, processing time was at a premium. Thus, when compressing data, encoding routines relied on previously established models. The best compression was achieved by applying the best model to the data in question. (For example, a model may be based on the frequency of letters in English text. When compressing English text, this model may be adequate; however, it may be poor for compressing program files.)

Advances in technology have increased the processing speed of computers to the point where it is possible to develop a unique data model for each item to be encoded. The extra speed also allows more complex algorithms to be used. However, these same advances have also given computers the ability to manipulate sound and graphics; these forms of data have unique characteristics, and so new data models must be developed to handle them.

Many forms of compression exist today, for general data, file archiving, still images and motion images. Often, compression is built directly into hardware.

Standardization is often a problem however. Many implementations are particular to a company or platform, so that quite often commercial products do not interact with each other. One of the most recent examples of this is the commercialization of 'videophone' products, many of which are incompatible with each other. In addition to the general problems of standardization is the increase in the number of software patents, which effectively restrict the use of certain algorithms.

2.1 Entropy Encoding

In information theory, the term **entropy** refers to the information content of data. This value can be calculated using the following formula:

$$Numberofbits = -\sum_i \log_2(p_i) \tag{2.1}$$

where p_i represents the probability (frequency) of value i. This value represents the absolute minimum number of bits required to store a message. Often, the number of bits used to represent data is much greater than what entropy would dictate. However, by using special codes to represent bytes or groups of bytes, the total number of bits required to represent a message can be decreased substantially. This is known as **entropy encoding**, and by this method we can approach (but never get below) the entropy content of the data. The effectiveness of entropy encoding depends very much on patterns within the data and/or having a skewed distribution of values.

Entropy encoding forms the basis for many general data compression programs (such as program archivers and general compression tools). It can be used for image compression as well. However, entropy encoding by itself does not compress images well, since pixel values tend to be uniformly distributed and lacking in patterns (especially in the least significant bits). Instead, entropy encoding methods are often combined with various image transforms to improve compression ratios.

2.1.1 Minimum Redundancy Encoding

Within any set of data, certain bytes occur more frequently than others. The letter 'e' may occur more frequently than 'x' in a certain text file, while one colour may occur more frequently than another within an image file. By developing a code in which a byte is represented by a variable number of bits (more frequent bytes being represented by shorter codes), the total number of bits required to represent the data can be reduced. (Note that to make decoding possible, no code may be a prefix of another code.)

The first well-known method of building these codes was developed almost simultaneously by C. Shannon and R. M. Fano. Developing codes using Shannon-Fano encoding depends on developing tables of byte value frequencies. The table is repeatedly subdivided into portions where the sum of the frequencies is approximately equal. every subdivision will add one bit onto the code for a byte in the table. If the frequency of bytes is greatly skewed, table divisions will be irregular. Shorter sub-tables (containing the more frequent bytes) will reach a point where they cannot be subdivided further; thus their codes will be shorter than the longer subtables.

Shannon-Fano encoding was quickly followed by a method developed by D. A. Huffman in the early 50s. Huffman encoding typically involves building binary trees, where each byte is represented by a leaf in the tree. Less frequent bytes appear further from the root. The number of bits used for the code depends on the leaf's distance from the root.

Huffman coding obtains better compression ratios however, so it has become predominate. The ability of minimum redundancy encoding to compress data is greatly increased if preceding bytes are considered in the calculation of frequencies.

2.1.2 Dictionary Compression

Dictionary compression methods owe much of their existence to the work of J. Ziv and A. Lempel in the late 70s. Their first technique (outlined in 1977) is known as LZ77, and is a 'sliding window' technique. In this form of compression, a fixed-size window is slid over the data, and bytes are compared to strings in a buffer of previously scanned data. If a string of bytes within the window matches a string within the buffer, it is replaced by a pointer into the buffer.

A second technique outlined in 1978 is known as LZ78, and follows a different approach. In LZ78 a table of strings is stored in a symbol table. As bytes within the file are read in they are added to a special string, and the table is scanned to see if the string has been previously encountered. If it has, then additional bytes are added. However, if the string is unique, a code representing the closest possible match is output, and the new string is added to the table.

A number of improvements have been made to the LZ algorithms. Most of the improvements of the LZ77 algorithm involve finding faster ways to search through the buffer for previously encountered strings. Some improvements have been made to the LZ78 algorithm as well. Most notable were changes made by T. Welch in the 1980s, creating the LZW algorithm [28]. A major difference between LZ78 and LZW encoding was the preloading of the symbol table with single byte values. This means that any symbol can be encoded, even if it has not been previously encountered.

The LZ algorithms and their variants have been used in many compression programs. Most notable has been the use of LZW in some graphical image formats, and in the Unix Compress program.

2.2 Still Image Compression

Images can be compressed using any of the entropy encoding schemes mentioned above. Many graphic image formats have entropy encoding built in. For example, the Graphic Interchange Format, developed by Compuserve, is an 8 bit/pixel lossless image format which uses a variant of LZW dictionary compression [7]. TIFF files (Tagged Interchange File Format) may use several compression schemes, including Huffman and LZW [18].

However, pixels within images are usually uniformly distributed and do not follow any patterns. Hence, entropy encoding schemes usually have limited success in compressing images. Therefore, other techniques are used in addition to entropy encoding.

2.2.1 JPEG Compression

The Joint Photographic Expert Group (JPEG) was a collaboration between the CCITT and the International Standards Organization (ISO) [3, 27]. The goal of the JPEG group was to produce a standard for general image compression, allowing several methods. Among the mandates of the JPEG group, two standards appear to be the most important: base-line sequential lossy and lossless encoding.

The JPEG lossless standard compresses images by estimating the value of a pixel from its surrounding neighbors. The difference between the pixel and its estimate are stored using Huffman encoding. If the estimation is good, most differences will be small, skewing the distribution of bytes towards 0. Such a skewed distribution will compress well using entropy encoding.

More attention has been given to the JPEG lossy compression scheme. The heart of the JPEG compression scheme is the two dimensional Discrete Cosine Transformation. The DCT is related to Fourier transforms, and transforms images from the spatial domain to the frequency domain (since every signal can be built using a sum of frequencies). The DCT is applied to 8*8 blocks within the image; thus, in the frequency domain, higher frequencies are represented by values in the lower right corner of the transformed block.

The DCT used in JPEG is given as follows:

$$F(u,v) = \frac{1}{4}C(u)C(v)\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y) cos\frac{(2x+1)u\pi}{16} cos\frac{(2y+1)v\pi}{16} \tag{2.2}$$

where $C(u), C(v) = \frac{1}{\sqrt{2}}$ for u,v=0, $C(u), C(v) = 1$ otherwise.

The DCT does not provide any compression by itself; it is simply a lossless (and reversible) transformation. However, because the human eye is less sensitive to higher frequencies quantization of the frequency data can be performed, with high frequency data (i.e. values closer to the lower right of the 8*8 block) given

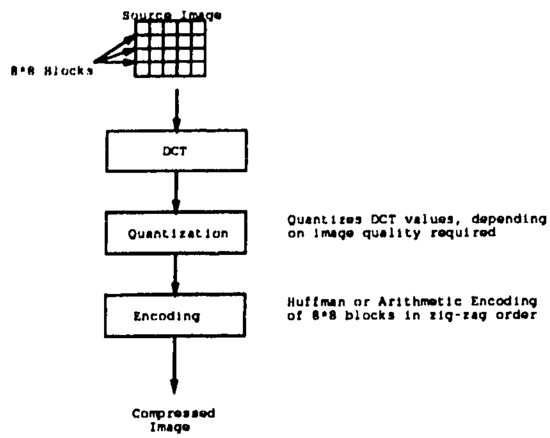

Figure 2.1: The JPEG Baseline sequential lossy compression scheme.

fewer quanta than lower frequencies. This quantized data is then compressed using Huffman encoding (or any other form of entropy encoding). Since there will be fewer quantization levels in the high frequency components, data will be skewed.

A number of additional 'tricks' are used within the JPEG compression scheme to improve performance. For example, the lowest frequencies of the blocks are coded with respect to the previous block. If the content of adjacent blocks is similar, the differences in the low frequencies will be small. When compressing colour images, pixel values are represented using a luminosity-chrominance colour space. Since the human eye is more sensitive to pixel brightness than colour, subsampling of the chrominance bands can be applied. During the encoding step, values are encoded in a zig-zag order, approximating increasing frequency.

One important advantage to JPEG compression is the control over the compression: If high-quality images are desired, using more quantization levels at the higher frequencies will give a better quality image (although with less compression). Using fewer quantization levels will result in a much more compressed image, although with more distortion. (At extremely high compression, the image will appear 'blocky'. This is an artifact caused by the use of 8*8 blocks with the DCT.)

2.2.2 Fractal Compression

Fractal compression attempts to represent images by developing models based on fractal equations [13]. The display of such images may be performed relatively easily, however, compression is quite difficult. Early attempts at fractal compression actually depended on human input in developing the fractal equations.

JPEG compression has had an earlier start over fractal compression in the race to become the dominant compression standard. If fractal compression is to become widely used, it will have to compete against JPEG.

2.3 Motion Image Compression

Many current standards for video are analog in nature (e.g. NTSC, PAL, SECAM). These standards are acceptable for what they were designed for. However, they are unsuitable for digital manipulation (such as in computerized videoconferencing).

A motion picture can be considered as a stream of stand-alone frames. (In fact, there are plans to use compressed JPEG images for video mail [6].) However, there is often a large degree of temporal redundancy between adjacent frames. Motion picture compression schemes make use of similarities between frames in order to improve compression.

2.3.1 p*64 Motion Picture Compression

The CCITT H.261 standard (also called p*64) was designed for the compression of video information over telephone links [3, 19]. The p*64 nickname refers to the fact that bandwidth for such links is offered in 64kb/s bandwidth increments as offered by phone companies; more channel bandwidth gives better quality video.

As with JPEG, p*64 uses frames compressed with DCT-based encoding. However, it also uses a predictive coding method, where blocks in the current frame are prediced from blocks in the previous frame. (This predictive encoding provides a

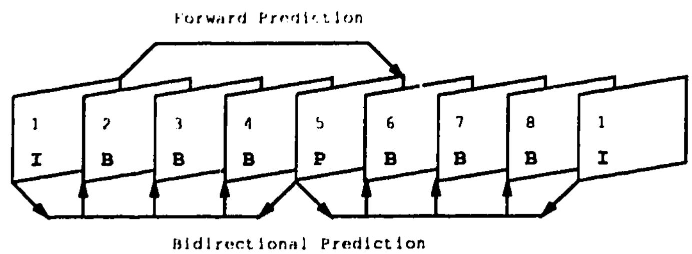

Figure 2.2: Layout of frames possible with the MPEG motion picture encoding scheme.

higher compression ratio, since it makes use of redundancies between frames.)

2.3.2 MPEG Compression

The Motion Picture Expert Group was given the task of designing a standard for compressed digital video [9, 14, 15] of a general nature, allowing full motion. The MPEG standard which was developed consists of three types of frames:

- Intrapictures (I) - stand-alone frames, compressed using DCT based methods. These frames allow random access.
- Predictive frame (P) - coded with respect to the previous I or P frame.
- Interpolated / Bidirectional frame (B) - Coded with reference to the previous I or P frame. Interpolation involves motion compensation based on blocks of 16*16 pixels. These are called 'bidirectional' because they are designed to allow reverse viewing of video frames.

The ratio of these frames can be altered, depending on the bandwidth and quality requirements. The initial standard was aimed at bit rates of 1.5 Mb/s. The next phase (MPEG II) is a standard for higher quality signals up to 10 Mb/s.

2.3.3 HDTV

Current standards for broadcast television are relatively primitive in terms of image quality. High Definition Television, supporting much higher resolution than current standards, is then next step [15]. However, the higher resolutions available under HDTV cannot be supported by the radio spectrum allocated by the government without some form of compression.

A number of standards for HDTV have been proposed. (In fact, too many to mention here.) Some of them are analog, although most are digitally based (including one MPEG-based proposal). Politics may prevent a standard from being adopted world-wide. (Japan and Europe are closer to defining standards than is North America.)

Chapter 3

The VR Transform

Studies have shown that animal visual systems have higher resolution in the center of the visual field than in the periphery. Swartz developed several mathematical models of vision systems based on this variable resolution concept [22].

Although computers cannot approach the ability of biological visual systems, such biological systems can serve as a model for transforming images when the scene contains regions of varying importance. This transformation can be performed using special camera lenses (i.e. 'fish eye' lenses). Another solution is to use the computer to vary the spatial resolution of images in the image periphery according to the mathematical models developed by Swartz. Results presented here used a simplified mathematical model to implement the VR transform.

3.1 The Basic Variable Resolution Model

The VR transform has two main factors which affect the resultant image: the expected size (compression), and alpha (α) which controls the distortion at the edges of the image with respect to the fovea. A high α value gives a sharply defined fovea with a poorly defined periphery; a small α value makes the fovea and periphery closer in resolution.

Under the VR transform, a pixel with polar coordinates (r, θ) is mapped to

(a)

(b)

Figure 3.1: A sample image (a), and image under the VR transform with the fovea at the center (b). Note the unused pixels in the corners of the transformed image.

(vr, θ) where

$$vr = \ln(r * \alpha + 1) * sf \tag{3.1}$$

In other words, the pixel is moved from r to vr units away from the fovea. This transformation is easily reversed, allowing r to be defined in terms of vr. The value sf is a scaling factor, and is calculated so that points at the maximum distance from the fovea in the original image are at the maximum possible distance in the VR image:

$$sf = \frac{vr_{max}}{\ln(r_{max} * \alpha + 1)} \tag{3.2}$$

The values of vr_{max} and r_{max} are dependent on the distance from the fovea to the edge in the compressed and original image respectively. Past work has involved the use of a single scaling factor for the entire image. Figure 3.1 shows an image under the VR transformation.

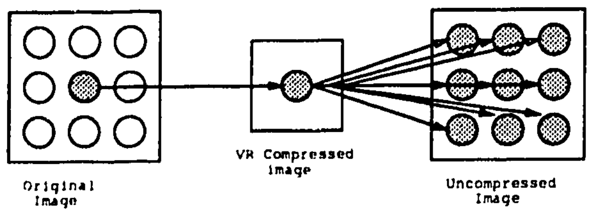

Figure 3.2: Sampling of pixels during compression and expansion of a VR image.

It must be noted that pixels are discrete elements. Thus, when reducing the size of the image via the VR transform, one VR pixel will represent several pixels in the original image. (See figure 3.2.)

3.2 Extended Model with Movable Fovea

There are a number of problems with the basic variable resolution model and its application to image compression. Under the VR transform images are not rectangular. If storage in a rectangular field is necessary, areas of the image must be clipped off, or the image will have unused pixels in the corners. The problem is magnified when the fovea is not located in the center of an image.

A number of attempts were made to solve this problem by altering the mapping equation from a logarithm to a sigmoid function. Although these attempts made better use of the image space than the basic model, they still suffered from the same problems.

The most successful method (and the method used in all subsequent work) was to use multiple scaling factors, each scaling factor dependent on the angle θ in the polar coordinates. Now, each angle θ has its own maximum distances to the image edge, and thus its own scaling factor. The transformed image can now be mapped to a rectangle with full space utilization, regardless of the location of the fovea. (Figure 3.3 shows an image compressed using the extended model.)

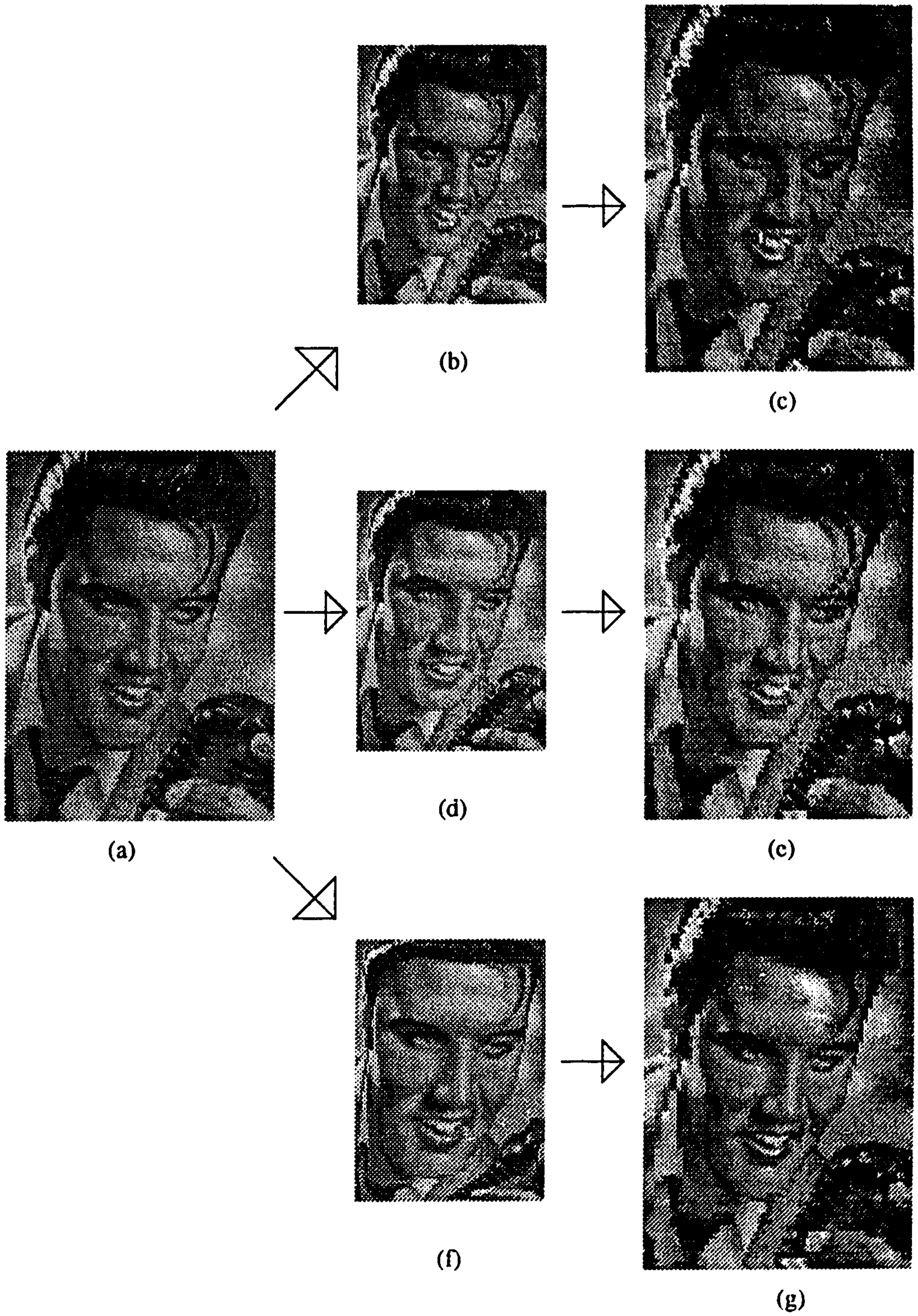

Figure 3.3: Sample image compressed and expanded using various α values and 50% compression. (a) Original image, (b) and (c) using $\alpha = 0.001$, (d) and (e) $\alpha = 0.01$, (f) and (g) $\alpha = 0.1$. Fovea is at center.

(a)

(b)

Figure 3.4: Sample image compressed and expanded using two foveas located at image center and in lower right corner. 50% compression, $\alpha = 0.1$. (Compare with figure 3.3.)

3.3 Multiple Foveas

There may be circumstances where there are more than one area of interest to the observer. This situation requires multiple foveas, where two or more regions are displayed with higher resolution than the remainder of the image.

There is no unique way of defining multiple fovea. However, with additional centers of attention a decision must be made to either reduce the resolution around each fovea or retain additional information for each fovea and reduce the compression ratio.

A method which works well for two fovea situations calculates the location of a point in the transformed image using each fovea separately. The true location is then found by weighting the two estimated points according to the distance of the original point from the fovea. A higher weight is given to the location calculated using the closer fovea:

$$l_{actual} = l_1 * \frac{d_2}{d_1 + d_2} + l_2 * \frac{d_1}{d_1 + d_2} \tag{3.3}$$

where l_i represents the coordinates of a point calculated using fovea i, and d_i represents the distance to fovea i. (See figure 3.4.)

Chapter 4

Compression Performance Measures

When comparing image compression algorithms, it is often difficult to decide which one is the best. The performance of an image compression system can be measured using several criteria: compression ratio, quality of compressed images, and time required for compression/expansion. Performance varies from image to image, and the relative importance of each of these criterion depends on the application. In some situations, the criteria are conflicting while in other cases they complement each other.

There are other criteria by which compression algorithms may be judged. Some of them include:

- Ease of hardware implementation
- Consistency of compression
- Standardization
- Variety of implementations

These measures are not directly related to performance and cannot be directly measured.

4.1 Compression Ratio

The compression ratio is the most straight forward criterion, and is often the most important factor in measuring performance. It is simply the percentage savings in the number of bytes required to represent the data. The compression ratio is a measure which not only can be applied to image compression, but is useful for all other forms of data as well.

In most cases, the compression ratio obtained by an algorithm cannot be determined precisely beforehand. Instead, it depends on the characteristics of the image being compressed; more complex images generally give lower compression ratios. (Variable resolution compression is an exception to this rule.) This can sometimes lead to problems where fixed image sizes are required, such as image transmission using fixed bandwidth. Image compression methods which are lossy in nature generally achieve higher compression ratios than their lossless counterparts.

4.2 Image Quality Measure

The subject of image quality is important only for lossy compression methods. These techniques generally try to alter images so that changes are imperceptible to humans, yet there is no guarantee that changes will go unnoticed.

It must be noted that there is no perfect way to measure the error (distortion or noise) caused by a compression routine since image quality is, to use a cliche, "in the eye of the beholder." Individual opinion on image quality varies, and accurate results based on subjective testing are difficult to obtain.

Some mathematical measures for image quality do exist however [11]. One such measure is the mean-square Signal to Noise Ratio (SNR). For an M*N image, where a pixel in the image is represented by f(x,y) and the same pixel in the compressed image by g(x,y), the SNR can be calculated using:

$$(SNR)_{ms} = \frac{\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} g^2(x,y)}{\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}(g(x,y)-f(x,y))^2}. \tag{4.1}$$

The root mean square Signal to Noise Ratio is simply

$$(SNR)_{rms} = \sqrt{(SNR)_{ms}}. \tag{4.2}$$

Note that the denominator term $g(x,y) - f(x,y)$ in the $(SNR)_{ms}$ equation defines the error in a pixel within the image. Pixels which are unaltered during compression will have this value equal to 0.

Other possible measures of image quality are the mean square error

$$e^2_{ms} = \frac{1}{M*N} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} (g(x,y) - f(x,y))^2 \tag{4.3}$$

and the root mean square error

$$e_{rms} = \sqrt{e^2_{ms}}. \tag{4.4}$$

A characteristic of the mean squared error is that it can be very sensitive to large errors in single pixels. For that reason, the the mean absolute error can be used instead:

$$e_{abs} = \frac{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} |g(x,y) - f(x,y)|}{M*N} \tag{4.5}$$

The mathematical measures for quality are based on global error. (That is, errors in pixels occurring throughout the entire image.) However, because quality in a VR compressed image is expected to vary spatially, special consideration must be given to the problem of analyzing these images. One possible solution is to scale results by considering pixels in the region of the fovea only. (In this thesis, the localized error involves pixels falling within 30% of the maximum possible distance to the fovea are used for scaled quality measures. Errors in pixels outside this area are simply not considered.)

4.3 Compression Time

Improvements in processor speed and the emergence of dedicated hardware has reduced the importance of execution time in compression and expansion. The importance of time cannot be ignored in all circumstances however; real time systems, and software-based systems may require the use of fast algorithms.

The consideration of time in a compression algorithm involves much more than simply its execution times. To begin with, compression times are dependent on the particular implementation and the platform being used. (A hardware implementation of even a slow algorithm may outperform a software implementation of a fast algorithm.) In addition, consistency in image compression and delivery times becomes important in interactive applications (such as videoconferencing).

Finally, the consideration of algorithm execution time by itself is not practical without some thought regarding the compression ratio. Compression methods do not exist in a vacuum; data must be input and output, and this involves a certain amount of overhead. An algorithm providing fast compression but small compression ratios may have its time advantage disappear because of the greater time needed for I/O.

4.4 Combinations of Measurements

The fact that importance of the various measurement criteria differs from application to application makes the performance comparison of different algorithms difficult. If the importance of the time, quality and compression factors are expressed by weights, then a simple additive model for comparison purposes can be determined as:

$$perf = w_{time} * \frac{Time_{max} - Time}{Time_{max}} + w_{quality} * \frac{SNR_{rms}}{SNR_{rmsmax}} + w_{comp} * \frac{Comp}{Comp_{max}} \quad (4.6)$$

where w_i represents the weight (or importance) of factor i in an application, and $\sum_i w_i = 1$. The values of $Time_{max}$ and SNR_{rmsmax} represent the maximum time and image quality obtained out of all compression methods under consideration, while $Comp_{max}$ represents the maximum possible compression. (For $Comp_{max}$, 100% can be used, which represents infinite compression.)

The above equation must be taken in context; there may be situations where a minimum requirement for one of the factors exists. In other cases, there exists a ceiling for certain measurements, and exceeding this ceiling for one criteria does not improve the overall performance.

Chapter 5

Performance Comparisons

The performance of VR compression was tested, and compared with JPEG, Huffman and LZW Compression. The measures used are the same as those presented in the previous chapter, and include compression, quality, and time. JPEG is selected as being representative of DCT-based compression techniques, while Huffman and LZW are typical entropy encoding schemes.

5.1 Testing: Environment and Implementation

The test environment for performance measuring consisted of a Sun SPARC station SLC under Unix. Programs were written to be platform independent, and for the purposes of considering image quality and compression ratios they are. However, the test environment is important in timing comparisons.

Portable Gray Map (PGM) was chosen as the image format for the storage of both uncompressed and VR compressed images [21]. A PGM image file consists of several header bytes (specifying the image dimensions) followed by the raw gray values. PGM was selected for several reasons:

- It is compatible with many image manipulation tools
- It is a simple format, with little overhead and easily-used library manipulation routines

- It is platform independent

One small modification was made to the PGM format: When an image is compressed using VR, several bytes are added at the end of the PGM file, specifying the compression parameters. When the file is manipulated using standard image tools, these extra bytes are ignored.

Data for JPEG compression was obtained using the free software of the Independent JPEG Group. This implementation is stable, platform independent, and the source code is easily accessible. (In fact, it has become a defacto standard for JPEG compression.) The IJG software is able to convert images from many different formats (including PGM) to the JPEG File Interchange Format (JFIF). Control over the amount of compression provided is accomplished with the 'Q' value, which affects the quantization step in the JPEG compression method. (Q must be in the range 0-100, with the highest quality being 100. The range of 'useful' values is smaller though. Note that the relationship between Q, image quality, and compression is not linear.) Some rewriting of the IJG software was necessary for the purposes of data collection.

Data for the LZW and Huffman compression methods were obtained by using the standard Unix implementations of **compress** and **compact** respectively.

5.2 Comparison Results

The sample image used for testing purposes is a 256 * 256 pixel grey scale image and consists of a face on a neutral back ground (as shown in figure 3.1). The uncompressed image is 65551 bytes, 65536 bytes for the image grey levels, and 15 bytes for the image header. The size and content of this image is representative of the type of image that a videoconferencing system might encounter. A single fovea, located at the center of the image, was used in VR compression tests.

5.2.1 Effect of Varying α on Image Quality

Because VR compression depends on two parameters (compression and α), the effect of both of these parameters on image quality is of interest. While it is expected that increasing the compression ratio will decrease image quality, the relationship between α and image quality is not as clear.

Using a fixed compression ratio, the value of α was varied from 0.001 to 1. The image quality for each value of α was tested, using the root mean SNR and the ABSE as measures. This experiment was repeated for compression ratios of 10, 30, 50 and 70%. The results are presented in figure 5.1.

While the value of α does have an effect on global image quality, the exact relationship depends quite heavily on the compression ratio. When the compression ratio is high, the best global image quality is obtained when the value of α is low (close to 0.001). For lower compression ratios, the optimum value of α tends to lie in the range of 0.005 and 0.05, with different optimum values for different compression ratios.

When local image quality near the fovea is considered, the situation changes. High quality with virtually no error can be achieved even at high compression by using a sufficiently high value of α. (The exact value of α where the local error becomes zero depends on the compression used.)

5.2.2 Comparison of VR and JPEG on Image Quality and Compression

The relationship between compression and image quality was examined by comparing images compressed with VR at compression ratios between 10 and 70% (using $\alpha = 0.1$) and JPEG (using quality settings between 10 and 100). The results are presented in figure 5.2. Note that simple LZW encoding can obtain 22% compression, while Huffman encoding can obtain 15%. Because these methods are lossless and cause no distortion in the image, they can be considered as lower limits; any lossy compression method should be able to exceed these values.

In addition to the basic VR and JPEG tests, the compression and quality rela-

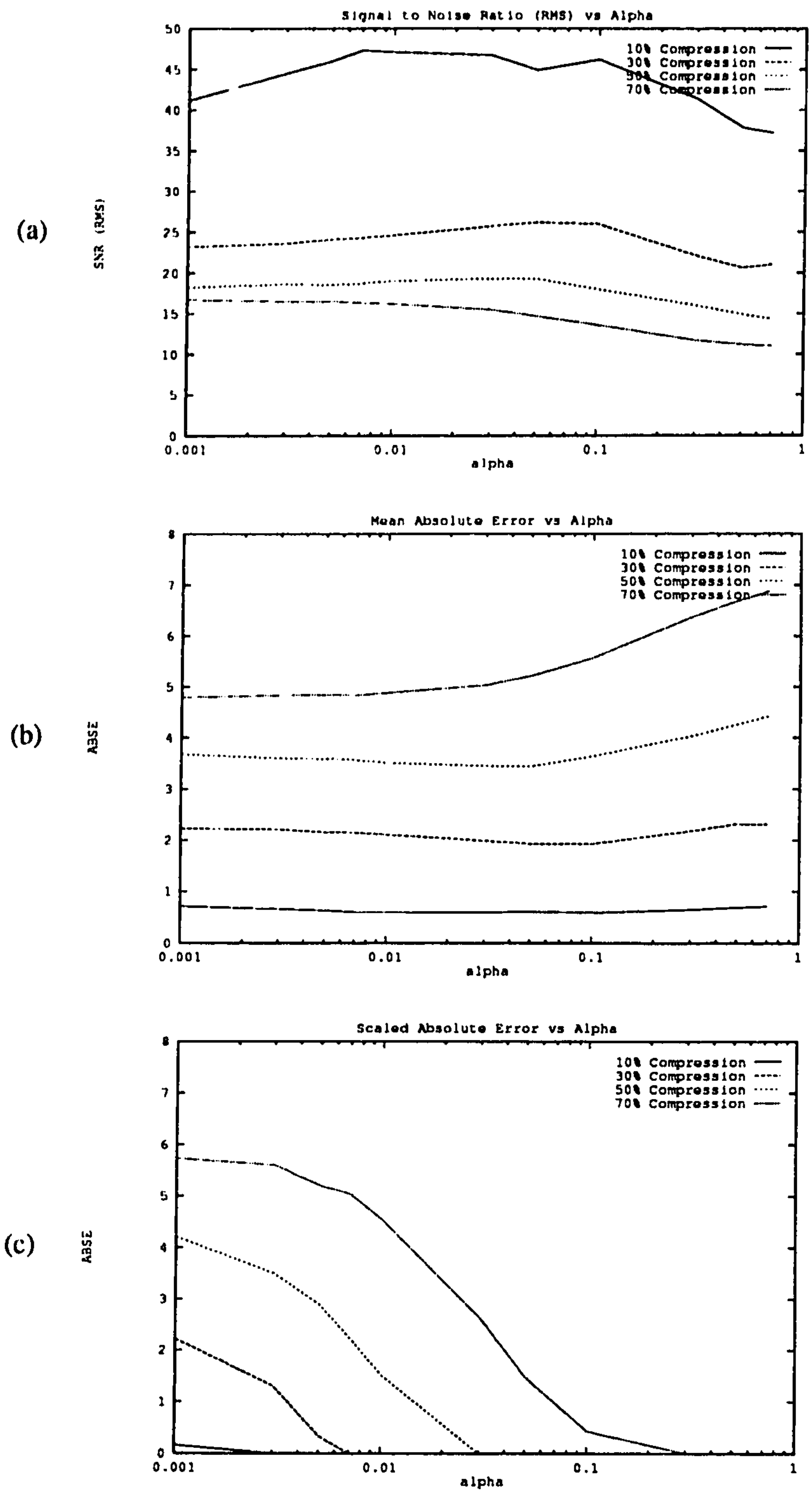

Figure 5.1: Measurement of the effect of changing α on image quality, using constant compression values. Graphs (a) and (b) measure the global signal to noise ratio and mean absolute error respectively. Graph (c) shows the localized absolute error, involving pixels located within 30% distance to the fovea.

tionship was tested for VR/JPEG and VR/LZW hybrids. In these hybrids, JPEG and LZW were applied to images already compressed using VR compression.

When global quality measures are applied, JPEG always manages to produce better compression and higher global quality than VR or the VR/LZW hybrid. Combining JPEG and VR gives better compression than JPEG alone, but with a substantial decrease in global image quality. When considering only compression and global quality, JPEG is clearly the superior method.

However, when local quality measures are used, VR becomes the superior method, compressing images with no pixel error except at high compression. The VR/JPEG hybrid also reacted differently for localized analysis as opposed to global analysis. For much of the compression range, local quality was unaffected by increased compression. In contrast, JPEG produces similar values for local quality as it did for global quality.

It is interesting to note that the performance of LZW and JPEG suffers when combined with VR. For example, LZW gives a compression ratio of 22% on the original image, but only provides an additional 14% compression on VR compressed files already compressed by 70%. This trend of diminishing returns can be observed by comparing the distance between the plots in figure 5.2 for VR and VR with LZW.

5.2.3 Compression, Time, and Quality

The relationship between image quality and compression ratio covered in previous sections is relatively straight forward. When compression time is also considered, analysis becomes much more complex. Comparing time with image quality only or compression only is meaningless. All three measures must be considered. A three dimensional plot, illustrating the relation between the three measures, is shown in figure 5.3.

The range of measured times differs widely between VR and JPEG. For example, the range of times for JPEG is 1.1–1.6 seconds, while VR compression times fall in the range of 0.05–0.14 seconds. JPEG and the VR/JPEG hybrid

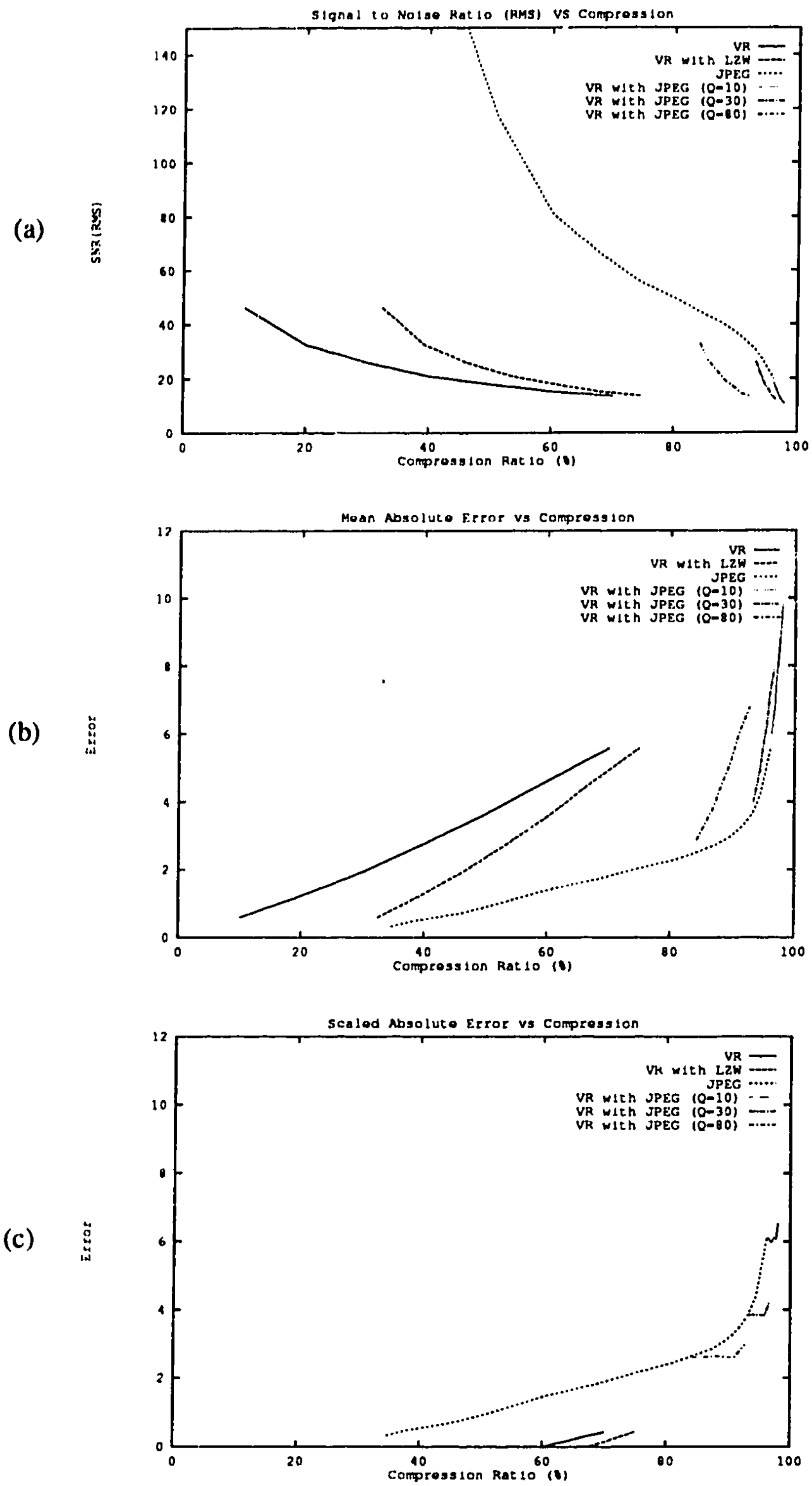

Figure 5.2: Measurement of the effect of changing compression on image quality for several compression methods. Graphs (a) and (b) show global image quality in terms of the signal to noise ratio and mean absolute error. Graph (c) shows the localized absolute error, involving pixels located within 30% distance to the fovea.

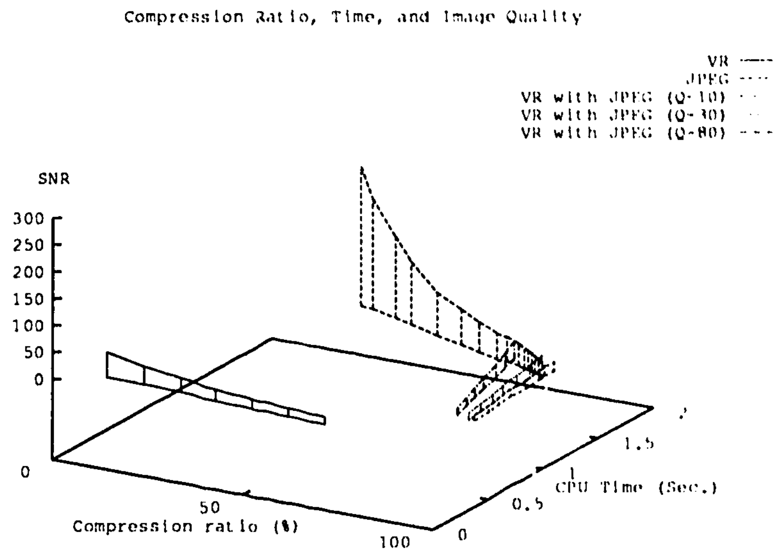

Figure 5.3: A 3 dimension graph showing the relationship between the compression ratio, time, and global image quality for VR and JPEG.

both provide more compression however.

5.2.4 VR/JPEG Hybrids

The compression and time obtained by the VR/JPEG hybrids have been presented in previous chapters. However, such hybrids demand further attention. In particular, the effect of the VR transformation on other compression steps is of interest.

When considering VR/JPEG hybrids, the times required for both the VR and JPEG compression steps must be considered. These are shown in figure 5.4. Clearly, VR/JPEG hybrids can lead to substantial time savings over pure JPEG. There is also an increase in compression. (See figure 5.2 in section 5.2.2.) Global quality does decrease substantially in hybrids, but local quality appears to be very stable, except at extremely high compression.

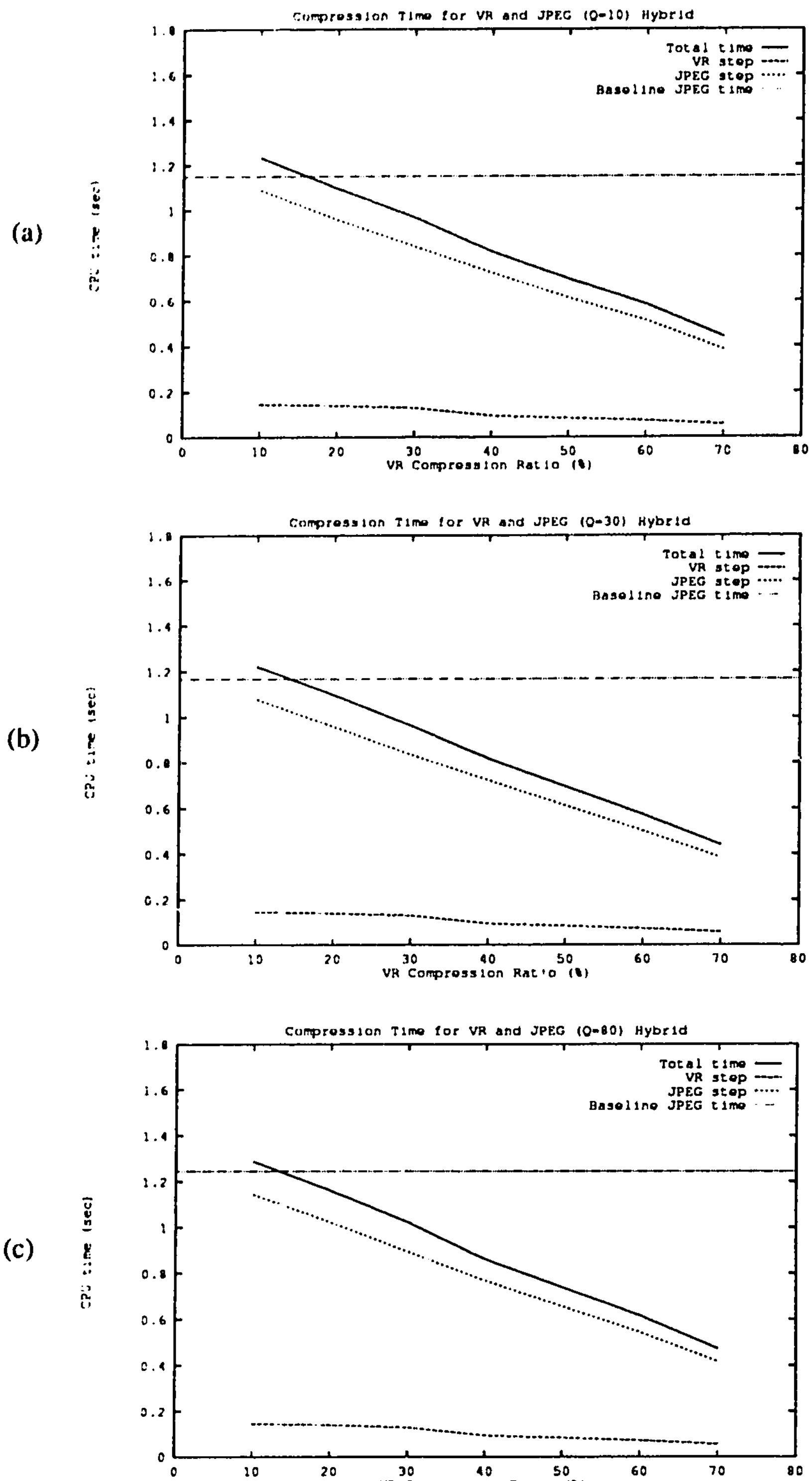

Figure 5.4: CPU time used in the compression of images using a hybrid of VR and JPEG, using various VR compression ratios. JPEG Quality values of (a) Q=10, (b) Q=30 and (c) Q=80 are used. The execution times of simple JPEG compression (no VR preprocessing) is shown for comparison.

Compression Method	Compression Range(%)	Timing Range (Sec)	Global SNR Range	Combined measure Range
JPEG	35.0–96.2	1.15–1.61	19.9–259	0.34–0.45
VR	10.0–70.0	0.06–0.14	13.6–46.2	0.40–0.57
VR/JPEG(Q=80)	84.0–92.7	0.47–1.28	13.0–32.9	0.39–0.56
VR/JPEG(Q=30)	93.4–96.6	0.44–1.22	12.2–26.0	0.42–0.58
VR/JPEG(Q=10)	96.3–98.0	0.44–1.23	10.4–18.2	0.42–0.58

Table 5.1: A table showing ranges of values, along with the combined measure. Equal weights are given to compression, time and SNR when finding the combined measure.

5.2.5 General Performance Measure

The performance of VR, JPEG and the VR/JPEG hybrid were compared using the general performance measure introduced in chapter 4. Ranges for all of the performance measures, plus the combined measure are given in table 5.1. Equal weights were given to all factors in calculating the combined measure.

The best performance comes from the VR/JPEG hybrids. Although image quality is low, they compensate with good compression ratios and times. VR also performs well, giving its best performance with high compression/low quality images. Although JPEG can obtain high quality and good compression its performance lags behind the other compression schemes, mostly due to its high compression times.

It is also interesting to note that unlike VR, which gives its best performance with high compression and low quality, JPEG gives its best performance at both extremes, namely high compression/low quality and low compression/high quality. JPEG's worst performance comes with mid-rage compression and mid-range quality.

Chapter 6

A Videoconferencing System

The idea of videoconferencing (transmitting motion pictures with an ease similar to that of using a telephone) has been around for a long time. Videoconferencing offers some of the advantages of in-person meetings while avoiding many of the costs (e.g. transportation).

Attempts to implement digital videoconferencing systems using low bandwidth transmission facilities goes back to the early 1980s [2]. The early systems were not ideally suited to casual use; equipment was expensive, bandwidth was limited, and compression routines were not efficient. Instead of transmitting full-motion pictures, these systems often would transmit frames only when major scene changes happened.

Increases in computer power, coupled with improvements in networking [16] have opened up videoconferencing to casual use. High speed optical fiber is gradually taking the place of existing copper transmission media, offering much higher bandwidths and greater reliability in both local area and wide area networks [1, 10, 12, 25, 26]. Commercial products, often featuring specialized hardware, bring the ability to videoconference on desk top workstations and PCs. Full motion images can be transmitted, and integrated with text, audio and other forms of data to allow true multimedia collaboration [5]. Transmission of images over standard telephone lines is the goal of companies producing **videophone** products.

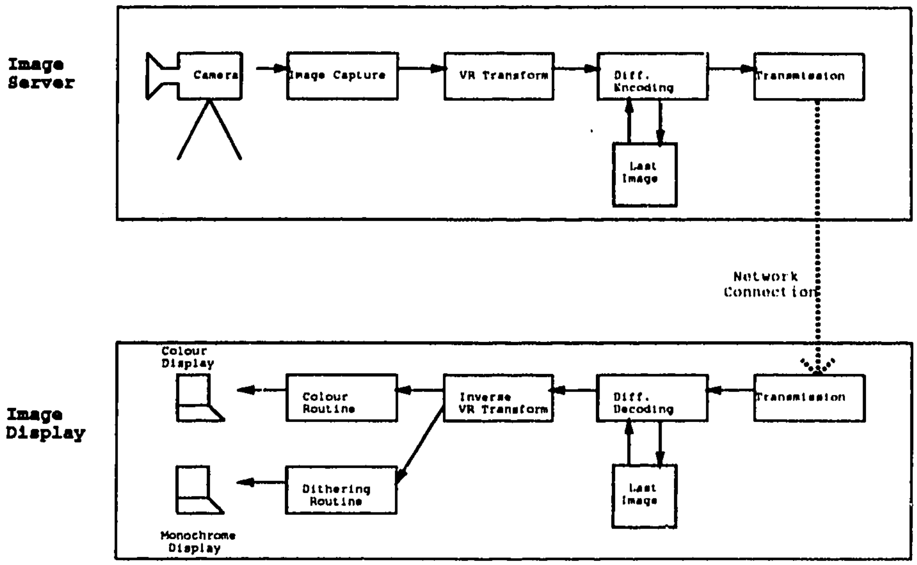

Figure 6.1: Outline of VC, a variable resolution based video conferencing system.

Videoconferencing is a prime application for VR compression for several reasons:

- Videoconferencing requires fast compression of images
- Image quality is not a high priority
- The typical 'talking head' scene provides for an ideal fovea location
- VR provides constant compression, suitable for transmission of images over channels with fixed bandwidth

A prototype of a VR-based videoconferencing system was developed in order to demonstrate its viability. The videoconferencing system is able to provide transmission of grey scale images from an image **server process** to a **display** or **viewer process**. These processes run on separate computers connected by a computer network. The server process is responsible for capturing, compressing and transmitting the image. The viewer process accepts images, expands and displays them on screen.

The system consists of approximately 3000 lines of C code, and was designed to run on Sun SPARCstations running X/Openwindows and Unix. Both colour and monochrome monitors are supported. Unix sockets are used for communication between server and display. Subroutines were written to be general in functionality, so that they may be used in other applications.

The system does not have any facilities for bidirectional communication built in. However, bidirectional communication is possible by having an independent server and display process running simultaneously on a workstation. Some contention for the CPU will occur in this situation, but functionality is not affected. Sound is not handled by the videoconferencing system. However, there are commercial products which can provide sound transmission capability independently.

In addition to variable resolution, additional compression is provided by an intraframe difference encoding routine. The difference between pixels in successive frames is found (most pixels will not change if the image is static, so their difference will be 0) with any series of unchanged pixels transmitted using run-length encoding. The high-order bit of each byte indicates whether the value is a run length of unchanged values or a pixel value.

6.1 Protocols and Communications

Communication between the server and display processes consists of a simple frame request/response protocol. Requests for frames are initiated by the display process, and consist of a request token, an indication of the compression parameters to use, and a time stamp. In fact, the display process is responsible for controlling the entire data stream. Multiple requests may be outstanding, allowing a simple form of pipelining. (The number of outstanding frames allowed is set at compile time.) The display process may also send messages to the server process indicating changes in compression parameters.

In contrast to the display process, the function of the server process is purely reactionary. The server process will remain idle until a frame request is issued

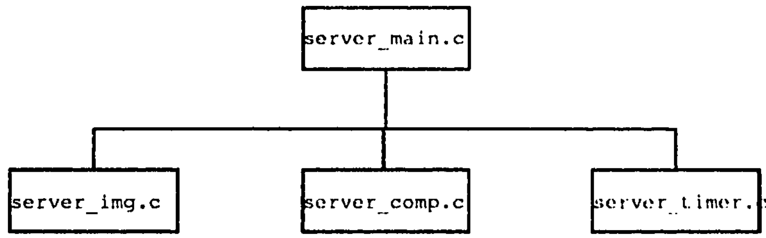

Figure 6.2: Modules of the videoconferencing image server.

by the display. When a request is made, it is accepted, an image is captured, compressed, and transmitted back to the display. Frames are accompanied by a header indicating the type of frame (stand alone or difference encoded), the compression parameters to use, and the time of the initial frame request.

Communication between the server and display processes is done using Unix stream sockets [23, 25]. Stream sockets were chosen to simplify the protocol, since error correction is handled by the underlying network routines. In fact, a pair of unidirectional sockets is used. The **control socket** transmits the frame requests from the display to the server process, while the **data socket** sends the frames from the server to the display process.

The underlying network hardware used in development was Ethernet, but there is no reason why any local or wide area network cannot be used, assuming the correct Socket implementations exist. Note that the Ethernet was shared among many people; although no problems were encountered, contention may be a factor in a heavily loaded network.

6.2 Server Process

The server process does not require any graphics capabilities, and may be run in the background. Any messages that it produces (such as timing statistics for data gathering purposes) are sent to the standard output. Source code for the server is divided into four modules with the following functions: Main, imaging, compression and timing. (See figure 6.2.)

6.2.1 server main.c

The main module contains most of the control code. It creates the sockets used for communication, and handles all communication with the display.

6.2.2 server img.c

The imaging module contains all image hardware dependent routines, including initialization, image capture, and shutdown procedures. The current implementation uses the Videopix board and Vfc software routines for the capture of images [24]. The camera used is statically mounted and provides an NTSC signal to the Videopix board. Any changes resulting from the use of alternate image capture hardware will be localized to this module.

6.2.3 server comp.c

The module contains the routines necessary for VR and intraframe compression. When using the VR routines, an initial call is made to the routines which build the look up table. This lookup table is kept localized to the compression module and consists of a series of memory references where pixels in the compressed image can be found in the image captured by the camera.

6.2.4 server timer.c

This module contains routines used for the collection and display of timing statistics generated by the server. Calls to the timing routines will update a table within the timing module, which is kept internal to the module.

6.3 Display Process

The display process requires the capability of displaying graphics with single pixel resolution. X-windows provides the display process with the ability to display images and to interact with the system. The videoconferencing display

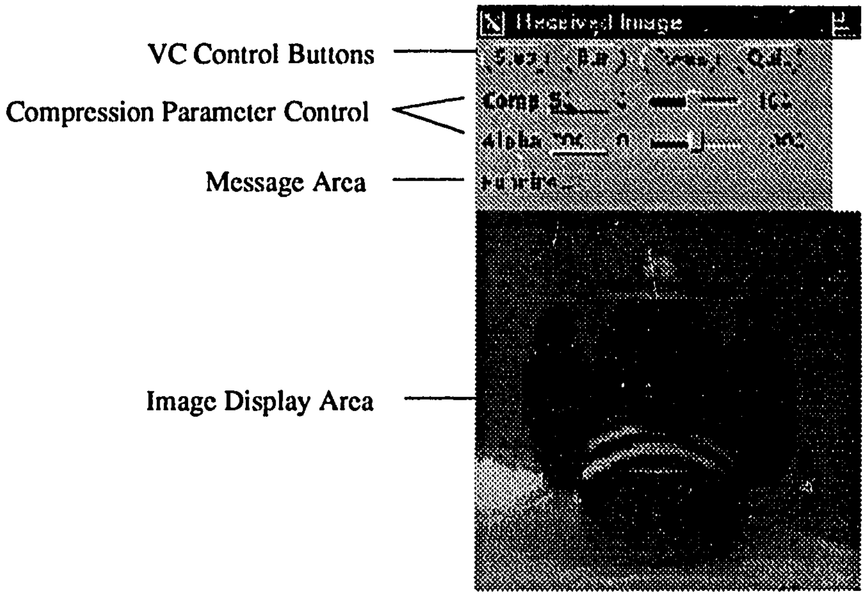

Figure 6.3: Sample videoconferencing display window.

consists of a panel with several control buttons and sliders, a message display line, and a window canvas for the display of pictures. The user may change the location of the foveas and the compression parameters via the control panel.

The display may be in one of two states: **running**, in which frames are continually requested and displayed, and **stepping**, in which frames are requested individually by the user. A change of compression parameters will automatically cause the system to stop while the system recalculates pixel locations for the VR compression.

The source code for the viewer/display is divided into five modules with the following functions: Main, viewing, colour, compression and timing. (See figure 6.4.)

6.3.1 vc main.c

The main module is responsible for establishing the socket connection with the image server. After that, it calls the routines to establish the display window, and starts up the X-windows main loop.

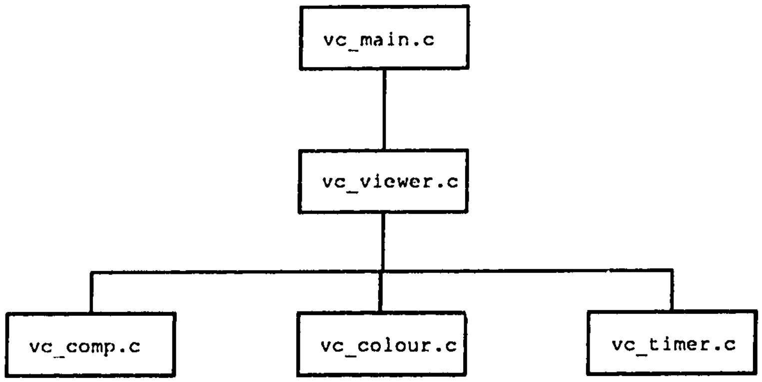

Figure 6.4: Modules of the videoconferencing image viewer.

6.3.2 vc viewer.c

The viewer module contains all of the routines to build the display panel, handle all of the X-window events (pressing of buttons or mouse events inside window, and the arrival of new frames at the data socket.) A number of variables have been localized to this module, including the compression parameters in use, and an indication of the current state of the viewer (running or stepping).

6.3.3 vc comp.c

The compression module contains the routines required for both VR expansion and intraframe difference decoding. As with the server compression routines, an initial call must be made to build a look up table, which will then be used for all subsequent expansion steps.

Note that although every effort was made to make the module non-application specific, this desire conflicted with the need for efficiency in the expansion routines. Therefore, two sets of expansion routines were written, and the choice of routines is selected using a compile time flag. If the flag is set, the VR routine expands the image directly into the X-windows display canvas. Otherwise, the routine provided does a simple memory buffer to memory buffer expansion.

When using monochrome monitors for display purposes, the image is dithered using a 4*4 ordered dither matrix [8, 18]. Ordered dithering is used because of

Compression Setting	Frame types	Frame Rate (per sec)	Compression (%)	Turn around time
0%	No Diff	1.5	0	1.2
	Diff	1.6	87	1.2
50%	No Diff	1.7	50	1.0
	Pred	1.8	95	1.0
90%	No Diff	2.0	90	0.8
	Diff	2.0	98	0.8

Table 6.1: Typical performance for the videoconferencing system, using 256*256 images, based on the compression setting used, and whether frame difference encoding is used.

its speed and ease of implementation.

6.3.4 vc colour.c

X-windows colour routines can be very complex, so a decision was made to isolate the colour specific routines to their own module. Future updates to the system which involve updating the colour allocation will affect this module.

6.3.5 vc timer.c

This module contains routines used for the collection and display of timing statistics generated by the display process. Calls to the timing routines will update a table within the timing module, which is internal to the module.

6.4 Performance

When considering how well a videoconferencing system is working, the most important measures are the frame rate and the frame delay. The resources used by the system are also of interest. Typical frame rates, compression ratios, and turn around time (time from image request to screen display) are shown in table 6.1 for various compression settings, with the interframe difference encoding switched on and off. Also shown in figure 6.5 is the amount of CPU time required for various steps within the server and display programs for a compression setting of 50%.

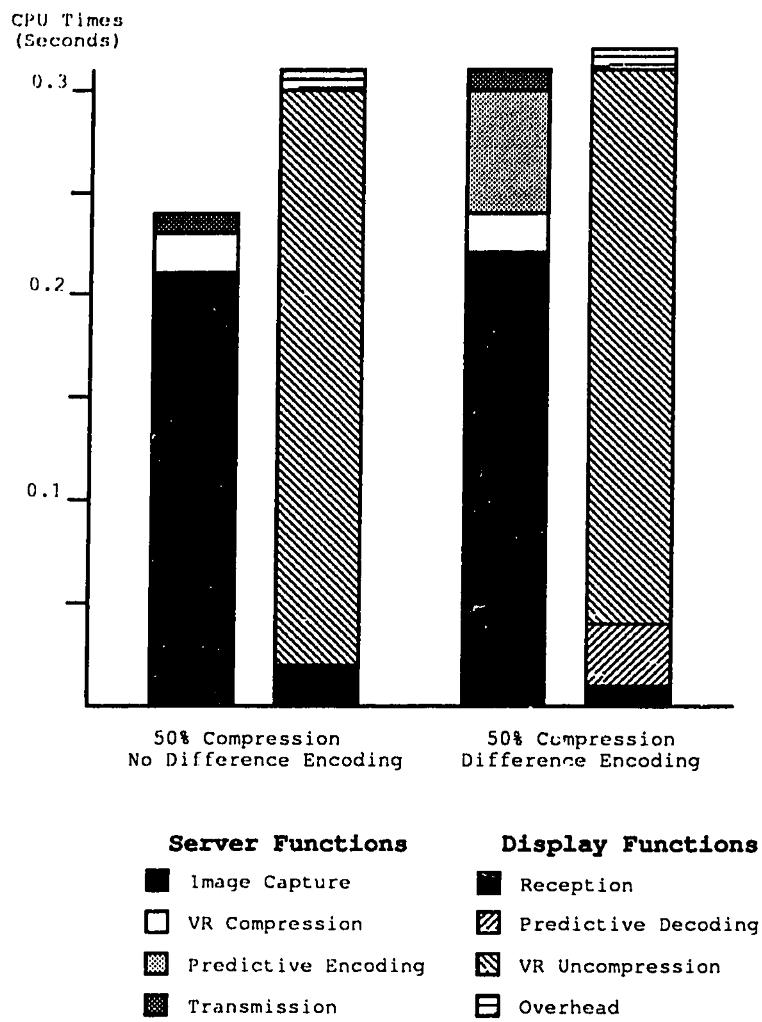

Figure 6.5: CPU times required for various steps within the videoconferencing system using a compression setting of 50%.

From the data provided, it appears that the interframe difference encoding scheme used is very efficient at providing compression, even without the use of VR. It does add significantly to the amount of CPU time used. However, the improved compression that it provides allows for faster communication and higher frame rates.

Higher frame rates are possible with the network that was used (Ethernet). However, the speed of the hardware is a bottle neck. Optimization of the software will partially solve this problem, but it will not be totally eliminated. There is a certain amount of overhead, such as in the image capture routines, which cannot be overcome under the current platform. In the long run, high frame rates will only be possible by using either faster hardware, or special dedicated hardware.

6.5 Possible Improvements

As the current videoconferencing system is only a prototype, there are many improvements which can be made. Many of the routines need to be optimized for efficiency. In particular, the colour allocation routines need to be rewritten. However, care must be taken to ensure that any changes do not make the system overly complex.

Some short-term goals for system improvements include:

- Support for full colour
- Porting server process to use alternate image capture hardware
- Porting display process to different platform
- Stereo image displays

Ultimately, compression which combines VR with other forms of compression (e.g. JPEG), and the ability for multiple display process to connect to the same server process are envisioned.

Videoconferencing is a relatively new application, and concepts for its use are in the early stages of development. However, as with any new technology, there exists a number of issues to be resolved with videoconferencing before its general use [4, 5]. Some of these issues include:

- Privacy
- Provisions for collaborative work (ideally, people may want to use existing application programs)
- Allowances for two way communication
- Providing a paradigm to allow free interaction (e.g. the **virtual room** concept)
- Network control protocols

Eventually, there may come a time where the use of videoconferencing becomes as common as the use of telephones today.

Chapter 7

Conclusion

Variable resolution has a number of attractive qualities as a compression method. The high quality presented in the fovea region makes it ideal for situations when there is a clearly defined point of interest. The ability to compress images to a fixed size makes it ideal for transmission across fixed bandwidth channels. The speed of variable resolution compression makes it ideal for software based implementations of motion picture displays on low powered computers. This is especially true when hardware support of other compression schemes is limited.

Based on the experience obtained from the videoconferencing prototype, one can envisage a University, business or research institution with a VR based videoconferencing system allowing visual communication across networks on standard computers and workstations. Users will be able to have face to face discussions from their offices with people in adjacent offices or around the world just as easily as they use the telephone and intercom today. Or perhaps users could use such systems to tap into lectures or meetings and follow the action through the use of motion tracking cameras. The guaranteed compression ratios provided would benefit communications over networks with fixed bandwidth, such as ISDN networks.

Such a videoconferencing system is obtainable now, using existing computer technology and VR compression. And unlike many other compression systems, no special hardware is needed. Thus, it can be implemented at virtually no cost.

It is a case of providing a service for free which otherwise may be limited to those who can afford expensive hardware. And in situations where computers already have the ability to compress images using other techniques, VR hybrids can be used to provide a widened field of view.

Unfortunately the amount of distortion in the periphery caused by the variable resolution transform makes it unsuitable as a general image compression technique, especially for still images when compression time is not important or hardware support of other methods is available. Many of the other techniques maintain a much higher global image quality even at significantly higher compression (although they often require much more computer time). Using a lossless compression method in addition to VR will help improve compression, but it is doubtful that the compression and quality characteristics of VR will ever exceed those of other lossy methods without improvements to the sampling methods.

With respect to motion picture compression, variable resolution has several disadvantages. Standards for other compression techniques (e.g. MPEG and p*64) are well defined, and commercial hardware implementations are already appearing. Establishing a VR presence in the videoconferencing market will be difficult. While it is possible to use a VR/JPEG or VR/p*64 hybrid, adding the additional layer to existing standards will cause compatibility problems.

The future of image compression likely lies in hardware. When the cost of hardware implementations for JPEG, MPEG, p*64, fractal or other compression schemes decreases enough to make them easily affordable, the importance of compression time will be all but eliminated. However, the time when dedicated hardware implementations become common may still be years away. Until that happens, variable resolution can be used to fill the niche of motion picture compression on computers with limited speed.

7.1 Suggestions for Future work

The work done on VR has a number of possible extensions. The spatial sampling method, while executing quickly, can be improved. Using interpolation during the sampling processes can improve image quality by making the sampled pixel representative of all points that it represents. Although this may slow down compression, early results show that it greatly improves image quality [29].

Image quality may also be improved by using an 'intelligent' VR compression scheme. Rather than allowing the user to set the compression parameters, fovea location could be selected based on a search for areas of complexity within the image. The value of α can then be set in order to optimize the image quality.

When using VR for motion pictures, it may be possible to improve resolution in the periphery by using successive frames to transmit previously unsampled pixels. Where regions of the image are static, quality will be improved over time.

The use of VR to compress colour images needs to be studied further. Because the human eye is more sensitive to pixel intensity than colour, images can be transformed from the red-green-blue colour model to an intensity-chromaticity model. The intensity and chromaticity bands can then be compressed individually, using different compression parameters. (Less detail is required for chromaticity than intensity, so chromaticity can be stored using a higher compression ratio.) This may be preferable to the use of the red-green-blue colour model, where each band would need the same VR parameters.

Finally, other methods of performing VR compression apart from spatial sampling are possible. One such method is to maintain a constant spatial resolution, but reduce the number of bits used to represent each pixel in the periphery. This will lead to greater quantization of levels away from the fovea (in the extreme case, the image will appear bi-level). Some form of dithering may be useful during display.

Another method for implementing VR involves JPEG and its quantization levels. Implementations of JPEG use a constant 'Q' value for blocks throughout the entire image. In theory, a value of Q which varies according to the distance from the block to the fovea could be used. Blocks close to the fovea would use many quantization levels, representative of a high quality JPEG image. Blocks further from the fovea would use few quantization levels, and would likely exhibit the blockiness characteristic of low quality JPEG images.

Bibliography

[1] Bandula Abeysundara and Ahmed Kamal. High-speed local area networks and their performance: A survey. *ACM Computing Surveys*, 23(2):221–264, June 1991.

[2] D. Anastassiou, M. Brown, H. Jones, J. Mitchell, W. Pennebaker, and K. Pennington. Series/1 based videoconferencing system. *IBM Systems Journal*, 22(1/2):97–110, 1983.

[3] Peng Ang, Peter Ruetz, and David Auld. Video compression makes big gains. *IEEE Spectrum*, 28(10):16–19, October 1991.

[4] Alan Borning and Michael Travers. Two approaches to casual interaction over computer and video networks. In S. Robertson, G. Olson, and J. Olson, editors, *CHI '91 Conference Proceedings*, pages 13–19. ACM, 1991.

[5] Mon-Song Chen, Zon-Yin Shae, Dilip Kandlur, Tsipora Barzilai, and Harrick Vin. A multimedia desktop collaboration system. In *Globecom 92*, pages 739–746. IEEE, 1992.

[6] Greg Cockroft and Leo Hourvitz. NeXTstep: Putting JPEG to multiple uses. *Communications of the ACM*, 34(4):45, April 1991.

[7] CompuServe. Graphics interchange format, 1987.

[8] J. Foley, A. van Dam, S. Feiner, and J. Hughes. *Computer Graphics: Principles and Practice*. Addison-Wesley, 2 edition, 1990.

[9] Didier Le Gall. MPEG: A video compression standard for multimedia applications. *Communications of the ACM*, 34(4):46–58, April 1991.

[10] Mario Gerla. High speed local area networks. *IEEE Spectrum*, 28(8):26–31, August 1991.

[11] Rafael Gonzalez and Paul Wintz. *Digital Image Processing.* Addison Wesley, 2 edition, 1987.

[12] Delon Hanson. Progress in fiber optic LAN and MAN standards. *IEEE LCS Magazine*, pages 17–25, May 1990.

[13] Arnaud Jacquin. Image coding based on a fractal theory of iterated contractive image transformations. *IEEE Transactions on Image Processing*, 1(1):18–30, January 1992.

[14] Ronald Jurgen. Digital video. *IEEE Spectrum*, 29(3):24–30, March 1992.

[15] Ronald Jurgen and William Schreiber. All-digital tv's promise/problems. *IEEE Spectrum*, 28(4):28–30,71–73, April 1991.

[16] Peter Kirstein and Jon Crowcroft. International links for research collaboration. *Computer networks and ISDN Systems*, 21:261–273, 1991.

[17] Sergio Licarde. Variable resolution vergence control matching. Master's thesis, University of Alberta, 1993.

[18] C. Lindley. *Practical Image Processing in C.* Addison-Wesley, 1991.

[19] M. Liou. Overview of the p*64 kbit/s video coding standard. *Communications of the ACM*, 34(4):59–63, April 1991.

[20] Mark Nelson. *The Data Compression Book.* M and T Books, 1992.

[21] PBM utilities, 1991. Distributed by the University of Alberta.

[22] Eric Schwartz. Computational anatomy and functional architecture of striate cortex: a spatial mapping approach to perceptual coding. *Vision Research*, 30:645–669, 1980.

[23] Sun Microsystems. Inter-process communication primer, 1986.

[24] Sun Microsystems. Using VideoPix, 1991.

[25] Andrew Tanenbaum. *Computer Networks*. Prentice Hall, 2 edition, 1988.

[26] David Tsao. FDDI: Chapter two. *Data Communications*, 20(13):59–70, September 1991.

[27] Gregory Wallace. The JPEG still picture compression standard. *Communications of the ACM*, 34(4):31–44, April 1991.

[28] Terry Welch. A technique for high performance data compression. *IEEE Computer*, 17(6):645–669, June 1984.

[29] K. Wiebe, 1993. personal communication.