Development of Coding Schemes for Video

Anshuman Biswal & Ashrit Das

under the guidance of

Prof. Pankaj Kumar Sa

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela – 769 008, India

Development of Coding Schemes for Video

Report submitted in

May 2010

to the department of

Computer Science and Engineering

of

National Institute of Technology Rourkela

in partial fulfillment of the requirements

for the degree of

Bachelor of Technology

by

Anshuman Biswal

(Roll 10606018)

Ashrit Das

(Roll 10606017)

under the supervision of

Prof. Pankaj Kumar Sa

Department of Computer Science and Engineering

National Institute of Technology Rourkela

Rourkela – 769 008, India

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, India. `www.nitrkl.ac.in`

Pankaj Kumar Sa
Assistant Professor

May 7, 2010

Certificate

This is to certify that the work in this Project Report entitled Development of Coding Schemes for Video submitted by Anshuman Biswal and Ashrit Das, has been carried out under my supervision and guidance, in partial fulfillment of the requirements for the degree of Bachelor of Technology in Computer Science during session 2006-2010 in the Department of Computer Science and Engineering, National Institute of Technology, Rourkela. To the best of my knowledge, the matter embodied in this report is authentic and has not been submitted to any other University/Institute for the award of any Degree or Diploma.

Pankaj Kumar Sa

Acknowledgment

No thesis is created entirely by an individual, many people have helped to create this thesis and each of their contribution has been valuable.

The enthusiastic guidance and support of *Prof. P K Sa* inspired me to stretch beyond my limits. His profound insight has guided my thinking to improve the final product. My solemnest gratefulness to him.

My sincere thanks to Research Scholar *Mr. Suvendu Rup* for their continuous encouragement and invaluable advice.

Their consistent support and intellectual guidance made us energize and innovate new ideas.

Last, but not least we would like to thank all the professors and lecturers, and members of the Department of Computer Science and Engineering, National Institute of Technology, Rourkela for their generous help in various ways for the completion of this thesis.

Finally, my heartfelt thanks to my family for their unconditional love and support. Words fail me to express my gratitude to my beloved parents, who sacrificed their comfort for my betterment.

Anshuman Biswal
Ashrit Das

Abstract

Video compression refers to the process of reducing the quantity of data used to represent digital video images, and is a combination of spatial image compression and temporal motion compensation. Spatial image compression is done by exploiting the spatial redundancy. Temporal motion compensation is done by exploiting the correlation of the pixels in the nearby frame.

In this thesis, investigations have been made to understand the actual mechanism of compression of still images and applying the principle to the video frames. Initially JPEG compression is analyzed and then it is used in MJPEG compression. In later stages motion estimation techniques are analyzed so as to achieve compression by exploiting the temporal redundancy. Three algorithms for motion estimation are analyzed and compared with each other through their results.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

In the last decade we have seen a boom in digital video industry. Earlier the digital video editing systems were expensive capital items of equipment that only major broadcasters and production companies could afford. But now the same capability is available in a laptop that one can buy off the shelf and it comes with the software for Rs 50,000 which is quite amazing.Now anyone can attain a TV broadcast service because the price of the required hardware and software needed is now within the reach of any organization or individual who cares to get involved. Now we know that the digital representation of raw video signals requires a high capacity, therefore low complexity video coding algorithms must be defined to efficiently compress video sequences for storage and transmission purposes. The proper selection of a video coding algorithm in multimedia applications is an important factor that normally depends on the bandwidth availability , the minimum quality required and the application itself . For instance, a surveillance application may only require limited quality, raising alarms on identification of a human body shape, and a user of a video telephone may be content with only sufficient video quality that enables him to recognise the facial features of his counterpart speaker. However, a viewer of an entertainment video might require a DVD-like service quality to be satisfied with the service.Therefore, the required quality is an application-dependent factor that leads to a range of options in choosing the appropriate video compression scheme [1].

1.2 Types of Data Compression

Types of data compression(according to data loss):

Lossless compression

Lossy compression

1.2.1 Lossless compression

Lossless data compression is a class of data compression algorithms that allows the exact original data to be reconstructed from the compressed data. Lossless compression is used when it is important that the original and the decompressed data be identical, or when no assumption can be made on whether certain deviation is uncritical. Lossless compression is necessary for text, where every character is important. In other words each and every input symbol is very vital. Typical examples are executable programs and source code.

1.2.2 Lossy Compression

A Lossy data compression method is one where compressing data and then decompressing it retrieves data that may well be different from the original but is close enough to be useful in some way. It allows an approximation of the original data to be reconstructed in exchange for better compression rates. In many applications this lack of exact reconstruction is not a major problem. For example, while transmission of speech each and every value of sound sample is not important. The thing that matters the quality of the reconstructed speech is within tolerable limits or not.

Advantages of data compression:

More memory space is available for use

Files can be uploaded and downloaded faster

Increase in file storage options

Disadvantages of data compression:

Increase in complexity

Detrimental effect of transmission error

Slower processing for sophisticated techniques

Requirement for decompressing the previous data

1.3 Objective for Video Compression

As previously stated the raw video signals have a staggering capacity,so to transmit them without compressing means wastage of channel bandwidth and memory space. For example, let us assume that a video frame is digitised in the form of discrete grids of pixels with a resolution of 176 pixels per line and 144 lines per picture. If the picture colour is represented by two chrominance frames, each one of which has half the resolution of the luminance picture, then each video frame will need approximately 38 kbytes to represent its content when each luminance or chrominance component is represented with 8-bit precision. If the video frames are transmitted without compression at a rate of 25 frames per second, then the raw data rate for video sequence is about 7.6 Mbit/s and a 1-minute video clip will require 57 Mbytes of bandwidth. With a similar frame rate as above, the raw video data rate for the sequence is almost 30 Mbit/s, and a 1-minute video clip will then require over 225 Mbytes of bandwidth [1]. Therefore, digital video data must be compressed before transmission so as to optimise the required bandwidth for the provision of a multimedia service.Video compression iz all about trade-offs.The actual importance of the video lies in its use.It all depends on the user requirements ie on which factors they are going to compromise.The various parameters are : Image quality, sound quality, frame rate ,saving disk space, moving content around our network more quickly, saving bandwidth, reducing the playback overhead for older processors, portability across platforms,portability across players ,open standards, licensing costs for the tools, licensing costs for use of content and various other related parameters [2]

Techniques to achieve video compression(according to data redundancy)
Compression for removing Spatial Redundancy
Compression for removing Temporal Redundancy

1.3.1 Compression for removing Spatial Redundancy

Since the pixels in a frame of most 2-D intensity arrays are similar to nearby neighbour pixels,information is unnecessarily replicated in the representation of the correlated pixels. Spatial compression exploits this property of spatial redundancy in the frames to achieve compression.This compression technique is applied to individual frames. The data considered by the encoder is self-contained within a single picture and bears no relationship to other frames in a sequence. Like this sequence of frames are coded by simple video codecs.Motion JPEG is an example of this type of codec.

1.3.2 Compression for removing Temporal Redundancy

Generally in a video sequence,pixels in a frame are similar or dependent on pixels in the nearby frames.This property of pixels being temporally correlated is called temporal redundancy.Temporal compression exploits this property of temporal redundancy between the frames to achieve compression. This compression technique is always lossy,because it is founded on the concept of calculating the differences between successive images and describing those differences, without having to repeat the description of any part of the image that is unchanged.

Chapter 2

Related Work

2.1 JPEG Compression

2.1.1 Introduction

Out of the several things realized,one thing that the JPEG realized is that many digital images have very gradual changes in the intensity over most of the image. Not only this, they also realized that the human eye can only differentiate between similar shades of light-intensity, or luminance, to a certain extent.One can compress digital images by exploiting this weakness of human eye.JPEG realized that the human eye cant notice much of a changes in the levels of luminance because of this one can change most of luminance in each row to a single luminance and then store this one luminance as a single luminance. It also realized that the human eye is only so effective at differentiating between similar colors. The JPEG discovered that apart from removing the most of the variations in luminance, one can still remove most of the slight changes in color (from pixel to pixel) and still end up with a very good representation of the image which is not much different from the original image. This way, instead of storing the individual pixels color and intensity, only the gradual changes of color and luminance (across the picture) need to be stored which results in smaller file size [3]. The basic steps for JPEG compression are : Breaking the image into 8x8 sub-images Shifting the pixel values so that they centre around zero Performing forward DCT Quantization of the resultant matrix Rounding each pixel value to the nearest integer Rearranging the image components in a ziz-zag order Applying entropy encoding(like Huffman coding)

2.1.2 Block Splitting

The image block is first divided into blocks of size 8x8,which are processed left to right,top to bottom. Before computing the DCT of the subimage, its gray values are shifted from a positive range to one centered around zero. For an 8-bit image each pixel has 256 possible values: [0,255]. To center around zero it is necessary to subtract by half the number of possible values, or 128. This centers the activity

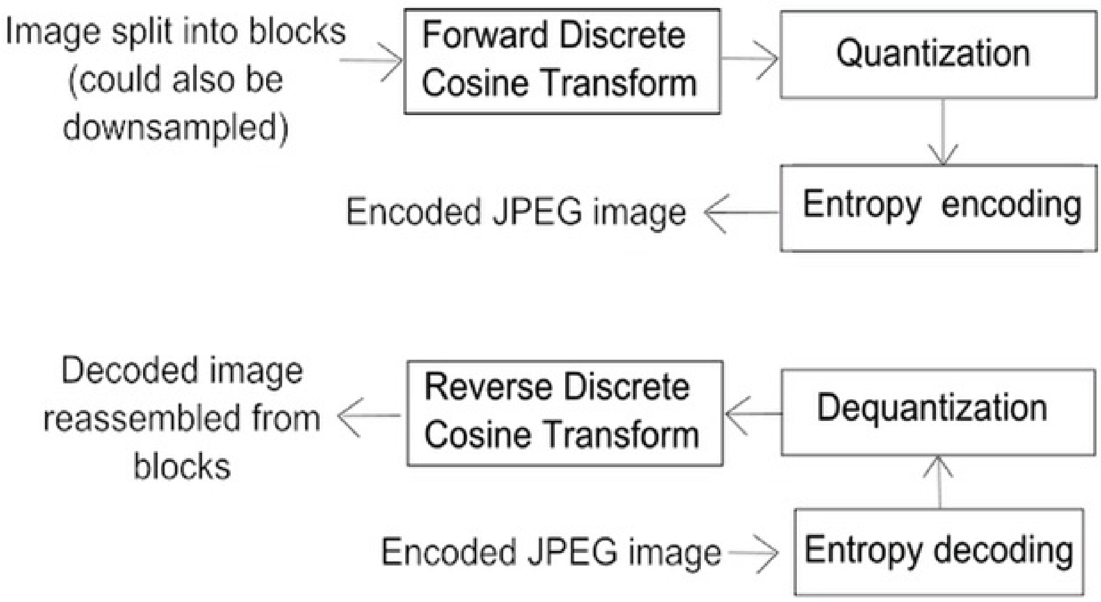


Image split into blocks (could also be downsampled)
Forward Discrete Cosine Transform
Quantization
Entropy encoding
Encoded JPEG image
Decoded image reassembled from blocks
Reverse Discrete Cosine Transform
Dequantization
Encoded JPEG image
Entropy decoding

Figure 2.1: JPEG encoding and decoding process

around 0 for easier handling with cosine functions.

2.1.3 Discrete Cosine Transform

The concept behind JPEG is that one can take the values stored in a picture matrix and transform those numbers from one basis to another, where the new basis stores the relevant information in a more compact form. For the JPEG, the original basis was the two-dimensional spacial basis where every element in the picture matrix represents an actual square pixel which has a spatial position in the picture. A more useful basis which would store the image values more compactly is the frequency basis, where the frequencies represent changes in the luminosity values. Higher frequencies represent rapid changes of luminosity from pixel to pixel, and the low frequencies represent gradual changes across the entire picture. In other words high frequencies refer to the edges and boundaries whereas low frequencies refer to the region having gradual change in the intensity values. The process of getting this basis is by changing coordinates from the spatial domain to the frequency domain by using a transform function like DCT(Discrete Cosine Transfrom). The DCT not only transform any matrix from the spatial basis to the frequency basis, but it segment them into individual frequencies, and this is what makes the DCT so valuable.The Discrete Cosine Transform is of the form:

$$G_{u,v} = \alpha(u)\alpha(v)\sum_{x=0}^{7}\sum_{y=0}^{7} g_{x,y} \cos\left[\frac{\pi}{8}\left(x+\frac{1}{2}\right)u\right] cos\left[\frac{\pi}{8}\left(y+\frac{1}{2}\right)v\right] \tag{2.1}$$

where

- u is the horizontal spatial frequency, for the integers $0 \leq u < 8$.
- v is the vertical spatial frequency, for the integers$0 \leq u < 8$.

$$\alpha_p(n) = \begin{cases} \sqrt{\frac{1}{8}} & \text{if} \quad n = 0 \\ \sqrt{\frac{2}{8}} & \text{if} \quad \text{otherwise} \end{cases} \tag{2.2}$$

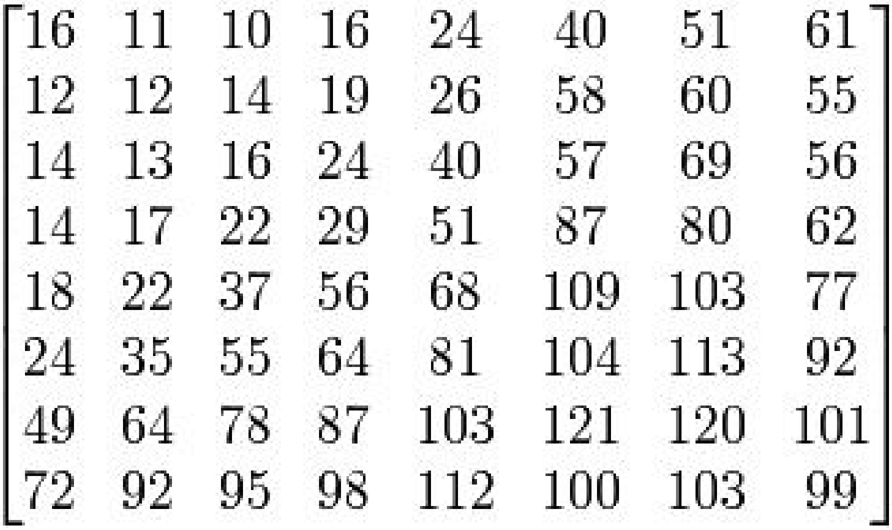
- $g_{x,y}$is the pixel value at coordinates(x, y).
- $G_{u,v}$is the DCT coefficient at coordinates(u, v).

$$\begin{array}{cc} \overset{u}{\longrightarrow} & \\ \begin{bmatrix} -415 & -30 & -61 & 27 & 56 & -20 & -2 & 0 \\ 4 & -22 & -61 & 10 & 13 & -7 & -9 & 5 \\ -47 & 7 & 77 & -25 & -29 & 10 & 5 & -6 \\ -49 & 12 & 34 & -15 & -10 & 6 & 2 & 2 \\ 12 & -7 & -13 & -4 & -2 & 2 & -3 & 3 \\ -8 & 3 & 2 & -6 & -2 & 1 & 4 & 2 \\ -1 & 0 & 0 & -2 & -1 & -3 & 4 & -1 \\ 0 & 0 & -1 & -4 & -1 & 0 & 1 & 2 \end{bmatrix} & \Big\downarrow v \end{array}$$

Figure 2.2: a typical sub-image matrix after DCT

2.1.4 Quantization

We already know that the human eye is efficient at seeing small differences in brightness over a relatively large area, but not so good at distinguishing the absolute strength of a high frequency brightness variation. This allows one to greatly reduce the amount of information in the high frequency components. This is done by simply dividing each component in the frequency domain by a constant for that component, and then rounding to the nearest integer. This is the main lossy operation in the whole process. As a result of this, it is typically the case that many of the higher frequency components are rounded to zero, and many of the rest become small positive or negative numbers, which take many fewer bits to store. A typical quantization matrix, as specified in the original JPEG Standard, is as follows:



$$\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

The quantized DCT coefficients are computed with

$$B_{j,k} = \text{round}\left(\frac{G_{x,y}}{Q_{j,k}}\right) \text{ for } j = 0,1,2,...N_1 - 1;\;\; k = 0,1,2,...N_2 - 1; \qquad (2.3)$$

where G is the unquantized DCT coefficients; Q is the quantization matrix above; and B is the quantized DCT coefficients.

2.1.5 Entropy Coding

Entropy coding is a special form of lossless data compression. It involves arranging the image components in a ”zigzag” order employing run-length encoding (RLE) algorithm that groups similar frequencies together, inserting length coding zeros, and then using Huffman coding.

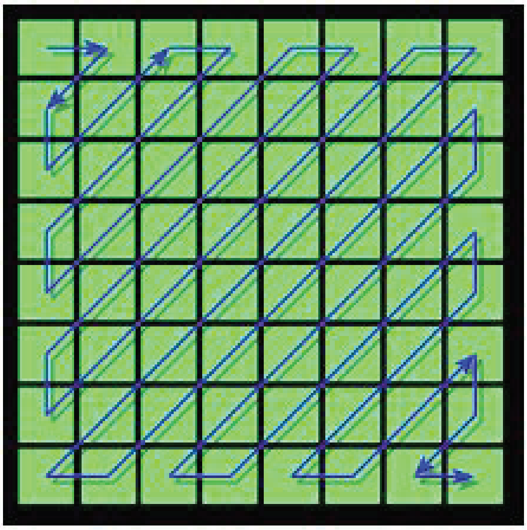


Figure 2.3: Zigzag ordering of JPEG image components

JPEG has a special Huffman code word for ending the sequence prematurely when the remaining coefficients are zero. The JPEG standard provides general-purpose Huffman tables; encoders may also choose to generate Huffman tables optimized for the actual frequency distributions in images being encoded.

2.2 JPEG 2000 Compression

JPEG 2000 is a wavelet-based image compression standard and coding system. It was created by the Joint Photographic Experts Group committee in 2000 with the intention of superseding their original discrete cosine transform-based JPEG standard (created in 1992). The standardized filename extension is .jp2 for

ISO/IEC 15444-1 conforming files and .jpx for the extended part-2 specifications, published as ISO/IEC 15444-2.While there is a modest increase in compression performance of JPEG 2000 compared to JPEG, the main advantage offered by JPEG 2000 is the significant flexibility of the codestream. The codestream obtained after compression of an image with JPEG 2000 is scalable in nature, meaning that it can be decoded in a number of ways; for instance, by truncating the codestream at any point, one may obtain a representation of the image at a lower resolution, or signal-to-noise ratio. By ordering the codestream in various ways, applications can achieve significant performance increases. However, as a consequence of this flexibility, JPEG 2000 requires encoders/decoders that are complex and computationally demanding. Another difference, in comparison with JPEG, is in terms of visual artifacts: JPEG 2000 produces ringing artifacts, manifested as blur and rings near edges in the image, while JPEG produces ringing artifacts and 'blocking' artifacts, due to its 88 blocks. The aim of JPEG 2000 is not only improving compression performance over JPEG but also adding (or improving) features such as scalability and editability. In fact, JPEG 2000's improvement in compression performance relative to the original JPEG standard is actually rather modest and should not ordinarily be the primary consideration for evaluating the design. Very low and very high compression rates are supported in JPEG 2000. In fact, the graceful ability of the design to handle a very large range of effective bit rates is one of the strengths of JPEG 2000. For example, to reduce the number of bits for a picture below a certain amount, the advisable thing to do with the first JPEG standard is to reduce the resolution of the input image before encoding it. That's unnecessary when using JPEG 2000, because JPEG 2000 already does this automatically through its multiresolution decomposition structure.

Chapter 3

Study of video compression scheme using BWT

3.1 Introduction

The idea in BWT Compression is to apply a reversible transformation to a block of text to form a new block that contains the same characters, but is easier to compress by simple compression algorithms. The transformation tends to group characters together so that the probability of finding a character close to another instance of the same character is increased substantially [4]. When a character string is transformed by the BWT, none of its characters change value. The transformation permutes the order of the characters. If the original string had several substrings that occurred often, then the transformed string will have several places where a single character is repeated multiple times in a row. This is useful for compression, since it tends to be easy to compress a string that has runs of repeated characters by techniques such as move-to-front transform and run-length encoding.

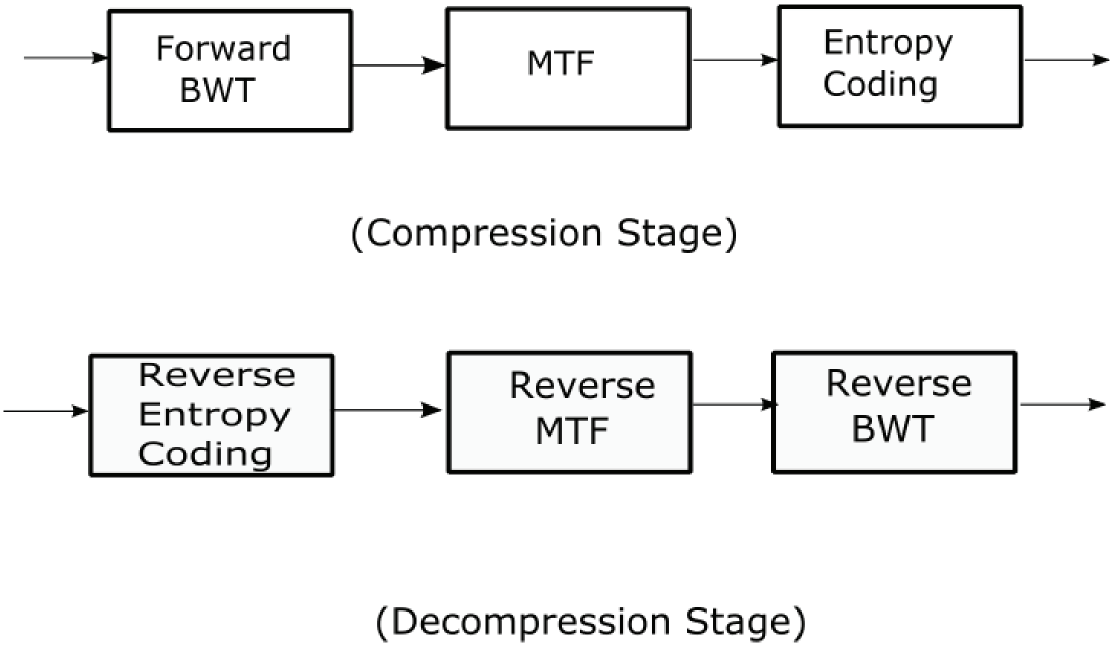


Figure 3.1: Steps for Compression and Decompression

The following pseudocode gives a simple but inefficient way to calculate the BWT and its inverse. It assumes that the input string s contains a special character 'EOF' which is the last character, occurs nowhere else in the text, and is ignored during sorting.

```
function BWT (string s)
    create a table, rows are all possible rotations of s
    sort rows alphabetically
    textbfreturn (last column of the table)

textbffunction inverseBWT (string s)
    create empty table

    repeat length(s) times
        insert s as a column of table before first column of the table // first
        //insert creates first column sort rows of the table alphabetically


    sort rows of the table alphabetically
    return (row that ends with the 'EOF' character)
```

To understand why this creates more-easily-compressible data, let's consider transforming a long English text frequently containing the word "the". Sorting the rotations of this text will often group rotations starting with "he " together, and the last character of that rotation (which is also the character before the "he ") will usually be "t", so the result of the transform would contain a number of "t" characters along with the perhaps less-common exceptions (such as if it contains "Brahe ") mixed in. So it can be seen that the success of this transform depends upon one value having a high probability of occurring before a sequence, so that in general it needs fairly long samples (a few kilobytes at least) of appropriate data (such as text). The remarkable thing about the BWT is not that it generates a more easily encoded outputan ordinary sort would do thatbut that it is reversible, allowing the original document to be re-generated from the last column data. The inverse can be understood this way. Take the final table in the BWT algorithm, and erase all but the last column. Given only this information, one can easily reconstruct the first column. The last column tells one all the characters in the text, so just sort these characters to get the first column. Then, the first and last columns together give us all pairs of successive characters in the document, where pairs are taken cyclically so that the last and first character form a pair. Sorting the list of pairs gives the first and second columns. Continuing in this manner, we can reconstruct the entire list.In case of image compression we can convert the pixels values into corresponding equivalent alphabets and then apply BWT on them.

3.2 Video Compression using BWT

Numerous video coding schemes available today,but the selection of an appropriate coding algorithm for a specific service becomes a important matter.Nowadays the block based video coders are more popular in multimedia services available today.Basically two different types of coding exist in a block based video coder, namely INTRA and INTER coding modes. INTRA mode treats a video frame as a still image without taking care of the temporal redundancy present among the frames.All macro blocks (MB) of a frame are INTRA coded in INTRA frame coding mode. But,in INTER frame coding, some MBs could still be INTRA coded if a motion activity threshold has not been attained.Now generally in a video sequence, nearby frames could have strong correlation. This temporal correlation could be exploited to achieve higher compression efficiency by swichting to INTER mode. In our work we have applied the BWT to a video compression standard. Normally in a video sequence there is a strong correlation between the adjacent frames. So the Burrows Wheeler Transform takes this opportunity and forms a good regularity structure with the help of Move to Front encoding. Then BWT is used before entropy coding.The output of BWT is fed to entropy encoder. Simulation has been carried out for the proposed scheme in a standard video. Scheme which uses only the entropy coding is also been simulated. Comparative analysis in terms of bit rate and PSNR are compared. The proposed scheme shows a significant improvement over the entropy coding.The complete description of hybrid video compression using BWT is explained below.

Frame Decison

H.263 supports inter picture prediction that is based on motion estimation and compensation.INTER mode is used where temporal prediction is required . In this mode only the prediction error frames that are the difference between original frame and motion compensated predicted frames need to be encoded.INTRA coding mode is used if temporal prediction is not employed.

Motion Estimation and Compensation

Motion compensation prediction suggests that for each macro block in the current frame is predicted from the previous frame.Motion vector or a two dimensional displacement vector represents motion information. There are several kinds of motion estimation methods are exist such as Full search motion estimation, Half pixel motion estimation, Three step motion estimation

DCT Transform

To reduce the correlation between the coefficients of a MB, The pixels are transformed from spatial domain into frequency domain by means of transform function called DCT(Discrete Cosine Transform). In this work the 8X8 DCT is applied to decorrelate the 8x8 blocks of original pixel or motion compensated difference pixels.DCT is used to compact energy of the pixel into few numbers of coefficients.

Quantization

The most important component of the video encoder is the quantizer since it controls both the coding efficiency and the quality of the reconstructed video sequence. For every element position in the DCT output matrix, a corresponding quantization value is computed.The same step size within a macro block is used in H.263 quantization.

ZigZag Scan Coding

The output of two dimensional quantized blocks performs a zigzag pattern coding. In a Zigzag coding arrangement places the DC coefficient first in the array and the remaining AC coefficients are ordered from low to high frequency.

Burrows Wheeler Transform and Move to Front Coding

We note that the resulting output of zigzag coding is concentrated sequentially in a one dimensional stream with a number of successive zeros.This regular structure is the point where BWT comes into role because it takes the advantage of successive or repeative pattern. The BWT takes a cyclic shift for the given data. With this cyclic permutation the BWT sorted the data in a lexicographic manner. From this sorted matrix it takes the final column and passes to MTF encoding. MTF encoding assumes that symbols can be ranked according to their closeness of their last occurrence. For strong correlation of video sequence the long runs of the same data will give long runs of zeros in the MTF output.

Entropy Encoding

After MTF the resulting output along with the motion vector passed to the entropy coding.In entropy coding discrete-valued source symbols are represented in a manner that takes advantage of the relative probabilities of the various possible values of each source symbol. A well-known type of entropy code is the variable length code (VLC), which involves establishing a tree-structured code table that uses short binary strings to represent symbol values that have higher probability of occurence and longer binary strings to represent less likely symbol values.Huffman code method is used for designing VLCs since it produces an optimal VLC. At the decoder side the reverse entropy coding is performed followed by reverse BWT and MTF followed by inverse quantization and IDCT.

Chapter 4

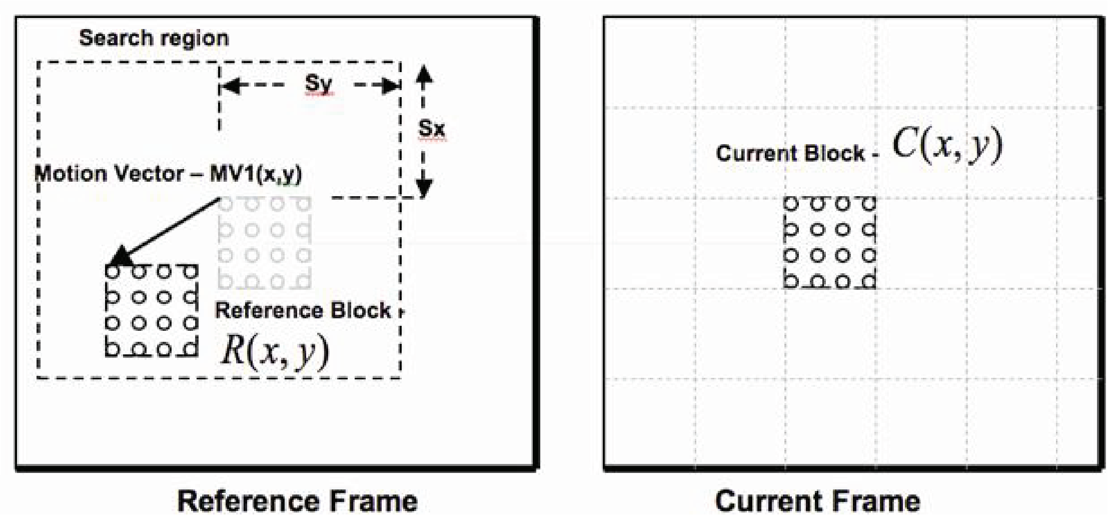
Motion Estimation using Block Matching Algorithms

4.1 Introduction

Motion estimation is the process of determining motion vectors that describe the transformation from one 2D image to another; usually from adjacent frames in a video sequence. It is an ill-posed problem as the motion is in three dimensions but the images are a projection of the 3D scene onto a 2D plane. The motion vectors may relate to the whole image (global motion estimation) or specific parts, such as rectangular blocks, arbitrary shaped patches or even per pixel. The motion vectors may be represented by a translational model or many other models that can approximate the motion of a real video camera, such as rotation and translation in all three dimensions and zoom. The methods for finding motion vectors can be categorised into pixel based methods (”direct”) and feature based methods (”indirect”). Motion estimation is an indispensable part of video compression and video processing. It helps to bring out information about motion of certain objects or features from the video sequence. The motion is typically represented using a motion vector (x,y). The motion vector gives the displacement of a pixel or a pixel block (called macroblock) from the current pixel or macroblock location due to motion. Motion vector which contains motion information helps to find out best matching block using block matching techniques to generate and predict temporally interpolated frames. It is also used in applications such motion compensated de-interlacing, video stabilization, motion tracking etc. There are several kinds of motion estimation techniques available. Some work on a pixel level and find motion vector for each pixel. However the most well known and widely used technique is block matching algorithm (BMA).

4.2 Block Matching Algorithm

Block matching finds the motion vector not for a single pixel but for a group of pixels called a macroblock. Some common macroblock sizes are 8×8 and 16×16 pixels. The are square shaped macroblocks. Using macroblocks instead of individual pixels saves on computations greatly and is also more intelligent and intuitive as objects have features in clusters rather than manifesting in single pixels. Block matching is illustrated in the above figure. The frame under consideration is divided in to macroblocks and motion estimation is performed on each macroblock. Motion estimation is done by identifying a pixel block from the reference frame that best matches the current block, whose motion is being estimated. The reference macroblock is created by displacement from the current blocks location in the reference frame. This displacement is denoted by the motion vector (MV). MV consists of is a pair (x, y) of horizontal and vertical displacement values. Two popular techniques for finding a block match among so many techniques available are Sum of square error(SSE)



$$\sum_{x=1}^{N}\sum_{y=1}^{N}(C(x,y)-R(x,y))^2 \tag{4.1}$$

And the other one is , Sum of absolute differences (SAD)

$$\sum_{x=1}^{N}\sum_{y=1}^{N}|C(x,y)-R(x,y)| \tag{4.2}$$

SSE is a better block matching technique as it gives better results in case a match does exist at the subjective human perception level , however is computationally more burdensome than the SAD technique which also gives fairly good estimates of a match if found. Hence there are other parameters that might also be checked to ensure better results like cross correlation, maximum matching pixel count etc. The reference pixel blocks are created only from a region called the search window or search area. Search window limits the number of blocks to be evaluated. The size of this window invariably depends on the amount of motion present and the computational challenge that can be dealt with properly. Larger search window demands more computation as it should be. Typically the search region is kept wider (i.e. width is more than height) since many video sequences often exhibit panning motion. The horizontal and vertical search range, Sx and Sy, define the search area (+/-Sx and +/- Sy) as illustrated in figure.

4.2.1 Exhaustive search

Exhaustive search generates the best block matching motion vectors because it searches every block in the search window. However due to astronomically high number of computations required, this technique can result in prohibitive cost of computation. The number of candidates to evaluate are $(2Sx + 1) \times (2Sy + 1)$. There are several other fast block matching algorithms available that reduce number of computations required significantly at the cost of slightly reducing performance because these techniques find the local minima and not the global one as in case of full or extensive search.

4.2.2 Three Step Search

The idea of the three step search is represented in the figure above. This procedure starts searching at the centre location i.e. (0,0). The step size is set to four for a fairly common search parameter value of seven. The eight pixels around the search origin are searched for the best match. The one among the nine locations (including the centre location) giving the least cost and hence then best match is made the new search origin. Then the above procedure is repeated but with step size half the original and then a third time with step size one fourth the first time value. The motion vector is calculated based on the final macro block reached at. This motion vector is then saved for transmission. This technique gives a

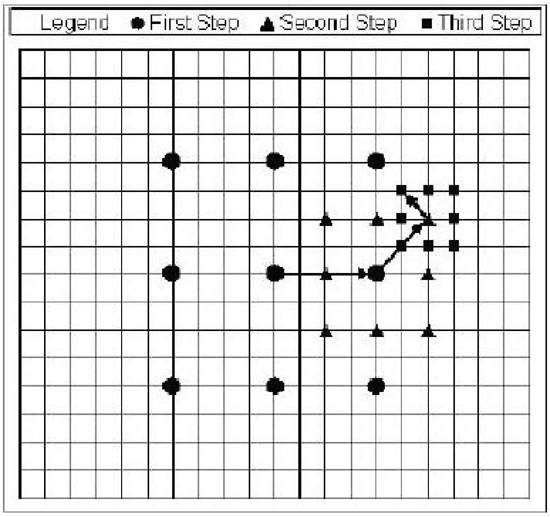


Figure 4.1: Three Step Search Procedure

straight forward reduction on the number of computations performed compared to the extensive search technique. The reduction factor is 9. The idea behind TSS is that the error surface due to motion in every macro block is unimodal. A unimodal surface is a bowl shaped surface such that the weights generated by the cost function increase monotonically from the global minimum [5].

4.2.3 Four Step

Four step searching is also a centre based searching technique and has a halfway stopping provision [6]. Irrespective of the value of the parameter p, 4SS has a search parameter value of S=2. Thus it looks for a match among the 9 windows in a 5X5 neighborhood. If the least weight is found at the center of search window the search jumps to fourth step. If the least weight is at one of the eight locations except the center, then we make it the search origin and move to the second step. Even now the search window is maintained 5×5 pixels wide. Depending on the last best match block we have to check either 3 or 5 locations as depicted in the figure. If the best match is at the centre of the window we go to 4th step. If not then we go into the third step of the procedure. The third step is a repetition of the second step. In the 4th step the search window is made of size 3×3, i.e. S=1.

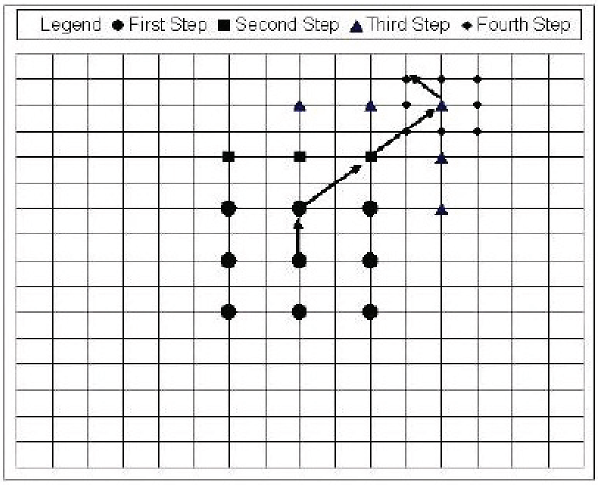

Legend ● First Step ■ Second Step ▲ Third Step ◆ Fourth Step

Figure 4.2: Four Step Search Procedure

The location with best match is set as the final location and the motion vector is calculated according to that point. This search algorithm has the best case of 17 checking points and worst case of 27 checking points [5].

4.3 Proposed Fast Motion Estimation Techniques Using H.264

Numerous video coding schemes available today,but the selection of an appropriate coding algorithm for a specific service becomes a important matter.Nowadays the block based video coders are more popular in multimedia services available today.Basically two different types of coding exist in a block based video coder, namely INTRA and INTER coding modes. INTRA mode treats a video frame as a still image without taking care of the temporal redundancy present among the frames.All macro blocks (MB) of a frame are INTRA coded in INTRA frame coding mode. But,in INTER frame coding, some MBs could still be INTRA coded if a motion activity threshold has not been attained.Now generally in a video sequence, nearby frames could have strong correlation. This temporal correlation could be exploited to achieve higher compression efficiency by swichting to

INTER mode. In our work we have applied the BWT to a video compression standard. Normally in a video sequence there is a strong correlation between the adjacent frames. So the Burrows Wheeler Transform takes this opportunity and forms a good regularity structure with the help of Move to Front encoding. Then BWT is used before entropy coding.The output of BWT is fed to entropy encoder. Simulation has been carried out for the proposed scheme in a standard video. Scheme which uses only the entropy coding is also been simulated. Comparative analysis in terms of bit rate and PSNR are compared. The proposed scheme shows a significant improvement over the entropy coding.The complete description of hybrid video compression using BWT is explained below.

Frame Decison

H.263 supports inter picture prediction that is based on motion estimation and compensation.INTER mode is used where temporal prediction is required . In this mode only the prediction error frames that are the difference between original frame and motion compensated predicted frames need to be encoded.INTRA coding mode is used if temporal prediction is not employed.

Motion Estimation and Compensation

Motion compensation prediction suggests that for each macro block in the current frame is predicted from the previous frame.Motion vector or a two dimensional displacement vector represents motion information. There are several kinds of motion estimation methods are exist such as Full search motion estimation, Half pixel motion estimation, Three step motion estimation

DCT Transform

To reduce the correlation between the coefficients of a MB, The pixels are transformed from spatial domain into frequency domain by means of transform function called DCT(Discrete Cosine Transform). In this work the 8×8 DCT is applied to decorrelate the 8×8 blocks of original pixel or motion compensated difference pixels.DCT is used to compact energy of the pixel into few numbers of coefficients.

Quantization

The most important component of the video encoder is the quantizer since it controls both the coding efficiency and the quality of the reconstructed video sequence. For every element position in the DCT output matrix, a corresponding quantization value is computed.The same step size within a macro block is used in H.263 quantization.

ZigZag Scan Coding

The output of two dimensional quantized blocks performs a zigzag pattern coding. In a Zigzag coding arrangement places the DC coefficient first in the array and the remaining AC coefficients are ordered from low to high frequency.

Entropy Encoding

After MTF the resulting output along with the motion vector passed to the entropy coding.In entropy coding discrete-valued source symbols are represented in a manner that takes advantage of the relative probabilities of the various possible values of each source symbol. A well-known type of entropy code is the variable length code (VLC), which involves establishing a tree-structured code table that uses short binary strings to represent symbol values that have higher probability of occurence and longer binary strings to represent less likely symbol values.Huffman code method is used for designing VLCs since it produces an optimal VLC. At the decoder side the reverse entropy coding is performed followed by reverse BWT and MTF followed by inverse quantization and IDCT.

4.4 Result and Discussion

To validate the efficiency of the proposed scheme, simulation has been carried out on a standard video sequence namely Foreman. For the video the first 25 frames are considered. The following schemes namely,

Scheme 1: TSS(Three Step Search) using H.264.

Scheme 2: FSS(Four Step Search) using H.264.

Scheme 3: ES(Exhaustive Search) using H.264

Scheme 3 clearly outperforms better result with scheme 1 and scheme 2.The scheme 3 is optimised for better bitrate efficiency to a great extent.

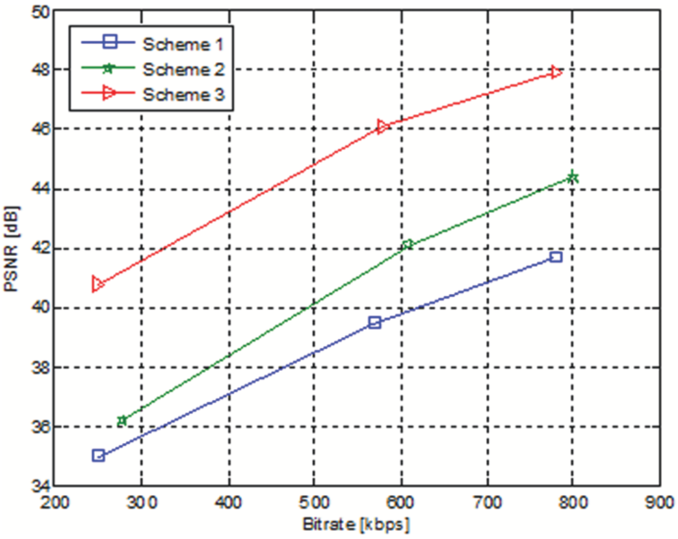


Figure 4.3: Rate Distortion performance

Chapter 5

Implementation and Result

5.1 JPEG Implementation

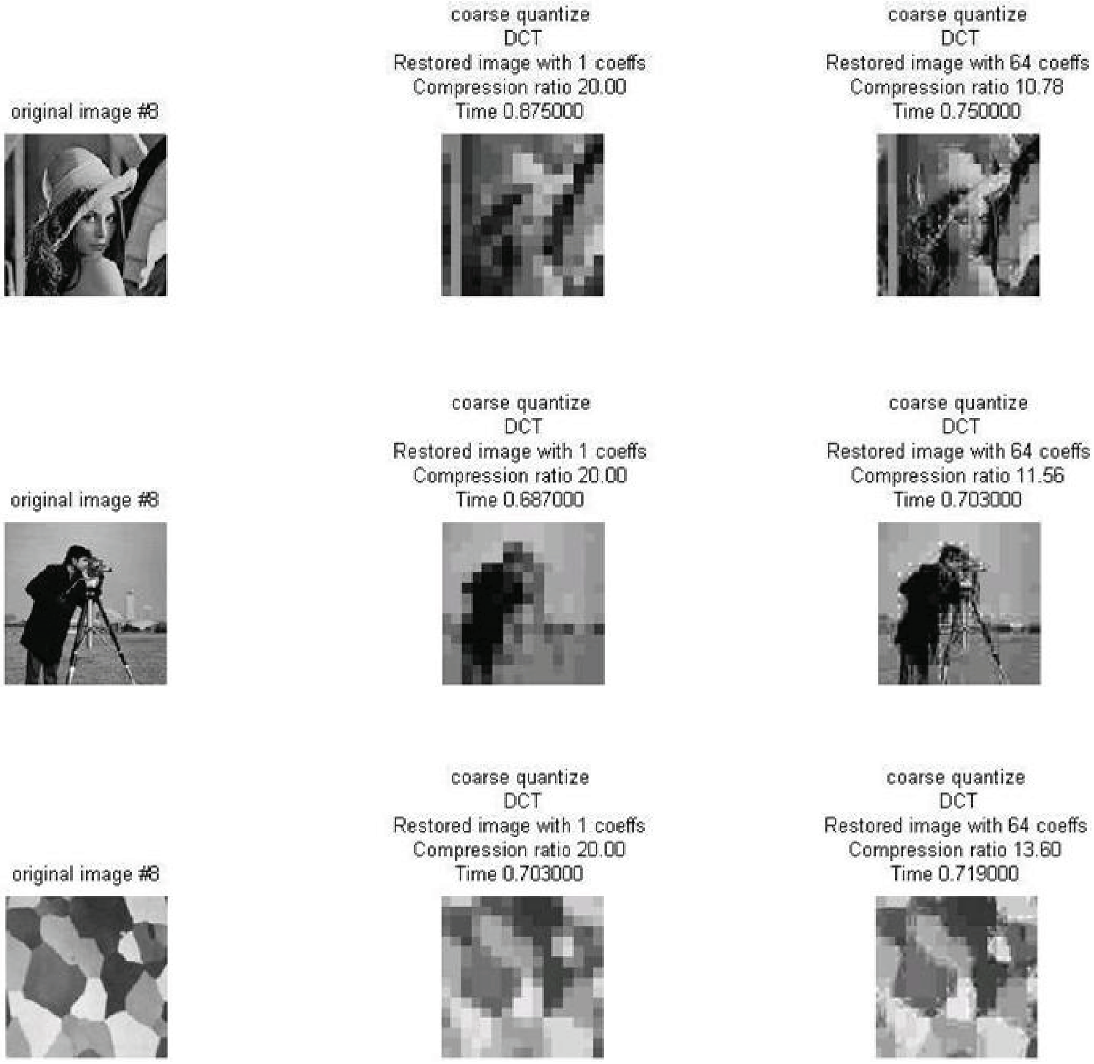


Figure 5.1: JPEG Results

5.2 Motion Estimation

Window size = 3

Exhaustive Search:

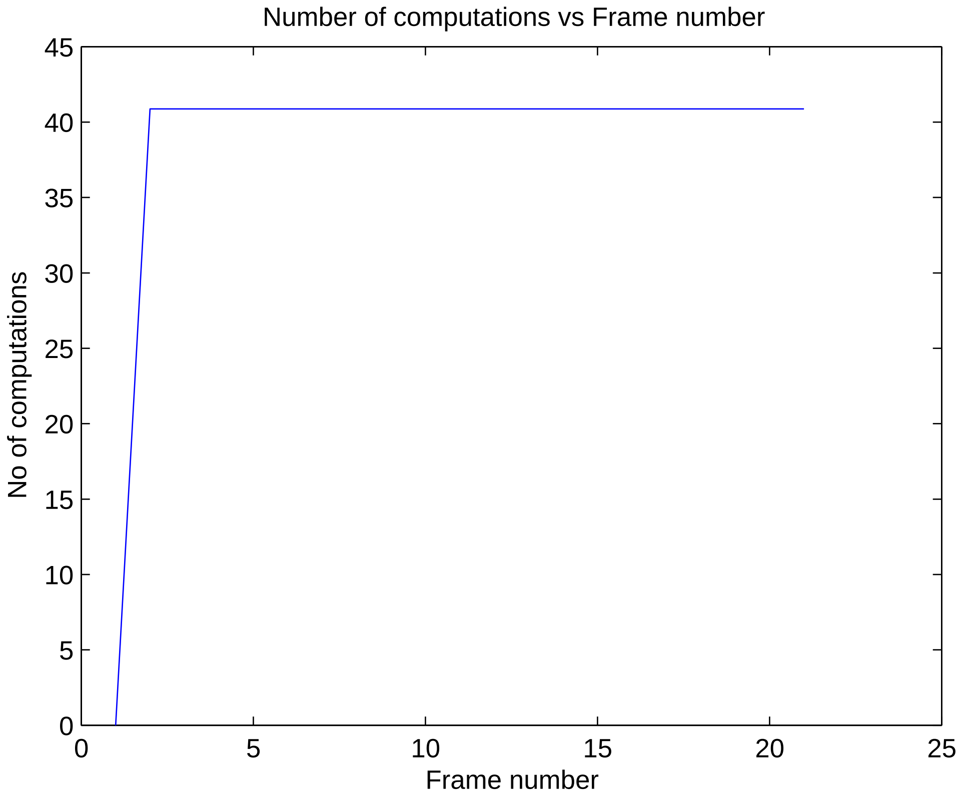


Figure 5.2: Exhaustive Search:No of Computations

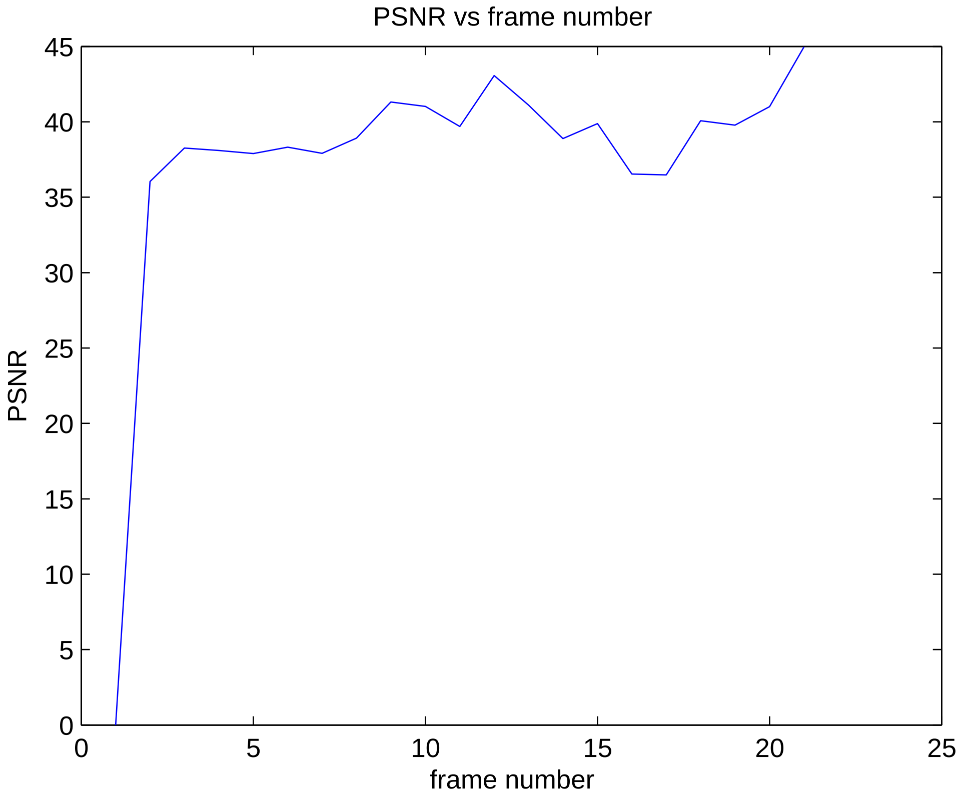


Figure 5.3: Exhaustive Search:PSNR

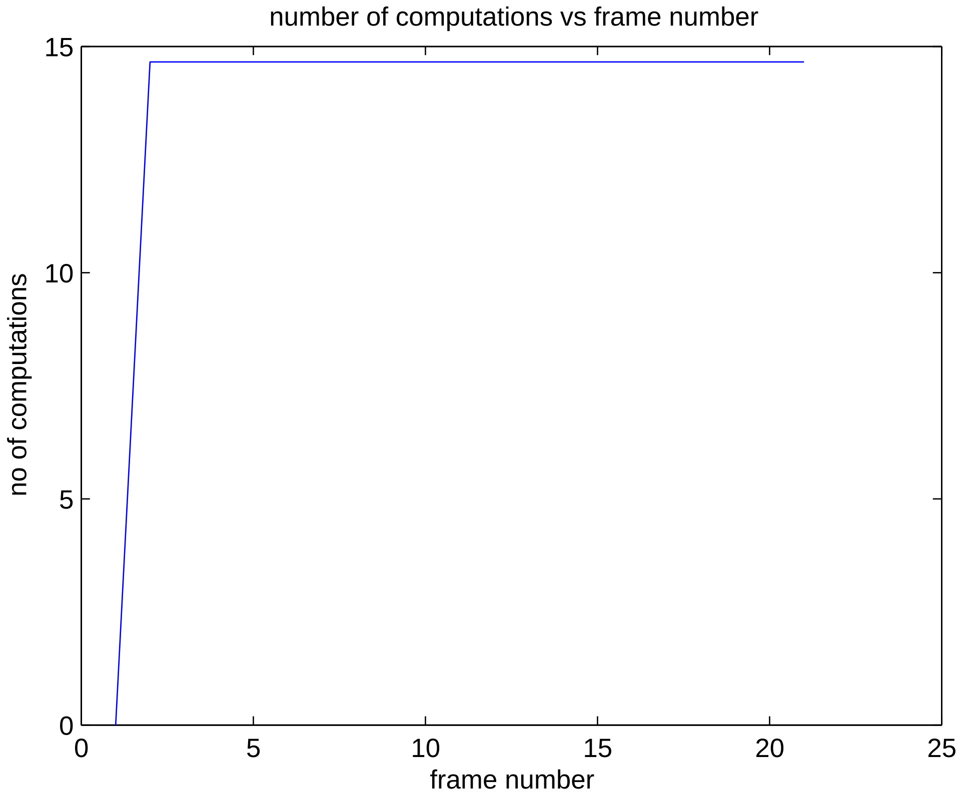


Figure 5.4: Three Step Search: Computation Time

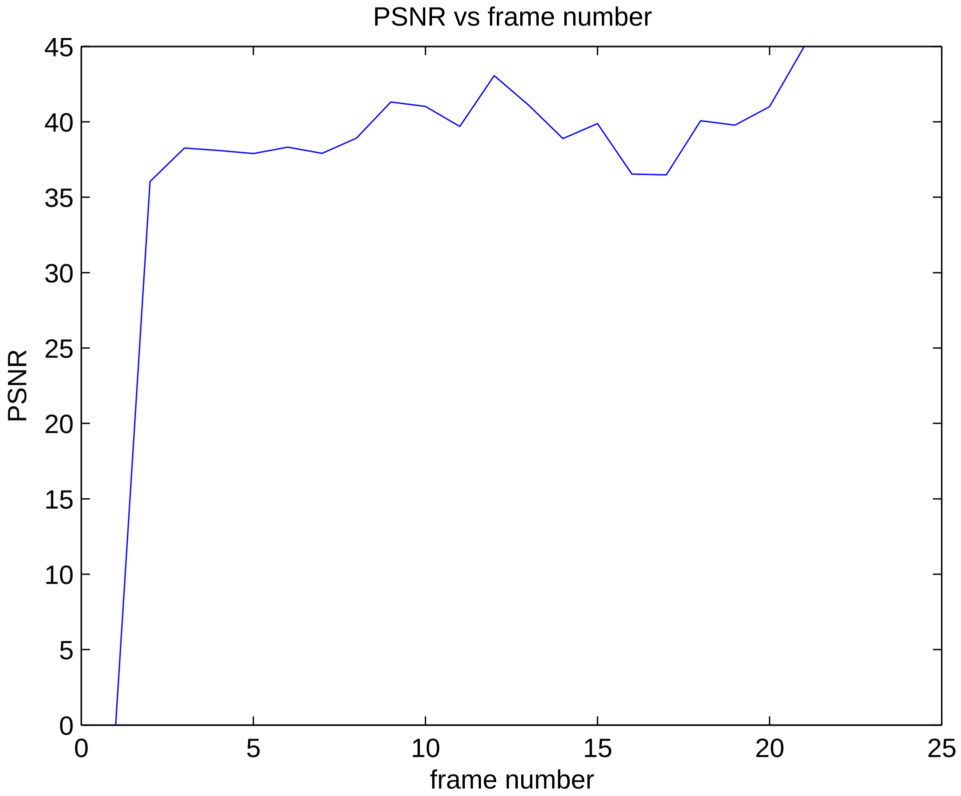


Figure 5.5: Three Step Search:PSNR

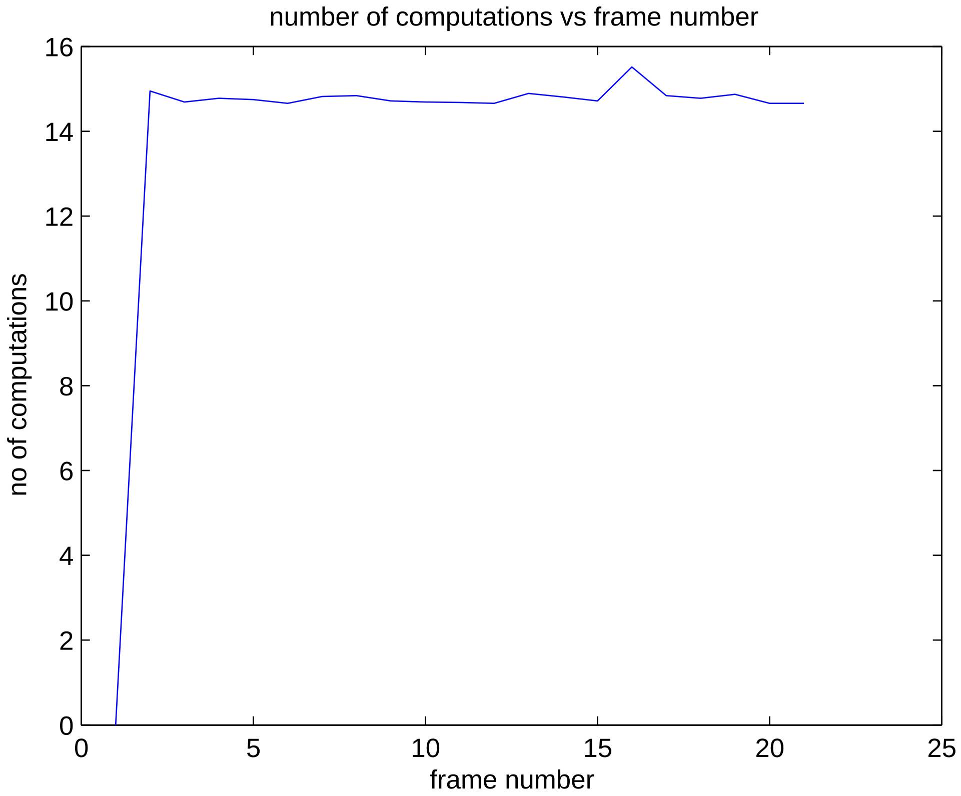


Figure 5.6: Window Size 3, Four Step Search: Computation time

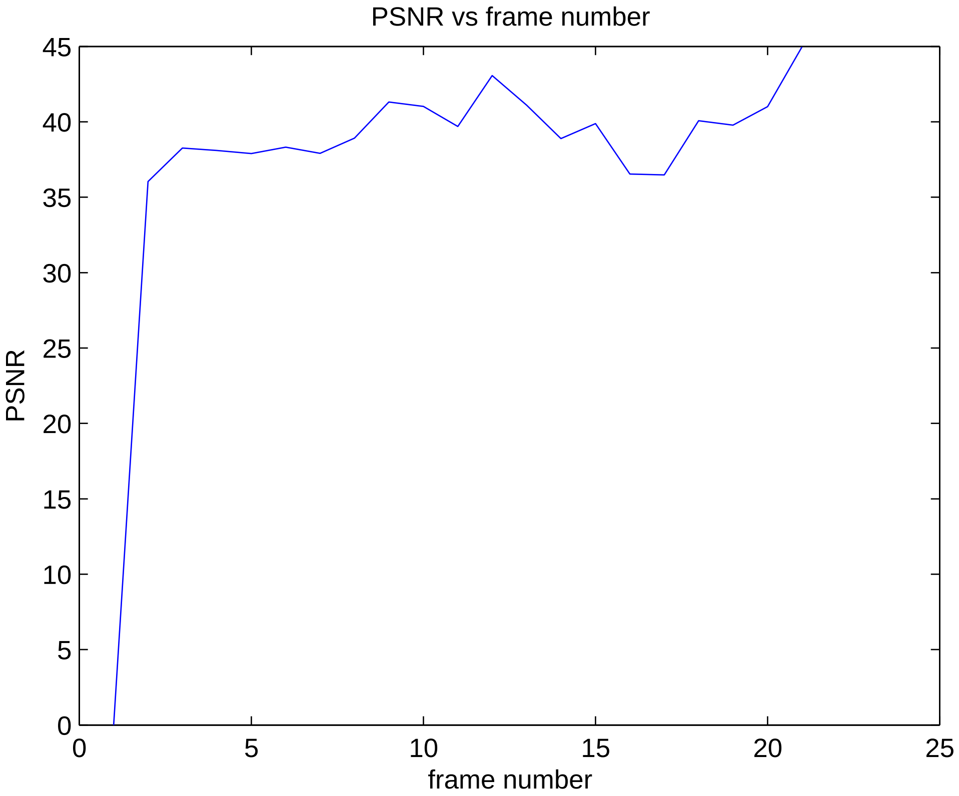


Figure 5.7: Four Step Search:PSNR

Window Size = 7

Extensive Search:

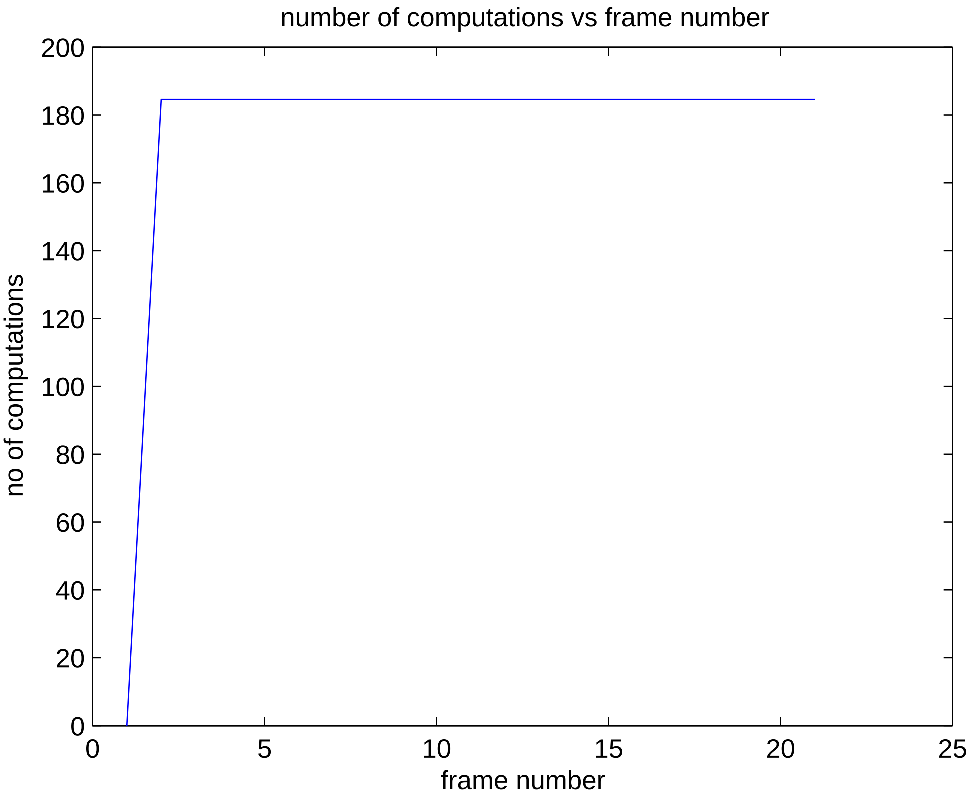


Figure 5.8: Extensive Search:Computation Time

5.3 Conclusion

Initially we have examined the actual mechanism of compressing images by implementing the already existing JPEG compression technique.Then we have shifted our focus to video compression. The past two decades have seen the growth of wide acceptance of multimedia. Video compression plays an important role in archival of entertainment based video (CD/DVD)as well as real-time reconnaissance / video conferencing applications. While ISO MPEG sets the standard for the former types of application, ITU sets the standards for latter low bit rate applications. In the entire motion based video compression process

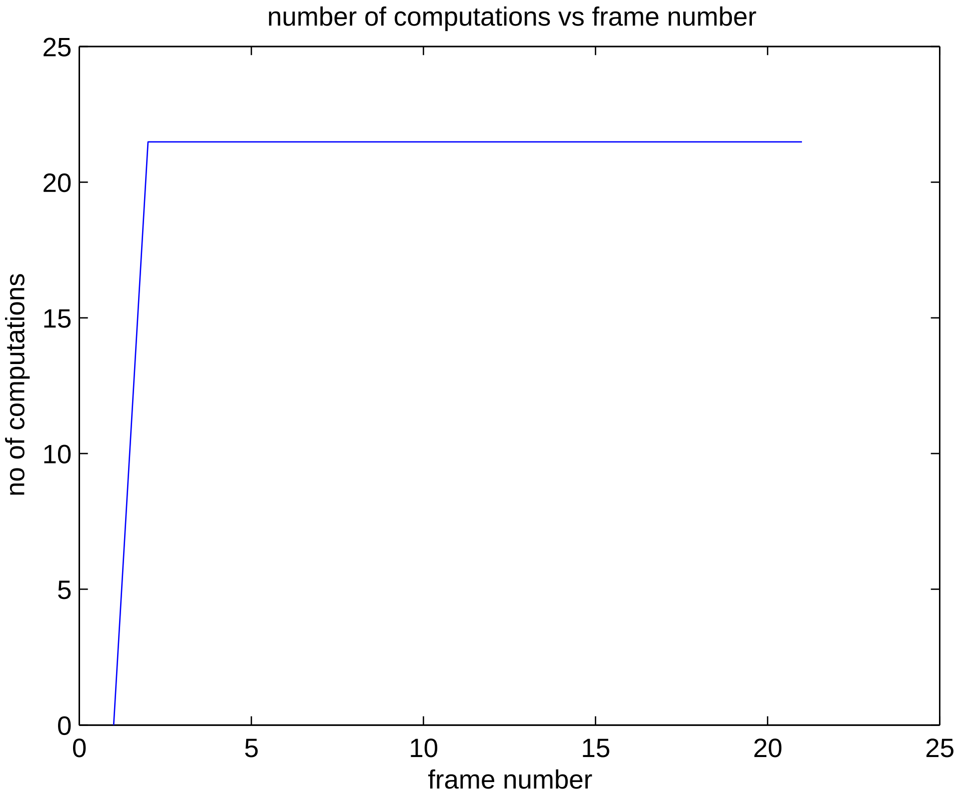


Figure 5.9: Three Step Search: Computation Time

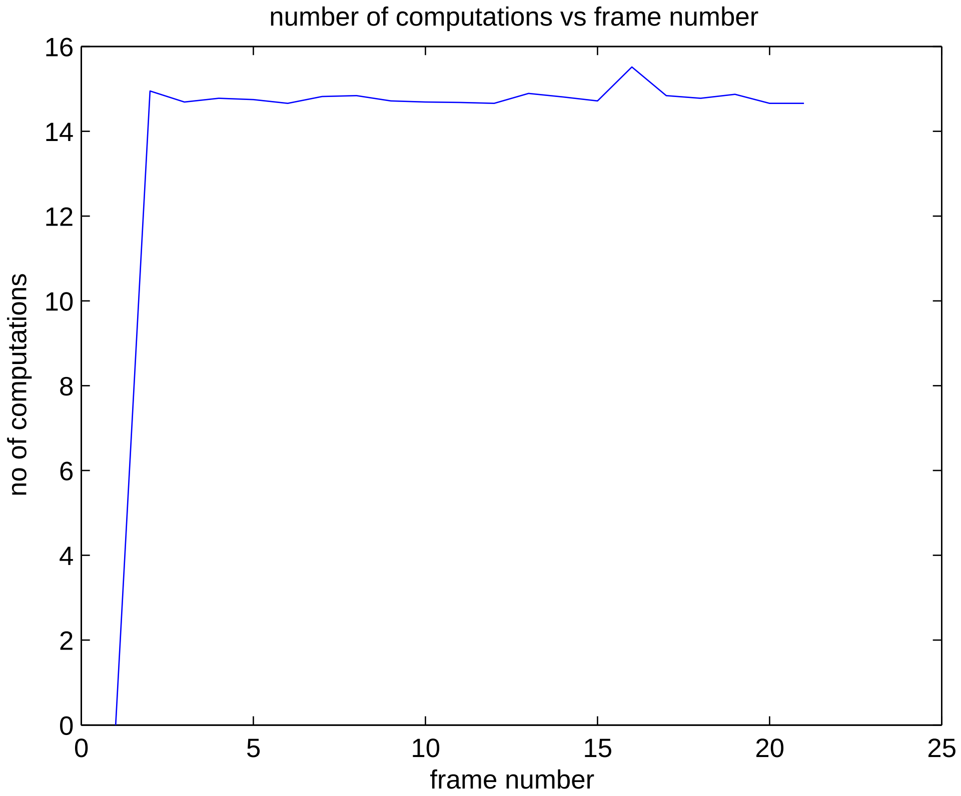


Figure 5.10: Four Step Search: Computation time

motion estimation is the most computationally expensive and time-consuming process.The research in the past decade has focused on reducing both of these side effects of motion estimation. Block matching techniques are the most popular and efficient of the various motion estimation techniques.Here we have first described the motion compensation based video compression in brief.Then we have illustrated and simulated 3 of the most popular block matching algorithms, with their comparative study at the end. We have also aimed at providing a novel approach to video compression using BWT. The BWT is a block sorting algorithm which gives excellent results in text and image compression. The BWT and MTF(Move To Front) encoding are just transformations and they do not perform any compression. The BWT takes advantage from reparative structure with the help of MTF and produces better compression with the help of entropy coding. In this paper we have proposed a novel video compression scheme using BWT. In a video sample the adjacent frames are strongly correlated, so BWT utilises this property from the correlated video sequences.

Bibliography

[1] A.Sadka. *Compressed Video Communications.* John Wiley and Sons, 2002.

[2] C.Wooton. *A Practical Guide to Video and Audio Compression.* Elsevier, 2005.

[3] JPEG JFIF.

[4] M. Burrows and D. J. Wheeler. A block sorting lossless data compression algorithm. *SRC Research Report 124*, May 1994.

[5] Aroh Barjatya. Block matching algorithms for motion estimation. *DIP 6620 Spring*, 2004.

[6] Lai-Man Po and Wing-Chung Ma. A novel four-step search algorithm for fast block motion estimation. *IEEE Trans. Circuits And Systems For Video Technology*, 6(3):313 – 317, June 1996.