CHAPTER - 3

VIDEO COMPRESSION TECHNIQUES

3.1 Introduction

This conducted research presents a survey and study of various image compression techniques primarily, The Principal Component Analysis approach. The PCA approach can be implemented in two ways namely, PCA Statistical Approach and PCA Neural Network Approach. It also deals with various benefits of using image compression methods.

The following subsection explores the various elements of the video compression

3.1.1 Image compression

Image compression addresses the problem of reducing the amount of data required to represent a digital image. It is a process intended to yield a compact representation of an image, thereby reducing the image storage/transmission requirements. Image can be defined as a two dimensional signal processed by the human visual system. The signals which are in analog form need to be converted to digital form in order to process, store and transmit. A digital image is a two dimensional array of pixels and each pixel is a digital value.

Data may be represented as an image which conveys more meaning. Visualizing objects is more suitable and easy to observe, easy to access, searchable hence the need to discover new methods for efficient storage and quick transmission of image data.

Images from the significant part of data, particularly in remote sensing, biomedical and video conferencing applications. The use of and dependence on information and computers continue to grow, so too does our need for efficient ways of storing and transmitting large amounts of data. Compression is achieved by the removal of one or more of the three basic types of data redundancies:

- Coding Redundancy
- Interpixel Redundancy

- Psycho visual Redundancy

Coding redundancy is present when less than optimal code words are used. Interpixel redundancy results from correlations between the pixels of an image. Psycho visual redundancy is due to data that is ignored by the human visual system (i.e. visually superfluous information).

Image compression methods reduce the number of bits required to represent an image by taking advantage of these redundancies. An inverse process called decompression (decoding) is applied to the compressed data to get back the reconstructed image. The objective of compression is to reduce the number of bits as much as possible, while keeping the resolution and the visual quality of reconstructed image as close to the original image as possible. Image compression systems are composed of two distinct structural blocks: an encoder and a decoder.

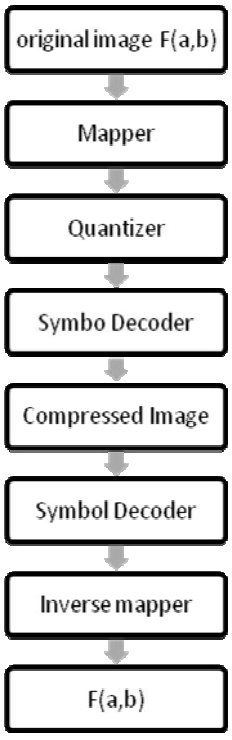


Figure 3.1. Decompressed image (Original image)

Image f(a,b) is fed into the encoder, which creates a set of symbols from the input data and uses them to represent the image. If we let n1 and n2 denote the number of information carrying units (usually bits) in the original and encoded images respectively, the compression that is achieved can be quantified numerically via the Compression Ratio,

CR = n1 /n2

As shown in the figure 3.1., the encoder is responsible for reducing the coding, Interpixel and psycho visual redundancies of input image. In first stage, the mapper transforms the input image into a format

3.1.2 Benefits of Compression

- It provides a potential cost saving associated with sending less data over switched telephone networks where cost of call is based upon its duration.
- It reduces storage requirements

3.1.3 Degree of Compression

How Much Compression?

Compression performance is often specified by giving the ratio of input data to output data for the compression process (the compression ration). This measure is a dangerous one unless you are careful to specify the input data format in a way that is truly comparable to the output data format. For example, the compressor might have used a 512 × 480, 24 bits-per-pixel (bpp) image as the input to the compression process, which then delivered a bit stream of 15,000 bytes. In that case, the input data was 737,280 bytes, and this would give a compression ratio 727,280 /15,000=49. However the output display has only 256 × 240 pixels, so we achieved 4:1 of that compression by reducing the resolution. Therefore, the compression ratio with equal input and output resolutions is more like 12:1. A similar argument can be made for the

bpp relationship between input and output - the output quality may not be anything near 24 bpp.

A much better way to specify the amount of compression is to determine the number of bits per displayed pixel needed in the compressed bit stream. For example, if we are reproducing a 256 × 240 pixel image from a 15,000-byte bit stream, we are compressing to (bits) / (pixels) (15,000 × 8) / (256 × 240) = 2 bits per pixel

3.2 Image Compression Methods

The image compression methods are broadly classified into two categories depending whether or not an exact replica of the original image could be reconstructed using the compressed image.

How fast does it compress or decompress?

In many applications, compression and decompression will be done at different times; they may even be done with totally different systems at different locations. The reason for this is that there is usually storage or transmission of the image in between the two processes storage or transmission is why we need compression in the first place. Therefore, we must evaluate the speeds of compression or decompression separately.

In most cases of storing still images, compression speed is less critical than decompression speed – since we are compressing the image ahead of time to store it, we can usually take our time in that process. On the other hand, decompression usually takes place while the user is waiting for the result and speed is much more important. With motion video compression there is a need for fast compression in order to capture motion video in real time as it comes from a camera or VCR. In any case, compression and decompression speed is usually easy to specify and measure.

These are:

- Lossless method
- Lossy method

3.2.1 Lossless Compression Method

In lossless compression methods, the original image can be perfectly recovered from the compressed (encoded) image. These are also called noiseless since they do not add noise to the signal (image). It is also known as entropy coding since it uses statistics or decomposition methods to eliminate or minimize redundancy. Lossless compression is used only for a few applications with stringent requirements such as medical imaging.

Following methods are included in lossless compression:

- Run length encoding
- Huffman encoding
- LZW coding
- Area coding

3.2.1.1 Run Length Encoding

This is a very simple compression procedure used for sequential data. It is very useful in case of redundant data.

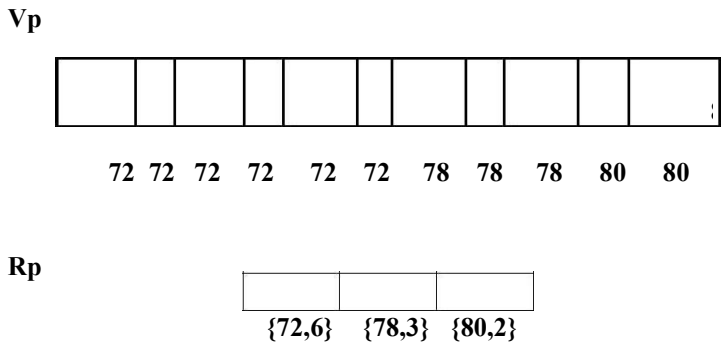


Figure3.2: Run –Length Encoding

This method replaces sequences of identical symbols (pixels), called runs by shorter symbols [MN2005]. The run length code for a gray scale image is represented by a sequence {Vp, Rp} where Vp is the intensity of pixel and Rp refers to the number of consecutive pixels with the intensity Vp as shown in the figure3.2.

3.2.1.2 Huffman Encoding

This is a common method for coding symbols based on their statistical occurrence of frequencies (probability). The pixels in the image are treated as symbols. The symbols that occur more frequently are assigned a smaller number of bits, while the symbols that occur less frequently are assigned a relatively larger number of bits. Huffman code is a prefix code. This means that the (binary) code of any symbol is not the prefix of the code of any other symbol. Most image coding standards use lossy methods in the earlier stages of compression and use Huffman coding as the last step.

3.2.1.3 LZW Coding

It is a universal lossless data compression algorithm created by Abraham Lempel, Jacob Ziv, and Terry Welch. LZW is a dictionary based coding which can be static or dynamic. In static dictionary coding dictionary is fixed during the encoding and decoding processes. In dynamic dictionary coding, the dictionary is updated on fly. LZW is widely used in computer industries and it is used as a compress command in UNIX platform.

3.2.1.4 Area Coding

Area coding is an enhanced version of run length coding, reflecting the two dimensional character of images. This is an important and advance over the other lossless methods. For coding an image it does not make too much sense to interpret it as a sequential stream, as it is in fact an array of sequences, building up a two dimensional object. The algorithms for area

coding try to find rectangular regions with the same characteristics. These regions are coded in a descriptive form as an element with two points and a certain structure. This type of coding can be highly effective but it bears the problem of a nonlinear method, which cannot be implemented in hardware. Therefore, the performance in terms of compression time is not competitive, although the compression ratio is.

3.3 Lossy Compression Methods

Lossy schemes provide much higher compression ratios than lossless schemes. Lossy schemes are widely used since the quality of the reconstructed images is adequate for most applications. Here the decompressed image is not identical to the original image, but reasonably close to it. Figure 3.3.

Here "Prediction – transformation – decomposition process is completely reversible. The quantization process results in loss of information. The entropy coding after the quantization step, however, is lossless. The decoding is a reverse process. Firstly, entropy decoding is applied to compressed data to get the quantized data. Secondly, dequantization is applied to it & finally the inverse transformation to get the reconstructed image.

Major performance considerations of a lossy compression scheme include:

- Compression ratio
- Signal - to - noise ratio
- Speed of encoding & decoding

Lossy compression methods include following schemes:

- Transformation coding
- Vector quantization
- Fractal coding

- Block Truncation Coding
- Sub band coding

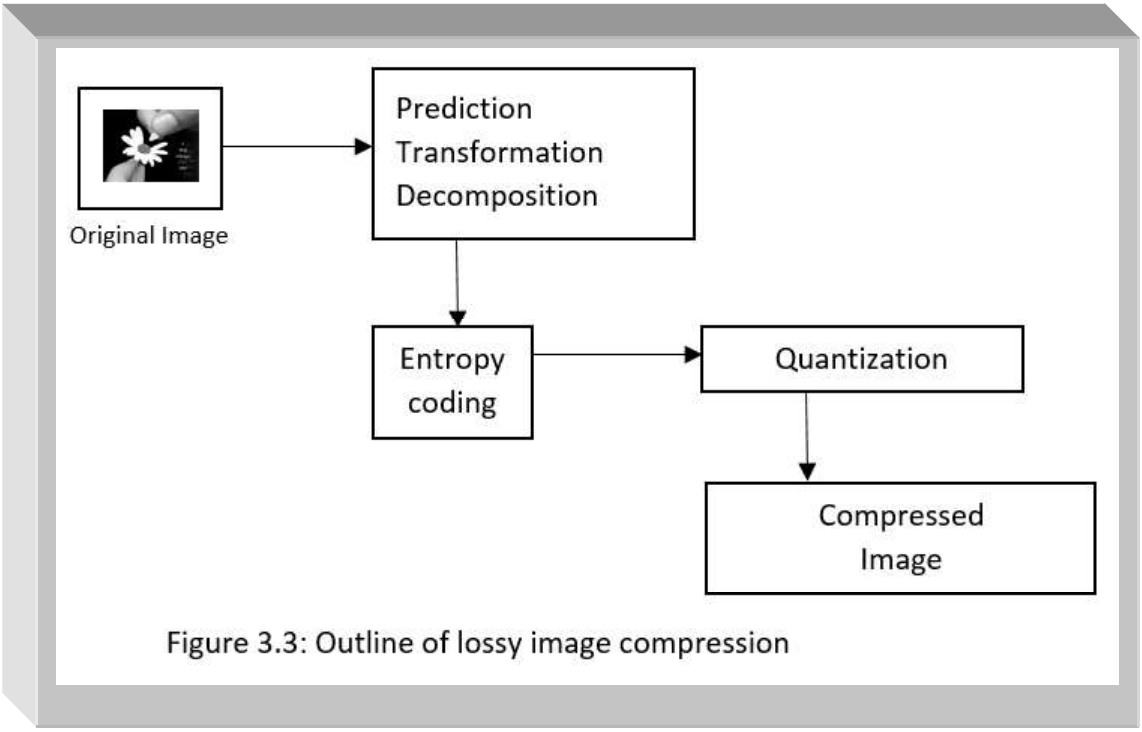


Figure 3.3: Outline of lossy image compression

3.3.1 Transformation Coding

In this coding scheme, transforms such as DFT (Discrete Fourier Transform) and DCT (Discrete Cosine Transform) are used to change the pixels in the original image into frequency domain coefficients (called transform coefficients). [MK,2004] These coefficients have several desirable properties. One is the energy compaction property that results in most of the energy of the original data being concentrated in only a few of the significant transform coefficients. This is the basis of achieving the compression. Only those few significant coefficients are selected and the remaining is discarded. The selected coefficients are considered for further quantization and entropy encoding. DCT coding has been the most common approach to transform coding. It is also adopted in the JPEG image compression standard.

3.3.2 Vector Quantization

The basic idea in this method is to develop a dictionary of fixed- size vectors, called code vectors. A vector is usually a block of pixel values. A given image is then partitioned into non-overlapping blocks (vectors) called image vectors then for each value in the dictionary is determined and its index in the dictionary is used as the encoding of the original image vector. Thus, each image is represented by a sequence of indices that can be further entropy coded.

3.3.3 Fractal Coding

The essential idea here is to decompose the image into segments by using standard image processing methods such as color separation, edge detection, and spectrum and texture analysis. Then each segment is looked up in a library of fractals. The library actually contains codes called iterated function system (IFS) codes, which are compact sets of numbers. Using a systematic procedure, a set of codes for a given image are determined, such that when the IFS codes are applied to a suitable set of image blocks yield an image that is a very close approximation of the original. This scheme is highly effective for compressing images that have good regularity and self-similarity.

3.3.4 Block Truncation Coding

In this scheme, the image is divided into non overlapping image of (M X N) dimension.

A = [Ø 1 Ø2… ……ØM] blocks of pixels. For each block, threshold and reconstruction values are determined. The threshold is usually the mean of the pixel values in the block. Then a bitmap of the block is derived by replacing all pixels whose values are greater than or equal (less than) to the threshold by a 1 or 0. Then for each segment (group of 1s and 0s) in the bitmap, the reconstruction value is determined. This is the average of the values of the corresponding pixels in the original block.

3.3.5 Sub Band Coding

In this method, the image is analyzed to produce the components containing frequencies in well-defined bands, and sub bands. Subsequently, quantization and coding is applied to each of the bands. The advantage of this method is that the quantization and coding well suited for each of the sub bands can be designed separately.

3.4 PCA (Principal Component Analysis)

In statistics, PCA is a method for simplifying a dataset by reducing multidimensional datasets to lower dimensions for analysis. PCA is a standard method commonly used for data reduction in statistical pattern recognition and signal processing. PCA is noted to be one of the most valuable results from applied linear algebra. It is used abundantly in all forms of analysis from neuroscience to computer graphics, because it is a simple non - parametric method of extracting relevant information from confusing datasets.

PCA is also called the KARHUNEN-LOEVE Transform (KLT, named after Kari Karhunen & Michel Loeve) or the HOTELLING Transform. Its general objectives are:

- Data reduction
- Interpretation

There are basically two approaches for performing PCA. They are Classical Statistical method and Artificial Neural Network method.

3.4.1 PCA Classical Statistical Method

It involves finding Eigen values and corresponding Eigen vectors of the data set using covariance matrix. [SZ+2005] The corresponding Eigen values of the matrix gives an indication of amount of information the respective principal components represent. The methodology for calculating principal component is given by the following procedure.

Let X1, X2,..., Xm be the sub images of dimension N. The corresponding algorithm is described as follows:

1. Computation of the global mean (X) from sub images.
2. Subtraction of the mean from each sub image to generate the mean removed image.

   Øi = Xi - X
3. Formation of the matrix using mean removed sub
4. Computation of the sample covariance matrix (C) of dimension (N x N) .
5. Computation of the Eigen values of the covariance matrix. Computation of Eigen values is performed by Jacobian iteration method. C : λ1 > λ2 > …… >λN
6. Computation of the Eigen vectors for the Eigen values C: u1 , u2, ……..,uN
7. Dimensionality reduction step.

Keep only the Eigen vectors corresponding to K largest Eigen values. These Eigen values are called as “principal components”. The above said steps are needed to generate the principal components of the image.

Corresponding Eigen vectors are uncorrelated and have greater variance. In order to avoid the components that have an undue influence on the analysis, the components are usually coded with mean as zero and variance as one. This standardization of the measurement ensures that they all have equal weight in the analysis.

3.4.2 PCA Neural Network

Artificial neural network are models that attempt to achieve performance via dense inter connection of simple computational elements. The most important property of a Neural Network is the ability to learn from its environment. PCA is a powerful linear block transform coding in which, an image is subdivided into non - overlapping blocks of N×N pixels which

can be considered as N-Dimensional vectors with N = n × n. A linear Transformation, which can be written as an M ×N – dimensional matrix W with M ≤ N, is performed on each block with the M rows of W, wi being the basis vectors of the transformation. An Adaptive Principal Component Extraction (APEX) is used to de-correlate the principal components. The main difference between this APEX architecture and the existing PCA networks lies in the additional lateral connections at the outputs of the network.

3.4.3 Applications of PCA in Computer Vision Representation

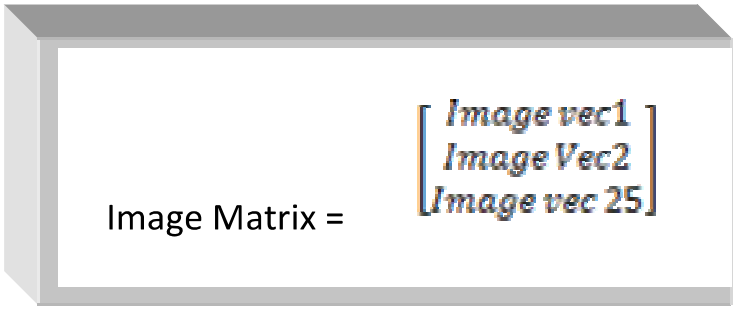
When using these sorts of matrix methods in computer vision, representation of images should be considered. [8] A square, N by N image can be expressed as an N2 – Dimensional vector.

X = (x1 x2 x3 ……xN2)

Where, the rows of pixels in the image are placed one after the other to form a one dimensional image. Eg. The first N elements x1 – xN will be the first row of the image, the next N elements are the next row, and soon. The values in the vector are the intensity values of the image, possibly a single gray scale value.

3.4.4 PCA to find patterns

Say we have 25 images with each image is N pixels high by N pixels wide. For each image we can create an image vector as described in the representation section. All these images can be put together in one big image-matrix as given below.



It works much better for recognizing faces, because the PCA analysis has given the original images in terms of the differences and similarities between them. The PCA analysis has identified the statistical patterns in the data.

3.4.5 PCA for Image Compression

If we have 25 images each with N2 vectors and 25 dimensions. Now, PCA can be implemented on this set of data. 25 Eigen vectors will be obtained because each vector is 25 –dimensional. To compress the data, choose the data using only 20 Eigen vectors. This gives a final data set with only 20 dimensions, which has saved one fifth of the space. However, when the original data is reproduced, the images have lost some of the information. This compression is said to be lossy because the decompressed image is not exactly the same as the original.

3.5 Introduction to Video Compression

Over the past decades, video compression technologies have become an integral part of the way we create, communicate and consume visual information. Digital video communication can be found today in many applications such as broadcast services over satellite and terrestrial channels, digital video storage, wires and wireless conversational services and etc. The data quantity is very large for the digital video and the memory of the storage devices and the bandwidth of the transmission channel are not infinite, so reducing the amount of data needed to reproduce video saves storage space, increases access speed and is the only way to achieve motion video on digital computers. The video contains much spatial and temporal redundancy. In a single frame, nearby pixels are often correlated with each other. This is called spatial redundancy, or the intra-frame correlation. Another one is temporal redundancy, which means adjacent frames are highly correlated, or called the inter-frame correlation.

Therefore, our goal is to efficiently reduce spatial and temporal redundancy to achieve video compression [A2013][OJ+2013], [KH+2015].

Video clips are made up of sequences of individual images, or "frames". Therefore, video compression algorithms share many concepts and techniques with still image compression algorithms, such as JPEG. In fact, one way to compress video is to ignore the similarities between consecutive video frames, and simply compress each frame independently of other frames. For example, some products employ this approach to compress video streams using the JPEG still-image compression standard [LZ+2011], [BD+2014],[TD+2102]. This approach, known as "motion JPEG" or MJPEG is sometimes used in video production applications. Although modern video compression algorithms go beyond still-image compression schemes and take advantage of the correlation between consecutive video frames using motion estimation and motion compensation, these more advanced algorithms also employ techniques used in still-image compression algorithms. Therefore, we begin our exploration of video compression by discussing the inner workings of transform-based still-image compression algorithms such as JPEG [LJ+2012].

3.5.1 Need of Video Compression

Compression Technology is employed to efficiently use storage space, to save on transmission capacity and transmission time, respectively. Basically, it is all about saving resources and money. Despite of the overwhelming advances in the areas of storage media and transmission networks it is actually quite a surprise that still compression technology is required. One important reason is that also the resolution and amount of digital data has increased (e.g. HD-TV resolution, ever-increasing sensor sizes in consumer cameras), and that there are still application areas where resources are limited, e.g. wireless networks. Apart from the aim of

simply reducing the amount of data, standards like MPEG-4, MPEG-7, and MPEG-21 offer additional functionalities [DI+2012]. During the last years three important trends have contributed to the fact that nowadays compression technology is as important as it has never been before – this development has already changed the way the user work with multimedia data like text, speech, audio, images, and video which will lead to new products and applications [WR-10].

1. The availability of highly effective methods for compressing various types of data.
2. The availability of fast and cheap hardware components to conduct compression on single-chip systems, microprocessors, DSPs and VLSI systems.
3. Convergence of computer, communication, consumer electronics, publishing, and entertainment industries.

The high bit rates that result from the various types of digital video make their transmission through their intended channels very difficult. Even entertainment video with modest frame rates and dimensions would require bandwidth and storage space far in excess of that available from CD-ROM. Thus delivering consumer quality video on compact disc would be impossible. This is analogous to an envelope being too large to fit into a letterbox. Similarly the data transfer rate required by a video telephony system is far greater than the bandwidth available over the plain old telephone system (POTS). Even if high bandwidth technology (e.g. fibre-optic cable) was in place, the per-byte-cost of transmission would have to be very low before it would be feasible to use it for the staggering amounts of data required by HDTV. Finally, even if the storage and transportation problems of digital video were overcome, the processing power needed to manage such volumes of data would make the receiver hardware very expensive [MS+2012][HH+2014][GY+2014].

Although significant gains in storage, transmission, and processor technology have been achieved in recent years, it is primarily the reduction of the amount of data that needs to be stored, transmitted, and processed that has made widespread use of digital video a possibility. This reduction of bandwidth has been made possible by advances in compression technology. Advances in compression technology more than anything else have led to the arrival of video to the desktop and hundreds of channels to the home. Compression reduces the bandwidth required to transmit and store digital video [WR-11].

3.5.2 Video Compression Techniques

A great deal of research has been done in image and video compression technology, going back more than 25 years. The references at the end of this chapter will lead you into the voluminous literature of this research. Many powerful techniques have been developed, simulated, and fully characterized in the literature; in fact, today it is quite difficult to invent something new in this field – it has been so well researched.

However, broad application of the more sophisticated video compression approaches has not been practical because of the cost of the hardware required. That is now changing because of the power of high-performance digital signal processing chips and custom VLSI devices.

We will use the word technique to refer to a single method of compression–usable by itself, but possible also used in combination with other techniques. On the other hand, an algorithm refers to the collection of all the techniques used by any particular video compression system.

We will assume that the input to the compression system is always a PCM digitized signal in color component (RGB, YUV, etc.)form. Most compression systems will deal with the color components separately, processing each on by itself. In decompression, the components similarly are separately recovered and then combined into the appropriate display format after

decompression. Note, however, that there is nothing that requires that the individual color components be processed in the same way during compression and decompression–in fact, there are sometimes significant advantages to handling the components of the same image by different techniques. This brings up immediately that we must choose the color component format to use, and that choice could make a big difference in what performance is achieved by the system. Two obvious choices that we have already discussed are RGB components or luminance/chrominance components. Where it is relevant, the significance of the color component choice will also be covered.

Similarly, there is always a possibility of making any technique adaptive which means that the technique can change as a function of the image content. Adaptively is not a compression technique itself; rather, it is a way to cause any given technique to be more optimized locally in the image or temporally in the frame sequence. Almost all the compression techniques we will be discussing can be made adaptive, but of course this adds complexity. Where adaptively is an important aspect, it will be discussed with each technique.

The output of a compression process is a bit stream–it is usually no longer a bitmap and individual pixels may not be recognizable. The structure of the bit stream is important, however, because it can also affect the compression efficiency and the behaviour of the system when errors occur in bit assignment–this is where the bit stream structure is imposed on the compressed data. It may be a task which is subsumed in the algorithm, or it may be a separate step in the process.

3.5.3 Redundancy and Visibility

Redundancy in a digital video image occurs when the same information is transmitted more than once. For example:

- In any area is the picture where the same color spans more than one pixel location, there is redundancy between pixels, since adjacent pixels will have the same value. This applies both horizontally and vertically.
- When the scene or part of the scene contains predominantly vertically oriented objects, there is a possibility that two adjacent lines will be partially or completely the same, giving us redundancy between lines. These two types of redundancy (pixel and line) exist in any image and are called spatial redundancy.
- When a scene is stationary or only slightly moving, there is a further possibility or redundancy between frames of a motion sequence – adjacent frames in time are similar, or they may be related by a simple function such as translation. This kind of redundancy is called temporal redundancy.

Compression schemes may exploit any or all of these aspects of redundancy. Another consideration in image reproduction is that we do not need to display more information than our viewer will be able to see. The best example of this arises from the human eye's poor special acuity for certain colors. Because of this characteristic of the viewer, it is not necessary to provide independent color values for every pixel. The color information can be transmitted at lower resolution than the luminance or black and white portion of the image. This principle, called color sub sampling, is used successfully in NTSC and PAL color television, and it can be implemented in a digital video system as well.

Lossless compression techniques, as their name implies, involve no loss of information. If data has been losslessly compressed, the original data can be recovered exactly from the compressed data after a compress/expand cycle. Lossless compression is generally used for so-called "discrete" data, such as database records, spread sheets, word-processing files, and

even some kinds of image and video information. Lossy compression works very differently. These programs simply eliminate "unnecessary" bits of information, tailoring the file so that it is smaller. This type of compression is used a lot for reducing the file size of bitmap pictures, which tend to be fairly bulky [TK 2011].

Few lossless video formats are in common consumer use, as they would result in video files taking up a huge amount of space. Common formats like H.264, MKV, and WMV are all lossy. H.264 can provide smaller files with higher qualities than previous generations of video codecs because it has a “smarter” algorithm that’s better at choosing the data to throw out. Some of these lossless formats also provide compression. For example, a WAV file is a completely uncompressed audio file, and takes up quite a bit of space. FLAC and ALAC are both lossless types of audio files that contain the same data as a WAV file, but they use a form of compression to create smaller files. Formats like FLAC and ALAC don’t throw any data away – they keep all the data and compress it intelligently, like ZIP files do. However, they are still significantly larger in size than MP3 files, which throw much data away.

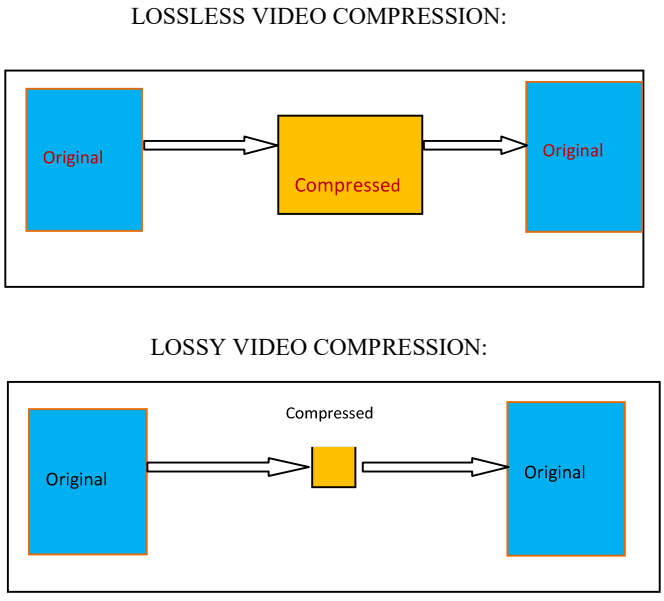


Figure 3.4**:** Lossless vs Lossy Video Compression

Lossy compression is best used to reduce the size of video data, where defects in the picture/video can be hidden as long as the general structure of the picture/video remains intact. This type of compression is called lossy compression because part of the reason that it compresses so well is that actual data from the video image, is lost and then replaced with some approximation. The typical model for lossless and lossy compression is shown in Figure-3.4

The image compression techniques used in JPEG and in most video compression algorithms are "lossy." That is, the original uncompressed image can't be perfectly reconstructed from the compressed data, so some information from the original image is lost. Lossy compression algorithms attempt to ensure that the differences between the original uncompressed image and the reconstructed image are not perceptible to the human eye [EM+2012],[WR-12].

The first step in JPEG and similar image compression algorithms is to divide the image into small blocks and transform each block into a frequency-domain representation. Typically, this step uses a discrete cosine transform (DCT) on blocks that are eight pixels wide by eight pixels high. Thus, the DCT operates on 64 input pixels and yields 64 frequency-domain coefficients. The DCT itself preserves all of the information in the eight-by-eight image block. That is, an inverse DCT (IDCT) can be used to perfectly reconstruct the original 64 pixels from the DCT coefficients. However, the human eye is more sensitive to the information contained in DCT coefficients that represent low frequencies (corresponding to large features in the image) than to the information contained in DCT coefficients that represent high frequencies (corresponding to small features). Therefore, the DCT helps separate the more perceptually significant information from less perceptually significant information. Later steps in the compression algorithm encode the low-frequency DCT coefficients with high precision,

but use fewer or no bits to encode the high-frequency coefficients, thus discarding information that is less perceptually significant. In the decoding algorithm, an IDCT transforms the imperfectly coded coefficients back into an 8x8 block of pixels.

The computations performed in the IDCT are nearly identical to those performed in the DCT, so these two functions have very similar processing requirements. A single two-dimensional eight-by-eight DCT or IDCT requires a few hundred instruction cycles on a typical DSP. However, video compression algorithms must often perform a vast number of DCTs and/or IDCTs per second. For example, an MPEG-4 video decoder operating at CIF (352x288) resolution and a frame rate of 30 fps may need to perform as many as 71,280 IDCTs per second, depending on the video content. The IDCT function would require over 40 MHz on a Texas Instruments TMS320C55x DSP processor (without the DCT accelerator) under these conditions. IDCT computation can take up as much as 30% of the cycles spent in a video decoder implementation.

Because the DCT and IDCT operate on small image blocks, the memory requirements of these functions are rather small and are typically negligible compared to the size of frame buffers and other data in image and video compression applications. The high computational demand and small memory footprint of the DCT and IDCT functions make them ideal candidates for implementation using dedicated hardware coprocessors [VG+2012], [RI 2011], [HP+2012], [RJ+2013]. Some of the exiting algorithms for lossless video compression:

CCITT group 3 & 4 compression

- ✓ Flate/deflate compression
- ✓ Huffman compression
- ✓ LZW compression

- ✓ RLE compression

Some of the exiting algorithms for lossy video compression:

- ✓ Motion JPEG
- ✓ H.264/MPEG-4 AVC
- ✓ Ogg Theora
- ✓ Dirac
- ✓ Sorenson video codec
- ✓ VC-1
- ✓ H.265/HEVC

3.6 Introduction to Surveillance System Videos

If the CCTV camera signals are digitized, then a digital stream with more than 150 Mbps will be interpreted, but such a stream contains a lot of redundant information. All image compression algorithms are divided in two groups namely: (i) lossless, and (ii) lossy. Most CCTV compression algorithms used in CCTV are lossy compressions, because such algorithms offer higher compression ratio (the ratio of the resulting video file size compared with the original file size). Video compression algorithms are divided in two groups: (i) Frame based compression (JPEG, Wavelet, JPEG 2000) (ii) Stream based compression (MPEG-2, MPEG-4, H.264, MPEG-7). Usage of stream based compression algorithms enables greater savings on storage space and network bandwidth but as a trade-off these algorithms require higher computing performance.

Four of the most common type of compressions widely used in CCTV are:

- ➢ Motion JPEG
- ➢ MPEG-4

- H.264
- JPEG2000.

Motion JPEG is very popular compression format. MJPEG fits very well for video archives because of its frame based nature. MPEG4 can be 3 times more efficient in terms of compression ratio in comparison toMotion JPEG. But MPEG4 is a bad choice for systems with frame rate less than 5-6 frames per second. H.264 can be 50-100% more efficient in comparison toMPEG-4. The MPEG-4 and H.264 are ideal for CCTV systems with limited but stable bandwidth. JPEG2000 is similar to JPEG, but uses wavelet transform instead of discrete-cosine-transform (DCT) of JPEG. JPEG2000 offers a better image quality on higher compression levels. Another great advantage is a possibility to decompress lower resolution representation of the image. This feature is good for motion detection algorithms. However JPEG2000 compression needs way higher CPU performance, than JPEG.

With the advent of IP Security Camera Systems and its increased popularity over Analogue CCTV Systems, it has now become inevitable for the Camera Manufacturers/System Integrators to focus more on the data compression to maintain the quality of image as well as save on transmission bandwidth and hard disc storage space. An application which requires live monitoring will need signal streams high quality image, whereas for recording purpose or even for viewing in Mobile from a remote location, different streams are required with different compression techniques. There are DVR's in the market with even five separate streams for five different kinds of applications[WR-7,9], [OG 2013].

At its most basic level, compression is performed when an input video stream is analyzed and information that is indiscernible to the viewer is discarded. Each event is then assigned a code commonly occurring events are assigned few bits and rare events will have codes with more

bits. These steps are commonly called signal analysis, quantization and variable length encoding respectively [HH+2014]. There are four methods for compression, discrete cosine transform (DCT), vector quantization (VQ), fractal compression, and discrete wavelet transform (DWT).

Discrete cosine transform is a lossy compression algorithm that samples an image at regular intervals, analyzes the frequency components present in the sample, and discards those frequencies which do not affect the image as the human eye perceives it. DCT is the basis of standards such as JPEG, MPEG and H.264 [LL+2013], [ZA+2014], [LJ+2012], [VS+2014]. Vector quantization is a lossy compression that looks at an array of data, instead of individual values. It can then generalize what it sees, compressing redundant data, while at the same time retaining the desired object or data stream's original intent. Fractal compression is a form of VQ and is also a lossy compression. Compression is performed by locating self-similar sections of an image, then using a fractal algorithm to generate the sections. Like DCT, discrete wavelet transform mathematically transforms an image into frequency components. The process is performed on the entire image, which differs from the other methods (DCT), which work on smaller pieces of the desired data. The result is a hierarchical representation of an image, where each layer represents a frequency band [WR-13].

3.6.1 Video Compression Formats

The CCTV industry continues to move towards digital devices, such as Digital Video Recorders (DVRs) and IP devices, technicians need to be familiar with the subject of Compression – the methods such as MPEG, Wavelet™, and similar. The volumes of data produced by digitizing CCTV image streams would swamp the available storage and communications systems. To overcome this, the process of compression is applied to the

image stream, reducing the amount of information that needs to be transmitted and stored. In fact compression of the camera signal is not new - many people do not realize that all ‘analogue video’ has always been compressed. Similarly, there has long been a need for data compression in the computer industry. Specialist mathematicians have worked for many years on solving the basic problem of how to reduce the image size to produce the best compromise between image clarity, the data size of the image, and the amount of processing power it takes to run the compression method. The compression formats used in CCTV vary by manufacturer and by product. But the four most commonly used compression formats are given below.

3.6.1.1 MPEG

MPEG (named after the Moving Pictures Experts Group) is purpose designed for moving pictures, rather than being based on still image compression. This means that each frame is defined as the previous frame plus changes, rather than a full frame. The advantage of this is that compression is more efficient – the same quality can be displayed from less data. However, the method has problems when there is extensive motion between one frame and the next – there is a danger that the image get s ‘blocky’ and vague, losing some definition in the areas of the frame where the movement occurs. There is not one MPEG standard but several, changing over time, of which only the first two are relevant at present.. MPEG -1 was designed to output 15 frames per second video from limited bandwidth sources, such as CD-ROMs. MPEG-2, designed for high bandwidth applications such as High Definition TV (HDTV), delivers 30 frames per second video at full CCIR 601 resolution but requires special high speed hardware for compression and playback – PCs cannot handle this.

3.6.1.2 MPEG Types

MPEG is an asymmetrical system. It takes longer to compress the video than it does to decompress it in the DVD player, PC, set-top box or digital TV set. There are many types of MPEG systems such as :

- **MPEG-1** (Video CDs): Although MPEG-1 supports higher resolutions, it is typically coded at 352x240 x 30fps (NTSC) or 352x288 x 25fps (PAL/SECAM). Full 704x480 and 704x576 frames (BT.601) were scaled down for encoding and scaled up for playback. MPEG-1 uses the YCbCr color space with 4:2:0 sampling, but did not provide a standard way of handling interlaced video. Data rates were limited to 1.8 Mbps, but often exceeded.
- **MPEG-2** (DVD, Digital TV): MPEG-2 provides broadcast quality video with resolutions up to 1920x1080. It supports a variety of audio/video formats, including legacy TV, HDTV and five channel surround sound. MPEG-2 uses the YCbCr color space with 4:2:0, 4:2:2 and 4:4:4 sampling and supports interlaced video. Data rates are from 1.5 to 60 Mbps.
- **MPEG-4** (All Inclusive and Interactive): MPEG-4 is an extremely comprehensive system for multimedia representation and distribution. Based on a variation of Apple's QuickTime file format, MPEG-4 offers a variety of compression options, including low-bandwidth formats for transmitting to wireless devices as well as high bandwidth or studio processing. A major feature of MPEG-4 is its ability to identify and deal with separate audio and video objects in the frame, which allows separate elements to be compressed more efficiently and dealt with independently. User-controlled interactive sequences that include audio, video, text, 2D and 3D objects and animations are all part of the MPEG-4 framework.
- **MPEG-7** (Meta-Data): MPEG-7 is about describing multimedia objects and has nothing to do with compression. It provides a library of core description tools and an XML-based

Description Definition Language (DDL) for extending the library with additional multimedia objects. Color, texture, shape and motion are examples of characteristics defined by MPEG-7.

- **MPEG-21** (Digital Rights Infrastructure): MPEG-21 provides a comprehensive framework for storing, searching, accessing and protecting the copyrights of multimedia assets. It was designed to provide a standard for digital rights management as well as interoperability. MPEG-21 uses the "Digital Item" as a descriptor for all multimedia objects. Like MPEG-7, it does not deal with compression methods.

3.6.1.3 H261

H261 is a digitization and compression scheme for analogue video. It is widely used in video conferencing and is aimed at providing digitized video at a bit rate of 64Kbps-1Mbps, which is the bandwidth range of public data networks. Compression rates as high as 2500:1 are achieved, but of course at the cost of quality. The format is good for high frame rates, showing movement, but the resolution of those frames is not high. This is not good if, say, person identification images are required. But if the application is a non-security application such as video-conferencing, the quality is likely to be adequate. Uniquely among the compression formats discussed here, H261 encoded signals can also be decoded or decompressed by reversing the process from a valid reference or I-Frame.

3.6.1.4 Motion JPEG (M-JPEG):

Motion JPEG (JPEG stands for Joint Photographic Experts Group) is an adaptation of the popular JPEG image compression for still digital photos. JPEG is a lossless compression technique, losing very little data in the image. Motion JPEG creates a video stream from a succession of JPEG-compressed still photos. Because it is based on these high quality lossless stills, it delivers a much higher quality image than H.261. But at a cost – it requires a

considerably greater transmission bandwidth and storage capacity compared to its H261 counterparts. An advantage of Motion JPEG is that, because it is based on still images, it can produce any of its frames as a single image for identification purposes.

3.6.1.5 MPEG–2 AND MPEG-4

The ISO committee which developed the MPEG standard is currently at work specifying a successor standard known as MPEG-2. The video component is targeted for bit rates in the range of about 2 to 15 Mbps, which is sufficient for supporting HDTV. Additionally, MPEG-2 includes a number of new feature with the intent of providing compatibility with existing standards such as terrestrial video, MPEG, and H.261.

Compatible transmission is conceptually similar to today's TV broadcasts, which can be received by both color and black and white television sets. The MPEG-2 encoding is intended to allow a single transmission to be received by a range of digital televisions, from small portable units that might only support NTSC resolution to HDTV receivers. Scalable digital video is also critical to transmission over packet switching networks. As the load on the network increases, the transmitting node adjusts by decreasing the quality of the transmitted video.

The audio encoding for MPEG-2 is also being extended. MPEG-2 audio will encode up to five full bandwidth channels (left, right, canter, and two surrounded channels), an additional low-frequency enhancement channel, and up to seven commentary or multilingual channels. Several improvements on the MPEG audio format are planned for lower sample rates.

More recently, another digital video encoding standard effort known as MPEG-4 is underway. This new initiative is for very low bit rate coding of audiovisual programs, with particular application to mobile multimedia communications. Although MPEG-2 and MPEG both use a

DCT algorithm, it is anticipated that MPEG-4 will be based on a new algorithm, which through computationally more expensive, results in significantly higher compression.

JPEG2000 is similar to JPEG, but uses wavelet transform instead of discrete-cosine - transform (DCT) of JPEG. JPEG2000 offers a better image quality on higher compression levels. Another great advantage is a possibility to decompress lower resolution representation of the image. This feature is good for motion detection algorithms. However JPEG2000 compression needs way higher CPU performance, than JPEG [WR-6].

The JPEG standard was developed by the Joint Photographic Expert Group (part of ISO) for efficient storage of individual frames. Motion JPEG or M -JPEG is a series of separate JPEG images that form a video sequence. When 16 JPEG image frames or more are joined together per second, the result is an illusion of motion video. Video reproduction at 30 frames per second (FPS) for NTSC signals or 25 FPS for PAL signals is called full motion video or continuous-motion video. Although Motion JPEG is an unlicensed standard it is widely compatible with many applications that require low frame rates or technologies such as Video Analytics where frame by frame analysis is crucial [HH+2014], [NV+2013],[LJ+2012].

Advantages include,

- Ability to support multi-mega pixel resolution
- Ideal for courtroom single frame evidence
- Clearer images at lower frame rates than MPEG-4
- Frame by frame playback offers more frames to view
- Technology is simpler; this can reduce the cost of a camera or video codec
- At low bandwidth priority is given to Image Resolution [CD+2013]

Disadvantages include,

- No M-JPEG standard often means incompatibility issues
- High bit rate for scenes with little or no activity increases bandwidth and storage
- Video quality deteriorates at higher compression ratios
- Converting M-JPEG into another format reduces video quality
- Dated technology superseded by more bandwidth-efficient encoding techniques [WR-14], [MS+2012].

3.6.1.6 JPEG

JPEG undertook to develop a single standard applicable to the still-imaging needs of a wide range of applications in all the different industries that might use digital continuous-tone imaging. The scope of this is best seen by listing the objectives in detail:

1. To be at or near the state-of-the-art for degree of compression versus image quality.
2. To be parameterizable so that the user can select the desired compression versus quality tradeoff.
3. To be applicable to practically any kind of source image, without regard to dimensions, image content, aspect ratio, etc.,
4. To have computational requirements that are reasonable for both hardware and software implementations.
5. To support four different modes of operation:
   (a) Sequential encoding, where each image component is encoded in the same order that it was scanned;
   (b) Progressive encoding, where the image is encoded in multiple passes so that a coarse image is presented rapidly, followed by repeated images showing greater and greater detail;

(c) Lossless encoding, where the image is encoding guarantees exact reproduction of all the data in the source image;

(d) Hierarchical encoding, where the image is encoded at multiple resolutions.

3.7 Wavelet Compression

Like Motion-JPEG, Wavelet™ compression delivers high-quality moving images by starting with still images, applying a compression method to them, and putting them together to form moving pictures. It compresses images by removing all obvious redundancy and using only the areas that can be perceived by the human eye. Wavelet™ is up to four times more effective in reducing the volume of data than JPEG and M-JPEG. Wavelet™ is also seen as offering superior development potential to current MPEG compression, giving a greater amount of compression with equivalent quality. It transforms the whole image and not just blocks of the image, so as the compression rates increase, the image degrades gracefully, rather than into the 'blocky' artifacts seen with some other compression methods. Wavelet™ applications can have their preferred level of compression selected by the user – higher or lower.

3.8 Motion Video Compression Techniques

In the still-image compression techniques that we discussed above, we gave little consideration to the matter of compression or decompression speed. With still images, processing only needs to be fast enough that the user does not get bored waiting for things to happen. However, when one begins to think about motion video compression systems, the speed issue becomes overwhelming. Processing is a single image in one second or less is usually satisfactory for stills. However, motion video implies a high enough frame rate to produce subjectively smooth motion, which for most people is 15 frames per second or

higher. Full- motion video as used here refers to normal television frame rates – 25 frames per second for European systems, and 30 frames per second for North America and Japan. These numbers mean that our digital video system must deliver a new image every 30 – 40 milliseconds. If the system cannot do that, motion will be slow or jerky, and the system will quickly be judged unacceptable.

At the same time that we need more speed for motion compression, we also need to accomplish more compression. This comes about because of data rate considerations. Storage media have data rate limitations, so they cannot simply be speeded up to deliver data mote rapidly. For example, the CD-ROM's continuous data rate is fixed at 153,600 bytes per second–there is no way to get data out faster. If CD–ROM is being used for full-motion video at 30 frames per second, we will have to live with 5,120 bytes per frame. Therefore, we face absolute limits on the amount of data available for each frame of motion video (at least on the average); this will determine the degree of compression we must achieve.

For CD–ROM at 5,120 bytes of data per frame (40,960 bits per frame) and at a resolution of 256 × 240 pixels, the required compression works out to be 0.67 bits per pixel. Some still compression systems can work down to this level, but the pictures are not very good, and 256 × 240 already is a fairly low pixel count. Therefore, we should look at motion video to see if there are possibilities for compression techniques which can be used in addition to the techniques we discussed for stills.

Fortunately, motion video offers its own opportunities to achieve additional compression. There is the redundancy between adjacent frames–a motion video compression system can (or must) exploit that redundancy. Techniques for dealing with this are prediction and

interpolation or a special technique called motion compensation. We will discuss motion compensation shortly.

Another concept that comes into play with motion video system is the idea of symmetry between compression and decompression. A symmetric compression/decompression system will use the same hardware for both same speed. Such a system for motion video will require hardware that is too expensive for a single-user system, or else it will have to sacrifice picture quality in favour of lower-cost hardware. The reason is that a symmetric system must digitize and compress motion video in real time, which implies that the system must process data rates that can exceed 20 Mb per second.

However, this problem can be effectively bypassed by the use of an asymmetric system where the compression is performed on extensive hardware, but the decompression is done by low-cost hardware. This works in situations where the single-user system needs only to play back compressed video which has been prepared ahead of time-it will never have to do compression.

In fact, most interactive video applications do not require that the end-user system contains a compression capability–only decompression. Motion video for this class of application can be compressed (once) during the application design process, and the final user only plays back the compressed video. Therefore, the cost of the compression process is shared by all the users of the application. This concept can lead to the establishment of a centralized compression service which performs compression for many application developers, thus sharing the costs even further.

3.9 Motion Compensation

Consider the case of a motion video sequence where nothing is moving in the scene. Each

frame of the motion video should be exactly the same as the previous one. In a digital system, it is clear that all we need to do is the previous one. In a digital system, it is clear that all we need to do is transmit the first frame of this scene, store that and simply display the same frame until something moves. No additional information needs to be sent during the time the image is stationary. However, if now a dog walks across our scene, we have to do something to introduce this motion. We could simply take the image of the walking dog by itself, and send that along with the coordinates of where to place it on the stationary background scene-sending a new dog picture for each frames. To the extent that the dog is much smaller than the total scene, we are still not using much data to achieve a moving picture.

The example of the walking dog on a stationary background scene is an overly simplified case of motion video, but it already reveals two of the problems involved in motion compensation:

How can we tell if an image is stationary?

How do we extract the part of the image which moves?

We can try to answer these questions by some form of comparison of adjacent frames of the motion video sequence. We can assume that both the previous and the current frames are available to us during the compression process. If we do a pixel-by-pixel compare between the two frames, the compare should produce zero for any pixels which have not changed, and it will be nonzero for pixels which are somehow involved in motion. Then we could select only the pixels with nonzero compares and send them to the decompressing system. Of course, we would have to also send some information which tells the decompressing system where to put these pixels.

However, this very simple approach, which is a form of frame-to-frame DPCM, is really not too useful because of several problems. First, the pixel compare between frames will seldom

produce a zero, even for a completely stationary image, because of analog noise or quantizing noise in the system. This could be alleviated by introducing a threshold that would let us accept small comparison values as zero, but there is a more serious problem – images from video or film cameras are seldom stationary. Even if the scene itself contains no motion (which is unusual in natural scenes) the camera may be moving slightly, causing all pixel compares to fail. Even partial pixel movements will create changes large enough to upset the comparison technique.

Therefore, more sophisticated techniques are needed to do the motion detection for the purpose of motion compensation. This problem is usually addressed by dividing the image into blocks, just as we did with still images for transform coding. Each block is examined for motion, using approaches which consider all of the pixels in the block for motion detection of that block. If the block is found to contain no motion, a code is sent to the decompressor to leave that block the way it was in the previous frame. If the block does have motion, a transfer may be performed and the appropriate bits sent to the decompressor to reproduce that block with the inverse transform.

If enough computing power is available for the compression process, still more sophisticated approaches can be pursued. For example, blocks which contain motion can be further examined to see if they are simply a translation of a block from the previous frame. If so, only the coordinates of the translation (motion vectors) need to be sent to tell the decompressor how to create that block from the previous frame. A variation of this approach is used in the MPEG video compression standard. Even more elaborate techniques can be conceived to try to create the new frame using as much as possible of the information from the previous frame instead of having to send new information.

3.10 Assessment of Video Compression Algorithms

Video compression algorithms are divided in two groups: (i) Frame based compression (JPEG, Wavelet, JPEG 2000), and (ii) Stream based compression (MPEG-2, MPEG-4, H.264, MPEG-7). Usage of stream based compression algorithms enables greater savings on storage space and network bandwidth but as a trade-off these algorithms require higher computing performance [SG+2005].

Four of the most common type of compressions widely used in CCTV are: (i) Motion JPEG, (ii) MPEG-4, (iii) H.264, and (iv) JPEG2000. IP Video System Design Tool can help to see how choosing of compression methods can affect required network bandwidth and CCTV storage space, so one can clearly see the difference in terms of compression ratio. Motion JPEG is very popular compression format. MJPEG fits very well for video archives because of its frame based nature. MPEG4 can be 3 times more efficient in terms of compression ratio in comparison toMotion JPEG. But MPEG4 is a bad choice for systems with frame rate less than 5-6 frames per second. H.264 can be 50-100% more efficient in comparison toMPEG-4 and H.264 are ideal for CCTV systems with limited but stable bandwidth [HH+2014].