MULTIPLE DESCRIPTION VIDEO CODING USING MPEG – 21

By

Rhoda Oluwatoni Gbadeyan, B.Eng

A thesis submitted to the faculty of Graduate Studies and Research

in partial fulfillment of the requirements for the degree of

Master of Applied Science in Systems Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering

Department of Systems and Computer Engineering

Carleton University

Ottawa, Ontario, Canada

August, 2009

Your file Votre référence
ISBN: 978-0-494-60268-3
Our file Notre référence
ISBN: 978-0-494-60268-3

ABSTRACT

A multiple description coding technique is presented in this research work. The technique proposes to use adaptive techniques described in MPEG–21 to implement an innovative technique aimed at splitting encoded video sources outside the encoding process. The focus of this research is to develop an adaptive way to generate correlated, independently decodable multiple descriptions from a single bitstream.

To implement the proposed technique, a complete description of the encoded bitstream was created using a generic bitstream syntax description (gBSD) tool. A splitting technique was implemented based on the gBSD that copies the DC coefficients unchanged within each block, while the AC coefficients are split into the number of descriptions desired.

The proposed MPEG–21 based multiple description technique has the advantage of increased flexibility and adaptability to deal with existing network conditions. While it performs well at high redundancies, it also produces low quality, coarse description at low redundancies.

DEDICATION

I dedicate this research work to the God, the King of Kings and Lord of Lords who has made this possible. To Him be glory and honour, now and forevermore, Amen.

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Prof. Chris Joslin for his guidance, encouragement, valuable suggestions and let's not forget his constructive criticisms which have propelled me to the successful completion of my thesis.

I would also like to express my thanks to my parents, Prof. and Mrs. Jacob Gbadeyan for their support, guidance and love. I also express my appreciation to all my siblings: Grace, Ruth, Ayo and Tanitoluwa for their patience with me when I was too busy to call them and for their understanding and support throughout the course of my studies. You have spent time praying and encouraging me. You all have been my bulwark. I love you guys.

To my friends, your friendships have meant the world to me; your support has encouraged me, your understanding has uplifted me in more ways than I can express. It has been a blessing to know you all. To Tosin, you have been invaluable, God bless you.

To Bayo, your love and support, through the good and not so good times has filled my life with sunshine. I love you more than you can comprehend. I thank God for bringing you into my life. You are greatly appreciated.

I would also like to thank Sis. Tayo and Uncle Chima, for guiding me on the path to success and for making me feel welcome in Ottawa. Finally, for everyone who contributed in one way or the other towards the successful completion of this research work, you have been invaluable. I am so grateful. God bless you all.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLE

ABBREVIATIONS

BSD	Bitstream Syntax Description
BSDL	Bitstream Syntax Description Language
BSD to Bin	Bitstream Syntax Description to Binary
Bin to BSD	Binary to Bitstream Syntax Description
BS Schema	Bitstream Syntax Schema
CBP	Coded Block Pattern
CIF	Common Intermediate Format
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DIA	Digital Item Adaptation
DID	Digital Item Declaration
EOB	End of Block Marker
EoD	Event on Demand
FEC	Forward Error Correction
gBSD	generic Bitstream Syntax Description
gBSD to Bin	generic Bitstream Syntax Description to Binary
gBin to BSD	generic Binary to Bitstream Syntax Description
gBS Schema	generic Bitstream Syntax
GN	Group of Number
GOB	Group of Blocks
GOBSC	GOB Start Code

GOP	Group of Picture
HAME	Home Ambient Middleware Equipment
HTML	Hyper Text Markup Language
IDCT	Inverse DCT
IEC	International Electro-technical Commission
IID	independent identically distributed
ITU	International Telecommunication Union
ISDN	Integrated Services Digital Network
ISO	International Organization for Standardization
JPEG	Joint Pictures Expert Group
LOT	Lapped Orthogonal Transforms
LSB	Least Significant Bits
MAE	Mean Absolute Error
MB	Macroblock
MBA	Macroblock Address
MD	Multiple Description
MDC	Multiple Description Coding
MPEG	Motion Pictures Expert Group
MPT	Multiple Path Transport
MRA	Multi-Resolution Analysis
MSE	Mean Square Error
MSG	Main Slice Group
MVD	Motion Vector Data

PEI	Picture Extra Insertion
PLR	Packet Loss Rate
PMF	Probability Mass Function
P-MD	Predictive Multiple Description
PSC	Picture Start Code
PSNR	Peak Signal to Noise Ratio
PSPARE	Picture Spare
PTYPE	Picture Layer Type Information
QCIF	Quarter CIF
QMF	Quadrature Mirror Filter
QoS	Quality of Service
ROI	Region-Of-Interest
RRD	Redundancy Rate Distortion
RS	Reed-Solomon
RTT	Round Trip Time
SD	Single Description
SGML	Standard Generalized Markup Language
SIF	Standard Input Format
SSG	Side Slice Group
TAMD	Terminal Ambient Middleware Device
TCOEFF	Transform Coefficients
TCP	Transmission Control Protocol
TR	Temporal Reference

UDP	User Datagram Protocol
UED	Usage Environment Description
UMA	Universal Multimedia Access
VLC	Variable Length Coder
WLAN	Wireless Local Area Network
XML	Extensible Markup Language
XSLT	Extensible Stylesheet Language Transformation

CHAPTER 1

BACKGROUND ON VIDEO CODING, MULTIPLE DESCRIPTION CODING AND MPEG-21

1.1 INTRODUCTION

The digitization of image, audio and video signals has become a common occurrence. It is infeasible to transmit uncompressed video files because of their large bandwidth requirements; therefore the subject of video compression has garnered much interest. This has led to the success of low bandwidth transmission of video signals [1]. To ensure that video coding is of practical importance, the Motion Pictures Expert Group (MPEG) has standardized several video coding techniques. The main focus of the earliest MPEG standards was video encoding/decoding algorithms. The later standards have dealt more with the issues of video storage and retrieval, video transmission and bandwidth requirements. The later standards have been possible due in part to the success of the earlier established video compression MPEG standards.

The goal of compression systems is to produce an alternative representation of a raw video signal that uses a reduced bitrate to represent as efficiently as possible the video signal. While video compression has several important advantages such as a decrease in transmission bandwidth and storage requirements, a video sequence with a high compression ratio is also more susceptible to losses due to transmission over unreliable networks such as the wireless network [1, 2]. Other issues to be considered when choosing or designing a video compression algorithm includes the compression algorithms complexity, the choice between variable rate coding as opposed to fixed rate coding, as well as bitrate savings against a distortion tradeoff [1, 2]. These issues and more are often considered during the design of a video codec.

1.2 CONTENT GENERATION AND CONTENT DELIVERY

Two key areas of research have been explored with regards to digital signals – content generation and content delivery [3]. Content generation includes all the methods used for the representation of image, audio and video signals while content delivery deals with the efficient transmission of these signals. However, greater success has been achieved in the area of content generation over content delivery. The problem encountered with content delivery has been how to provide error free content delivery across an unreliable network such as the wireless network. Some solutions have been proposed to solve this problem. The two main accepted solutions are layered coding and multiple description coding (MDC) [3].

1.2.1 CONTENT GENERATION

In order to generate image sequences suitable for use in digitized video applications, the raw video has to be sampled. Sampling can either be applied to horizontal and vertical pixels in the raw video or it can be applied to a sequence of frames thereby digitizing the video sequence. Sampling raw video sequences generates digital video content suitable for use in video streaming over the internet.

Overtime, several video compression techniques have been implemented. The most commonly used include those proposed by MPEG. These include H.261, H.262/MPEG-2, H.263 and H.264 [1]. These compression techniques have garnered widespread use and success. Wavelet coding has also been introduced for use in video content generation. The International Telecommunication Union (ITU) Specialist Group has proposed several digital video formats for use in MPEG video compression standards [1]. These digitized video resolutions include: The Standard Input Format (SIF), the Common Intermediate Format (CIF) and the Quarter CIF (QCIF) which is a low bitrate version of CIF. In general, MPEG compression standards support one or more of these digital video formats e.g. H.261 supports both CIF and QCIF digital video formats [4].

Today, due to the exceptional success achieved in the field of video compression and video formatting there are as many multimedia coding technologies and users of these technologies as there are the hardware devices that use these technologies. The user in this context is the application, program, device or person etc that makes use of any multimedia content. There is currently an unprecedented demand for high quality multimedia content and change seems to be the only constant. The multimedia industry is scrambling to keep up with the user's need while users are also demanding to have customized multimedia content based on their preferences, the usage environment characteristics and device capabilities etc. They seek the ability to request a given multimedia content compressed using a compression standard of their choice which supports a particular digital format.

Users seek the ability to adapt multimedia content to suit each user's needs, thereby maximizing the user's experience. To deal with this changing role of users, and to ensure universal multimedia access (UMA), it is only feasible to adapt the multimedia content to fit the environment and not vice versa [5].

1.2.1.1 Multimedia Adaptation

Several solutions have been proposed to solve the problem of multimedia adaptation. One of such solutions is instantaneous transcoding. This involves an intermediary node located between the multimedia provider and the user performing instant encoding/decoding operations based on each user's preference [5]. Needless to say, this solution is rather cumbersome, cost intensive and impractical. Another solution proposed to solve this problem involves the multimedia service provider making available a single multimedia content in multiple video coding standards, at different qualities and sizes. At the point of delivery, the receiving device has the opportunity to make a choice of the desired standard, size and quality. This choice is often made based on the receiving device's capabilities. The bandwidth requirement of this solution makes its implementation a less likely possibility. Also, due to the large number of available coding standards currently in existence, it becomes next to impossible to implement this solution.

However, the MPEG has standardized a framework for the adaptation of motion pictures. This framework is specified in part 7 of the MPEG-21 standard [6]. This framework provides a technique that is beneficial to both the content provider and the user. It specifies a basis for product compatibility and interoperability while at the same time enabling content adaptation based on user preferences. This work of research proposes a framework for the efficient delivery of video content across a network while also providing for content adaptation.

1.2.2 CONTENT DELIVERY

Although a lot of advancement has been made in the area of transmission losses and network performances, there is still currently no guarantee that information transmitted across a network would be delivered error free. The focus of current research is to find a way to transmit information given existing network conditions in such a way that the decoded information will still be of a visually acceptable quality.

1.3 AN INTRODUCTION TO MPEG-21

Overtime a lot of work has gone into the research, development and representation of multimedia information. Due to these researches, sufficient information has been made available about the consumption and delivery of multimedia. The problem was that there was not a coherent structure that collated all these information and provided a unified front that was inclusive of information about each individual element while at the same time explaining how each element relates to other elements. To solve this problem, MPEG introduced a new standard; the ISO/IEC 21000 standard commonly known as MPEG-21 [7].

MPEG-21 provides the required framework that enables the use of multimedia information over various networks. This framework has an open design that provides a structure for the "creation, consumption, production, personalization, delivery or trade of multimedia content" [7]. MPEG-21 provides a generic architecture that can be employed in a unified way across the board by engineers in the field of video compression while not specifying the exact methods that must be used to implement any aspect of the framework. MPEG-21 strives to provide standardization within an open framework that enables interoperability.

MPEG-21 provides a framework for the personalization of video content. Part 7 of the MPEG-21 standard specifies the framework for MPEG-21 Digital Item Adaptation. The digital item adaptation (DIA) framework enables multimedia content to be adapted based on several constraints which include but are not limited to the following:

1. User characteristics.
2. Terminal capabilities.
3. Network characteristics.
4. Natural environment characteristics etc.

1.4 AN INTRODUCTION TO MULTIPLE DESCRIPTION CODING

In the 1970's, the idea of an MD coder was initially proposed out of a practical need to solve the problem of speech transmission over a telephone network. The idea of an MD coder then went from a proposed solution to solve a practical problem to a research and theoretical idea. This theory eventually gave rise to a solution to a practical engineering problem when an MD coder was proposed by Jayant for speech splitting [8]. Some early researchers who also studied MDC include Boyle and Miller [9, 10].

Finally, sometimes in the early 1980's [11], the odd/even sample splitting technique proposed by Jayant was implemented for speech coding [3]. The MD problem was officially formalized in 1979 at an information theory workshop by Witsenhausen et al. [3]. Overtime, this problem has evolved from usage strictly in speech coding to find applications in the design of video codecs.

1.5 PROBLEM DESCRIPTION

The MD problem as identified by Goyal [3] is a channel splitting problem in which two individual descriptions of the source are transmitted separately and the limitations on the quality of these descriptions are observed either by studying each description separately or by studying both descriptions jointly.

For the purpose of this research, the MDC problem is considered not so much as a channel splitting problem but more as a source splitting problem. Consider a source coding problem in which it is desired to transmit information over a network whose QoS cannot be guaranteed. Assume that the information transmitted across such a network is either received error free or lost. The desire is to solve the problem of network losses by generating two equally important descriptions of the source, such that each description is a low quality version of the source. These two descriptions can either be sent across a single channel or across two different channels. Provided that both descriptions are not simultaneously subject to losses, and provided at least one description is received at the decoder error free, a low quality version of the original source can be reconstructed. Peradventure both descriptions are received at the decoder, an increased quality version of the source can be reconstructed.

The goal of this research is to develop a multiple description technique that uses the video adaptation tools of MPEG-21 DIA to split a video source into two or more descriptions.

1.6 CONTRIBUTIONS

This research proposes a framework for the efficient delivery of video content across an unreliable network while providing for content adaptation. The proposed technique performs the video encoding process using the intra coding mode of the H.261 video coding standard. While the H.261 video coding standard was used, the proposed technique can also be implemented using other video coding standards such as H.263 or H.264. An efficient way for adapting the gBSD of the encoded bit stream is proposed using MPEG-21 DIA. The proposed technique uses the powerful tools of MPEG-21 for the purpose of MDC thereby establishing a means for enabling post encoding distribution of the encoded video sample.

Taking advantage of the fact that the more redundancy present in any information source, the higher its compression performance, a splitting technique is implemented. This splitting technique is different not only because it is implemented by transforming the gBSD of the encoded bitstream, but also because the splitting technique is not implemented within the traditional framework of the encoding process, but rather outside the encoding process, thereby creating a more flexible, adaptable way to split the encoded bitstream.

The instructions used by the parser to split the gBSD is provided using a suitable transformation language, thereby generating two or more new transformed gBSDs that conform to the splitting constraints. Using the DIA engine provided by the MPEG-21 framework, the adapted bitstreams can be developed from the transformed gBSDs and the encoded bitstream. Finally, the side descriptions can be reconstructed by decoding the adapted bitstreams. The proposed technique provides an approach to MDC that enables UMA without the need for further processing such as instantaneous transcoding.

1.7 THESIS ORGANIZATION

This thesis work is organized as follows: Chapter one provides an introduction of the subject matter. A description of the thesis problem is also provided in this Chapter. Literature review of existing research is presented in Chapter 2. This Chapter also provides a more in-depth view of the information theory aspects of the MDC problem. An overview of the existing MDC implementations is also presented in Chapter 2. In Chapter 3, a detailed description of the methodology used to implement the proposed MDC process is presented. An overview and analysis of the results obtained from the implementation of the proposed technique is given in Chapter 4 while the conclusion and future works are presented in Chapter 5.

CHAPTER 2

AN OVERVIEW OF MDC

2.0 INFORMATION THEORY AND MULTIPLE DESCRIPTION CODING

Existing MDC research often elects to examine the standard MDC design [11, 12]. This decision is often formed based on the fact that any proposed design that is effective with the standard MDC design can often be generalized to any number of descriptions with a high level of success [3]. For the purpose of this research work, the dual description MDC is also first implemented. Provided that both descriptions are not consecutively lost, a reconstructed signal of acceptable quality can always be obtained while if both descriptions are received a higher quality reconstruction can be obtained. This makes MDC a viable option for real time applications such as video conferencing [11]. This standard MDC design with two descriptions, two channels and three decoders is shown in Figure 2.1.

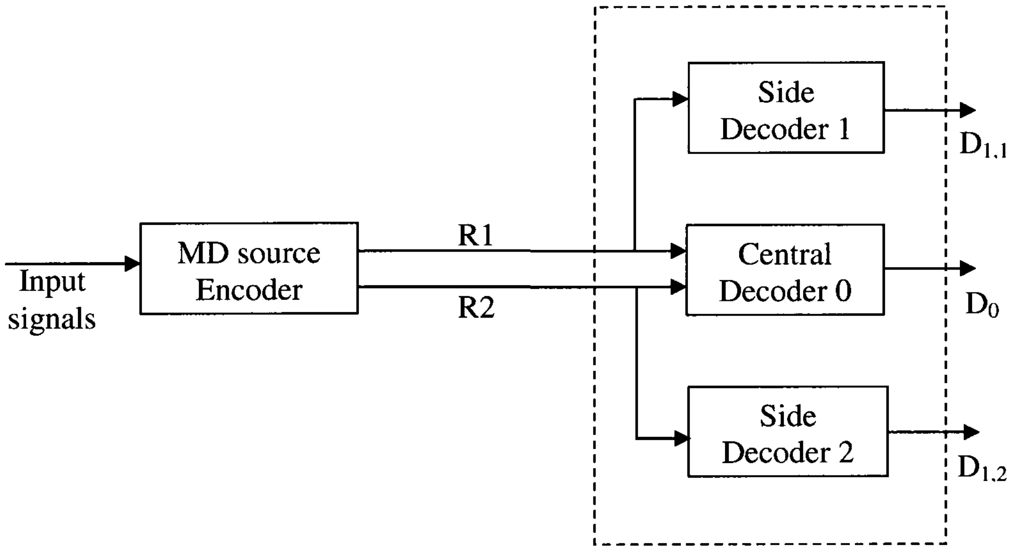

FIGURE 2.1 Standard 2 Channels, 3 Decoders Multiple Description Coder.

An ideal MD network is composed of two channels. Either of the two channels can fail with a probability of 0.5. Assuming the probability that any given channel fails is represented by P_{fi}, then

$$P_{fi} = 0.5 \qquad \text{where } i = 1, 2 \qquad \text{EQ. 2.1}$$

In MDC, a description is either received in its entirety error free or it is lost. Likewise, a channel either works throughout the transmission of a description or it fails [11, 12]. In the standard MDC described above, the following assumptions are made:

i. Both descriptions can be simultaneously received but cannot be simultaneously lost.

ii. Both channels cannot simultaneously fail.

The MD encoder is tasked with creating two descriptions. Both descriptions are sent individually over separate channels with bitrates (bits per source) R_i where i can be 1 or 2. The rate of the description is usually the length of that representation [3]. The total rate for both descriptions is given by

$$R_T = R_1 + R_2 \qquad \text{EQ. 2.2 [11]}$$

Given that the input to the encoder shown in Figure 2.1 is an n dimensional vector Y. The encoder is required to transmit the source vector Y across the two channels to the decoders. The middle decoder is the central decoder while the other two decoders are known as side decoders. The central decoder is capable of receiving information across both channels while the side decoders can only receive information over its respective channel. The encoder generates two codewords $f_1(Y)$ and $f_2(Y)$ which are transmitted across channels 1 and 2 respectively. The rate R for a (Y, n) code is given as

$$R = 1/n \log Y \qquad \text{EQ. 2.3 [13]}$$

This can be generalized for the case of 2 code words as

$$R_i = 1/n \log f_i(Y) \qquad i = 1, 2 \qquad \text{EQ. 2.4}$$

Where "i" denotes the channels of transmission. The encoder transmits the two code words at rates R_1 and R_2 respectively across the two channels to the decoders. Each decoder has a decoding function denoted by h_1, h_2 and h_0 where h_1 and h_2 are the decoding functions for the side decoders while h_0 is the decoding function for the central decoder. The decoding functions are responsible for generating a reconstruction of the received codewords. Based on the assumptions made previously, the central decoder and the side decoders generate reconstructions Y_0, Y_1 and Y_2 respectively as follows:

$$Y_0 = h_0(f_1(Y), f_2(Y)) \qquad \text{EQ. 2.5}$$

$$Y_1 = h_1(f_1(Y)) \qquad \text{EQ. 2.6}$$

$$Y_2 = h_2(f_2(Y)) \qquad \text{EQ. 2.7}$$

Although the encoder is not aware of any channel failures, the decoders have this information which is used to identify which decoder would be used for the reconstruction. As shown by the three equations above, three scenarios are possible at the decoder side. Scenario 1: Both descriptions are received and the central decoder generates the reconstruction Y_0. This gives rise to distortion D_0 at the central decoder. Scenario 2: Description $f_1(Y)$ is lost while $f_2(Y)$ is received. Therefore side decoder 2 is used to create a reconstruction Y_2. This gives rise to distortion $D_{2,1}$. Scenario 3: Description $f_2(Y)$ is lost while $f_1(Y)$ is received. Side decoder 1 is therefore used to create reconstruction Y_1. This gives rise to distortion $D_{1,2}$.

If both descriptions are balanced, then the following occurs

i) $D_{1,2} = D_{2,1}$ EQ. 2.8

ii) $R_1 = R_2$ EQ. 2.9

Since distortion relates to the quality of the reconstruction [3], generally, the central distortion is often much lower than the side distortions since the central decoder produces a higher quality reconstruction of the source compared with any of the side decoders.

2.1 STATE OF THE ART

The idea of splitting a source signal into two or more subsets was first researched in the 70's by Boyle and Miller while they were working at the Bell Laboratories [3, 9, 10]. This idea stemmed from the need to provide a more efficient way to transmit speech over a telephone network while taking into consideration the losses that can occur over the transmission line. Towards the end of the 70's and during the early 80's, the problem of MDC became well formulated and resulted in a number of publications both within the Bell Laboratories and at other international conferences [8, 9, 10, 14]. Early researchers in this field recognized MDC as a problem whose solution was to be discovered based on information theory principles.

In 1981, Jayant [8] proposed the use of sub-sampling and MDC principles for speech coding. Around that same time, El Gamal et al. [14] considered the MDC problem from a different perspective: Provided that a source signal can be split into two or more 'individually good' subsets, their research was targeted at discovering what rates can be achieved. This research led to the publication of a landmark paper which is still relevant and commonly referenced in the field of MDC.

Although the idea for MDC has been in existence since the 70's, it has not always been used for video coding. Prior to its introduction into video coding, layered coding was often used as the technique of choice for the transmission of video content across an unreliable network. As such layered coding has been studied for quite some time by a number of researchers [15, 16, 17]. Layered coding involves generating a base layer and one or more enhancement layers. The base layer is composed of the most important content of the source without which it may be impossible to reconstruct it while the enhancement layers contain information, that when added to the base layer, improves the quality of the reconstructed source. The base layer produces a coarse reconstruction of the source at the decoder. Without the base layer, the enhancement layer is not useful as it can produce no reconstruction of the source. The base layer is transmitted with a higher quality of service (QoS) requirement while the enhancement layer receives little or no service guarantees. For layered coding to be effective, the base layer has to be delivered virtually error free. This makes transmission over wireless networks even more unreliable. When received, each enhancement layer improves the quality of the reconstructed source [18]. Layered coding has been effective under conditions where in regards to rate distortion, packet scheduling has been optimized, but under circumstances where packet schedule optimization has not been achieved, the performance of layered coding has left more to be desired [18]. Regardless, due to the high QoS guarantee required by the base layer in layered coding, an alternative was needed.

The subject of MDC has been studied as an alternative to layered coding, especially for the transmission of a source over unreliable channels [19, 20, 21, 22]. In MDC however, the source is decomposed into multiple bitstreams or chunks of data which are often referred to as descriptions [2, 3] .Each description is of equal importance and is individually encoded and transmitted either in tandem with the other descriptions over the same channel or across separate channels over a network in multiple path transport (MPT). Each description received provides a low quality reconstruction of the source. The MDC decoder is then able to reconstruct a signal with an acceptable quality from each of the received descriptions [12]. The quality of the reconstructed signal in MDC improves with each successive description received. The goal of MDC is to allow reconstruction to proceed by combining any subset of the received descriptions even though there may have been losses of descriptions during transmission.

MDC is not always optimal compared to layered coding and vice versa, but it provides a suitable alternative depending on the required application especially since there is no need for retransmissions which significantly simplifies the required network design [11].

From the late 80's until date, research in this field has progressed. During this time, MDC has gained more relevance, not only in speech coding but also in video coding. MDC is currently used in video coding because it provides an efficient way to transmit video content across error prone and unreliable wired or wireless networks. While a general consensus exists about the relevance of MDC techniques in video coding, researchers have approached the study of MDC in video coding from two key directions. Some researchers have explored achievable rates and the relationship that exists between these achieved rates and the distortion introduced. This approach is representative of the El Gamal et al. approach to the MDC problem; however, it is geared towards video coding. On the other hand, other researchers have taken a more traditional approach by studying ways to efficiently encode and split a video source into individually good descriptions. As both of these approaches have led to important discoveries in this field of study and often tend to overlap, they are both presented in this Chapter. An examination of existing research, pertinent variations in techniques and their corresponding strong points are also presented.

2.2 LITERATURE REVIEW OF ACHIEVABLE RATES AND THE RATE DISTORTION THEORY FOR MDC

Rate distortion theory is the aspect of information theory that examines the relationship between the rate and the distortion levels that can be obtained for a given description. Rate is a measure of the length of a description while distortion is a measure of the quality of the reconstruction. In most practical applications, it is often impossible to achieve optimal rates while simultaneously achieving optimal distortion performance. There is therefore a tradeoff between both. Using this theory, a method for measuring the performance of a codec can be established assuming that a given reconstruction quality is associable with a given length of the description.

2.2.1 ACHIEVABLE RATES AND THE CORRESPONDING RATE- DISTORTION REGION FOR MULTIPLE DESCRIPTIONS

In 1982, El Gamal et al. [14] presented their proposal on the achievable rates for a multiple description system. Today, most multiple description systems are designed based on their proposed design. Assuming that we have a sequence X_i , i = 1,2 that is composed of independent identically distributed (I.I.D) discrete random variables which are drawn according to p(x); where p(x) is a known probability mass function (PMF), the question proposed by El Gamal et al. is how to determine what rates R_1 and R_2 should be used to send two descriptions of X such that a receiver 1, with access to only description 1 can recover $\hat{X}_i$ with distortion D_1; likewise, a receiver 2 with only description 2 can recover $\hat{X}_i$ with distortion D_2 and finally, a receiver with both descriptions can recover $\hat{X}_i$ with distortion D_0. Put simply, what set of (R_1, R_2) is necessary and sufficient to achieve the fixed distortions D_0, D_1 and D_2?

This problem came about as a result of the need to send a description of a given stochastic process to a destination over a given communication network across which losses may occur. To deal with these losses, the description is split, such that each description must be individually good. For each description to be individually good, they necessarily must be dependent since when both descriptions are independent, they generally cannot be individually good [14].

Goyal has a different approach to the rate-distortion problem. Goyal [3] defines a rate distortion pair (R, D). This rate distortion pair is said to be achievable provided that there exists a source with the distortion D and the rate R, given that the length of the source code is some positive integer n. The set of all achievable rate-distortion pairs form the rate-distortion region [3]. Based on this rate-distortion region, Goyal defined two functions: The rate-distortion function R(D) and the distortion-rate function D(R). R(D) is defined as the minimum rate achievable such that the rate distortion pair is located within the rate-distortion region. Likewise D(R) is defined as the minimum distortion achievable such that the rate-distortion pair is in the rate-distortion region.

Goyal concluded that although the rate distortion problem has been solved for a limited number of sources and distortion measure, it is generally very difficult to obtain an optimal solution. This is due to the fact that it is next to impossible to determine the boundary of the defined rate-distortion region based solely on a given set of definitions. It is also difficult to design a variable length n source code that effectively minimizes the rate for a given distortion or on the flip side minimizes the distortion for a given rate [3].

Wang et al. [11] proposed another method for the measuring of the performance of an MD coder based on the rate distortion theory. Given that the distortion D_0 is minimized for a fixed rate R in a single description coder (SD coder), the performance of the SD coder can be measured by its rate-distortion function R (D_0). While the task of measuring the performance of an SD coder becomes relatively easy, translating this performance measure to a standard MD coder poses some difficulty since there is a requirement to minimize the distortion for both the central decoder and the side decoders simultaneously. To solve this problem, the method proposed [11] uses the redundancy rate distortion curve (RRD) which was initially proposed by Wang et al. [23] to measure the efficiency of an MD coder.

To implement this method, first the best quality achievable by an SD coder when a predetermined rate R* is used is defined as the central distortion D_0 for the SD coder. The MD coder then attempts to match the central distortion D_0 achieved by the SD coder. The rate at which the MD coder achieves D_0 is defined as R. The redundancy required by the MD coder to achieve D_0 compared to the SD coder intuitively gives the redundancy measure ρ which is defined as

$$\rho = R - R^* \qquad \text{EQ.2.10}$$

From the redundancy measure, the RRD function was intuitively defined as $\rho\ (D_1;\ D_0)$ which is the additional bit rate required in an MD coder in order to obtain a minimized side distortion D_1 of that MD coder at central distortion D_0.

Wang et al. [11] then formulated an MD encoder optimization problem based on an ideal MD network. An assumption was made that the probability of failure of the channels in the ideal network was known. Subject to a rate constraint, their approach was to minimize the average distortion on the basis of the known channel probabilities. This gives

$$\min_{Z,M}((1-P)^2 D_0 + 2p(1-p)D_1 + \lambda R) \qquad \text{EQ.2.11 [11]}$$

Given the rate constraint that $R \leq R$ max where

Z is the performance parameter of the SD coder

M is the performance parameter of the MD coder based on Z and

R is the total rate

$$R = R^*(Z) + \rho(M,Z) \qquad \text{EQ.2.12 [11]}$$

Substituting EQ.2.12 into EQ.2.11 gives

$$\min_{Z} \{(1-p)^2 \; D_0(Z) + \lambda R^*(Z) \; + \min_{M} \{ 2p(1-p)D_1(Z,M) + \lambda\rho(M,Z)\} \quad \text{EQ.2.13 [11]}$$

Based on the equations above, the rate distortion problem was hence intuitively described as the problem of allocating a total rate R max between R* and ρ such that the average distortion is minimized [11].

2.2.2 GENERAL BASIC DEFINITIONS OF RATE DISTORTION THEORY

Given that we have a sequence X_i, i = 1, 2 that is composed of i.i.d discrete random variables which are drawn according to P (x); where P (x) is a pmf.

Given a reconstruction space $\bar{\bar{X}}$ that is associated with the distribution measure d: $\bar{X}$ x $\bar{\bar{X}}$ => R, the distortion measure on sequences can be defined by obtaining the average per symbol distortion.

Therefore, given n – sequences in $\bar{X}^n$ x $\bar{\bar{X}}^n$, the distortion measure is defined as

$$d(x, \hat{x}) = \frac{1}{n}\sum_{i=1}^{n} d(x_i, \hat{x}_i) \qquad \text{EQ.2.14}$$

The definition can then be extended to the case of descriptions and their reconstructions. If a description $x \in \bar{X}^n$ maps i: $\bar{X}^n$ => {1, 2 . . . 2^{nR}} where R is the rate of that description in bits per symbol of x, then the reconstruction of the description X is a map

$$\hat{x} : \{1, 2 \ldots 2^{nR}\} \;\Rightarrow\; \bar{\bar{X}}^n \qquad \text{EQ.2.15}$$

This reconstruction leads to a distortion $D^{(n)}$ which is the expected average per symbol distribution

$$D^{(n)} = Ed\,(X, \hat{x}\,(i\,(x))) \qquad \text{EQ.2.16}$$

$$= E\,(\tfrac{1}{n})\sum_{k=1}^{n} d\,(X_k, \hat{x}_k\,(i(X))). \qquad \text{EQ.2.17}$$

"This rate distortion pair is said to be achievable for $\{X_i\}_{i=1}$ if for $n = 1, 2 \ldots.,$ there is a sequence of rate R descriptions and reconstructions i: $\bar{X}^n \Rightarrow \{1, 2 \ldots., 2^{nR}\}$ and $\bar{\bar{X}} : \{1, 2 \ldots 2^{nR}\} \Rightarrow \bar{X}^n$, respectively such that $D^{(n)} \leq D$, for all n sufficiently large".

2.3 LITERATURE REVIEW OF TECHNIQUES FOR MULTIPLE DESCRIPTION CODING AND THE ARCHITECTURE FOR VIDEO ADAPTATION

Overtime, a lot of research has been done on the subject of MDC. These studies have led to the development of various techniques for implementing MDC. Some of these techniques are briefly discussed in this section and will be discussed further in subsequent sections. Goyal [3] gives a very detailed overview of the history and background of MDC as well as state of the art information on the current technologies and techniques used for the implementation of MDC. MDC is generally based on the premise of sub-sampling a source signal, thereby decomposing it into any number of subsets. This decomposition can either be done in the spatial, temporal or frequency domain [24, 25, 26, 27, 28.].

Wang et al. [11] proposed a technique for the implementation of MDC in video coders. This technique is the use of predictive coding in MD coders. For the standard two descriptions, three decoders MD coder, the state of the decoders can be any one of three possible states. In predictive multiple description coding (P-MD Coding), the video encoder tracks the state it expects to be present at the decoder and bases its predictor on that state. Currently, predictive coding is commonly used in video coders. Another technique is MDC using unequal forward error correction (FEC) [11, 29, and 30]. This method is suitable for use with scalable bitstreams. In this technique, different bitstreams have different levels of FEC applied.

Although both of these techniques provide a viable implementation of MDC in video coders, the drawback of using P-MD coders is the issue of mismatch control. Mismatch is a condition that occurs when the encoder uses a signal for prediction that is not present at the decoder [11]. This problem of mismatch can be avoided by using the appropriate class of predictors. Although MDC is generally based on the inclusion of a certain level of redundancy information in each description, MDC using unequal FEC requires a higher level of redundancy than most MDC techniques. An overview of both of these techniques is provided in Section 2.2.1.

There have been several MDC techniques that have been based on transform coding [2, 27, 28]. These techniques use wavelets, DCTs or LOT for MDC. Other techniques go a step further by incorporating motion compensated prediction [31]. Although 2D transform coding along with motion compensated prediction is a commonly used technique for video and MD coding, it is possible to achieve a 3D transformation of a video sample. This is done by replacing motion compensated prediction with a transform along both the spatial and the temporal axis; a temporal axis transform is required to handle the temporal correlation present between the frames in the video sample. A comparison of both of these techniques in terms of compression efficiency indicates that the motion compensated video compression technique out performs the 3D transform method. In terms of computational speed however, the 3D transform method out performs the motion compensated prediction method.

Tillo et al. have proposed a zero-padding DCT based MDC technique [32] that also falls in the transform coding group of techniques. This technique is suitable for both image and video coding. The basis of this technique is to oversample the original source. The source is discrete cosine transformed prior to being oversampled. Next, IDCT is performed on the zero-padded source to produce a transformed source with a new size characteristic. This new source is then sub-sampled thereby splitting it into two descriptions which can now be encoded. Provided both descriptions are received by the decoder, they are interlaced and the full source information is recovered, while if either description is lost, the missing information can be obtained through linear prediction.

Su et al. and Verdicchio et al. have also proposed innovative MDC techniques worth mentioning. Su et al. [33] proposed a novel multiple description technique based on the video coding tools provided by H.264/AVC. Their proposed technique uses a Main Slice Group (MSG) and a Side Slice Group (SSG) to split the video sequence into two descriptions. The MSG provides the most essential portion of the information contained in the source. This information is encoded using the standard H.264 technique, while the SSG provides the redundancy information. The portion of the source that comprises the redundancy information is coarsely encoded with larger quantization steps and fewer bits compared to the MSG. This splitting technique proposed by Su et al. is similar to that proposed by Norkin et al. [2] for use in 3D-DCT based MDC. This technique uses a coarsely quantized shaper along with a finely quantized residual sequence which provides the enhancement information.

Verdicchio et al. [34] have proposed a MDC scheme suitable for Scalable erasure resilient video coding. This technique is called Embedded MDC. It is so called because it uses embedded multiple description scalar quantization to provide video scalability and erasure resilience. Provided that transmission is done across reliable channels, improved visual quality is ensured as opposed to increased erasure resilience which is not essential under the current channel conditions. However, when transmission of the source information is done across unreliable channels, visual quality is traded for increased erasure resilience.

Chen et al. [35] have proposed a technique that combats the packet losses that can occur while roaming in a wireless local area network (WLAN) with MDC techniques, thereby ensuring that a constant video quality can be achieved. These losses often occur due to fading and handoff, hence multipath transport is used to increase network reliability. Therefore, combining the error resilience properties of MDC with the capabilities of multipath transport, a greater service guarantee can be achieved regardless of the inherent losses. In the proposed technique, network capabilities and conditions are obtained by utilizing active probing along with a channel status detection mechanism. A transcoder stores a list of possible paths based on the channel quality of each path. By making use of this information, the transcoder makes a decision about whether to send data without MDC over a single channel with excellent quality or to send data after applying MDC over two channels with good quality, thereby increasing the probability that an acceptable quality reconstruction will be achieved at the decoder.

In addition to the current advances made in the field of multiple description coding, advances have also been made in the area of video adaptation. Video sources can be adapted in several ways including but not limited to formatting adaptation or conversion, bandwidth reduction [36, 37], video transcoding [38] and quality adaptation

[39] etc. These adaptation techniques often make use of the powerful adaptation tools of MPEG-7 and MPEG-21. These tools were developed to help with the adaptation and delivery of digital video.

Devillers et al. [40] have proposed using the BSD tool of MPEG-21 for the adaptation of streaming video. The proposed adaptation technique uses both BS Schema and the BSDtoBin processor for adaptation. Iqbal et al. [5] have also proposed a similar adaptation implementation for use with H.264. However, this implementation replaces the BSD tool with the gBSD tool, thereby enabling a more abstract and general approach towards the transformation of a bitstream.

Negru et al. [41] have proposed a technique based on MPEG-21 that provides access to multimedia services irrespective of the geographical location of the user. This technique uses an ambient middleware along with the adaptation tools of MPEG-21 to adapt the services requested by the user in a way that maximizes the user experience. Two middleware were used in this implementation: the home ambient middleware equipment (HAME) on the home service platform and the terminal ambient middleware device (TAMD) at the mobile side. Based on the user specific metadata description provided by MPEG-21 i.e. user preferences, terminal capabilities and network characteristics, HAME is able to adapt the requested services prior to transmitting it to the mobile user. TAMD can also request those same multimedia services based on the mobile user's home service platform, thereby ensuring that the mobile user has access to the same multimedia services.

The authors in [42] have also proposed the use of the MPEG-21 DIA framework for event-on-demand (EoD) video adaptation. The proposed technique also uses the multimedia description schemes provide by MPEG-7. This technique was implemented in three basic steps: first, selected or identified event information was obtained from the bitstream along with an MPEG-7 annotation XML file, thereby generating a generic bitstream syntax description. Next, in order to make informed adaptation decisions, MPEG-21 metadata providing user preference and network characteristics information etc. was used. Finally, the adaptation was achieved using an adaptation engine that automatically parses the gBSD and the adaptation decisions formulated in stage two. This proposed technique obtains user preference information by allowing the user to choose their events of choice. The user can choose several events and assign varying levels of importance to such events. Based on these choices, the adaptation technique allocates more resources to the selected events in a priority based way. By using MPEG-21 DIA as well as the gBSD tool, this implementation ensures a more universal approach while minimizing computational cost.

Huang proposes the use of MDC for the adaptive delivery of media content across wireless networks [43]. This proposed technique is different from other techniques because it does not use an MPEG-21 based adaptation approach. The proposed technique uses MDC to maximize the quality and satisfaction of the media user. In general, there are several adaptation techniques that can be used to implement the MDC of a video sample. These techniques include format conversion, rate adaptation and MDC based on down sampling to achieve a lower resolution. Rate adaptation based MDC techniques can further be subdivided into adaptation based on frame or DCT coefficient dropping, scalable coding and object encoding where based on the region of interest (ROI), some objects can be disregarded based on network conditions and re-quantization [44]. Huang

[43] proposes a technique whereby based on current network conditions, MDC is implemented not only based on the adaptation techniques given above but also based on video to image adaptations, image to text adaptations and text to audio adaptations. The decision about the modality or level of adaptation used should be made based on user preferences, resource constraints or network conditions [43].

There has been sufficient research done on MDC by various researchers who have introduced several techniques for its implementation. Likewise, there have also been in-depth studies into the architecture of video adaptation as well as the techniques used to perform video adaptation. However, there has been little or no research that has investigated the use of MPEG-21 adaptation techniques to implement MDC.

2.3.1 THE ARCHITECTURE OF COMMON MULTIPLE DESCRIPTION CODING TECHNIQUES

The architecture of some of the commonly implemented MDC techniques which have been introduced earlier in this Chapter is described below. Their advantages, pros and cons are also presented.

2.3.1.1 Predictive Multiple Description Coders (P-MD Coders)

During the designing phase of a predictive coder, it is always important that predictions formed at the encoder matches those formed at the decoder. This is done to prevent mismatch. While mismatch control is an important design consideration for these types of MD coders, redundancy allocation is also important. There is a trade-off between mismatch control and redundancy allocation since mismatch control may lead to increased redundancy requirements. It is therefore necessary that depending on the actual goal of the design, mismatch control may be foregone in favor of optimal redundancy allocation and vice versa.

MD-Predictive coders are well discussed by Wang et al. [11]. Predictive coders base their predictor upon the state that the encoder expects to be present at the decoder. Provided that no information is lost, the encoder state and the decoder state are identical [11]. In the case of a generic predictive MD-Coder with 2 descriptions, 2 side decoders and a single central decoder, there are 3 states on which an encoder can base its predictor upon. However, there is no way for the encoder to determine precisely which of the 3 available states are present at the decoder. In a situation where the encoder then uses a predictor based on a state not available at the decoder, this leads to mismatch and invariably, error propagation since mismatch error in one frame will be propagated onto the remaining frames.

The generic P-MD Coder has to produce two useful descriptions irrespective of mismatch and state problems, therefore various techniques have been developed to deal with the problems associated with the design of predictive MD-Coders. The predictors that may be encountered in a P-MD Coder can be categorized into three different classes on the basis of the existing tradeoff between the side distortion and the overall redundancy as follows [11]:

1. Class A Predictors: This class of predictors has no mismatch i.e. the encoder has created a prediction error signal using identical predictors as the decoder. This is done irrespective of the current decoder state. This can be done either by using a two state encoder or by using only one predictor. A two state encoder [45] is one

that uses two separate predictors which make predictions based on information obtained from both descriptions.

2. Class B Predictors: The predictor used by this class of predictors is the same predictor used by any single description predictive encoder. Using this class of predictors introduces no additional redundancy, however, because the decoder requires both descriptions to form this prediction, Class B predictors introduce mismatch while minimizing the prediction error.

3. Class C Predictors: This class of predictors enables a trade-off to be made between prediction efficiency and mismatch control. Depending on the desired outcome, appropriate levels of prediction efficiency and mismatch can be chosen. The level of prediction efficiency is measured against the performance of a single description predictor.

In general, various predictive coders can be formed based on different classes of predictors. In predictive MDC, the mismatch introduced can be controlled by propagating intra-coded (I-Frames) frames periodically to remove mismatch that may have accumulated over time.

2.3.1.2 Multiple Description Coding Using Unequal Forward Error Correction (MD-FEC)

MD-FEC was first proposed by Albanese et al. [29] and by Davis et al. [30]. This method is based on scalable bitstreams. Unlike previous MDC techniques where the computational complexity of the source coder is increased by having the encoder split the source into multiple descriptions thereby yielding the M descriptions directly, in MD-

FEC, unequal cross-packet FEC is applied to various parts of the compressed bitstream thereby yielding various descriptions.

First, the scalable bitstream is divided into 'A' layers; each of these 'A' layers is further subdivided into 'B' groups each having an identical length. To yield the various descriptions, a Reed-Solomon (RS) code has to be applied across the equal-length 'B' groups, thereby yielding 'A' groups [11]. Each description is therefore composed of a combination of the bits from one particular group picked across all the layers. For example, if there are ten layers in the scalable source code, and each layer is divided into 'I' groups; then description 'C' would be composed of the bits from group 'C' across all ten layers. To recover the original bitstream from the descriptions, Reed-Solomon decoding would have to be applied.

Applying MD-FEC to scalable video coders involves each Group of Pictures (GOP) being partitioned into the required number of layers and then applying unequal FEC to the different layers. The optimal number of layers required can be decided by reducing the expected distortion based on a defined total rate constraint such as channel coding [11]. For temporal prediction, it is necessary to include a number of layers as reference layers. Class A predictors require that only one layer be used for prediction. When all available layers are used on the other hand for prediction, then a class B predictor without mismatch has been used. This method of grouping on the basis of the number of layers used for prediction can be made for all predictor classes.

2.3.1.3 Multiple Description Coding of Video Based on 3D Block Transforms

There is often a need to perform MDC using a low complexity video coder, especially for portable devices with low energy and computational capabilities. In cases such as this, the traditional H26x codecs may not be optimal due to the common computational complexities associated with them, therefore an alternative solution is needed. One of such possible alternatives is the use of 3D block transforms, which tend to have less complex computational requirements. Different 3D block transforms have been proposed for use in multiple description video coding including wavelets, DCT and LOT [46, 47, 48, 49].

Norkin et al. [2] have proposed a technique that uses 3D-DCT for MDC. 1D-DCT has very fast and efficient implementations. Research has shown that the 3D-DCT also has very efficient implementations [50, 51, 52]. Although in terms of compression efficiency, the 3D-DCT does not perform as well as the H.263 codec, in terms of computational speed, however, it performs at least three to four times better than the H.263 codec [53].

This proposed technique is a two staged MD Coder that uses 3D-DCT to divide a source into two independent descriptions. This coder tries to balance the computational load between the encoder and the decoder. A 16x16x16 3D-DCT that gives a coarse approximation of the source is used at the first stage to provide redundancy information that is included in both descriptions. This coarse approximation is called the Shaper [2]. The second stage uses a finer 8x8x8 3D-DCT to produce enhancement information at a higher bitrate from the residual sequence that is split between the two descriptions. The quantization steps used for the shaper are coarse while the quantization steps used for the residual sequence are finer [2]. The shaper combined with one half of the residual sequence forms one description while the shaper combined with the second half of the residual sequence forms the second description. Since the shaper provides redundancy information, peradventure one of the descriptions gets lost during transmission; the source can be reconstructed provided any one of the two descriptions is received.

CHAPTER 3

METHODOLOGY OF A MULTIPLE DESCRIPTION VIDEO CODING TECHNIQUE BASED ON MPEG-21

3.0 INTRODUCTION

All implementations of MD video coding have one element in common – A video codec. This is required to carry out the encoding/decoding compression process to ensure that the video is in a format suitable for transmission. While all MD coding implementations have to compress the raw video signal, they are not limited to any one particular video compression standard. Therefore, most MD techniques differ based on the compression standard used for video compression and the technique used to split the compressed source into the required number of descriptions. The goal of this work of research is to implement multiple description coding using the strong adaptation tools provided by MPEG-21 DIA. This proposed technique makes use of four important components: The H.261 video coding standard, the eXtensible Markup Language (XML), the eXtensible Stylesheet Language Transformation (XSLT) and the MPEG-21 Digital Item Adaptation (DIA) engine. These components are introduced next.

3.1 THE EXTENSIBLE MARKUP LANGUAGE (XML)

The Extensible Markup Language (XML) is a markup language that is used to describe data structure. XML is often used for web based applications; however, it is also suitable for a variety of non-web based applications such as the transfer of sensitive data between organizations or businesses. Since its inception, XML has become an important technology commonly used for description and/or structuring of data [54]. XML and the Hyper Text Markup Language (HTML) were created based on the Standard Generalized Markup Language (SGML). The problem with SGML is that although it is a very powerful language, it is too complex. HTML has enjoyed greater success than its parent language, but it has also fallen short in that it is only suitable for specifying how a given document should be displayed in a browser. It has no information about the content of the document nor can it provide any information about the document's content to any third party. XML is preferred to the other markup languages since it is less complex than SGML and not as limited in its scope as HTML. XML has gained widespread usage because it can provide information about its content to any third party program such as XSLT. Since XML is a subset of SGML, it is fully compatible with SGML [54]. XML by itself does not do anything; rather it provides syntax for language creation [55]. XML is a self describing language i.e. its syntax can easily be understood by a layman. The primary goal of XML is to serve as a means to enable software to communicate with each other. In general, XML satisfies two major usage criteria as follows [55]:

1. It is capable of distinguishing between the data and the instruction for displaying that data on a given device.

2. It is also able to transmit information between organizations without the need for these organizations to invest in any specific enabling software.

3.2 THE EXTENSIBLE STYLESHEET LANGUAGE: TRANSFORMATIONS

This language was established to transform the structure of XML documents. The XSLT version 1.0 Recommendation was released on 16th November 1999 [55]. The XSLT process of transformation is a two staged process: the first step involves transforming the structure of the input XML document into the desired output structure while the second step is to produce the final output document type. XSLT is not a traditional programming language, rather it is a declarative, not a procedural language. In procedural programming, a step by step instruction about what needs to be accomplished is given to the computer whereas in declarative programming, the computer is told what the programmer's end goal is and not how to achieve that goal [54]. An XSLT program is called a stylesheet. The input into the stylesheet is usually an XML document. The output document can be XML, HTML or text as required [55].

An integral part of programming using XSLT is XPath. XPath is an expression language used in XSLT [56]. It is basically used as a pointer to select or exclude specific portions of an XML document during processing. In an XPath model, key parts of the XML document needs to be represented as nodes. XPath expressions can either be absolute or relative. In absolute XPath expressions, the start point is always the root node while in relative XPath expressions; the start point could vary depending on what portion of the XML document is desired. In general, the starting point of an XPath is known as the Context [54].

XSLT together with XPath form a very powerful tool used in XML processing. This is especially due to the fact that the structure of an XML document typically needs to be changed prior to its usage [54]. XSLT can be used to convert XML to its final presentation format.

3.3 INTRODUCTION TO MPEG–21 DIGITAL ITEM ADAPTATION (DIA)

The MPEG-21 standard came into existence based on a universal consensus within the MPEG community, that despite the existence of MPEG-1, MPEG-2, MPEG-4 and MPEG-7, there was limited interoperability between these standards and the issue of multimedia distribution and consumption remained largely unsolved. The goal was to identify how the existing standards fit together and also to identify areas within existing standards that left more to be desired before full interoperability could be achieved.

MPEG-21 was in no way designed to replace or invalidate other existing MPEG standards; rather it was developed to enable the seamless exchange of any type of multimedia content between two or more users. The concept of MPEG-21 was designed around two actors: the user and the digital item. The technical report [6] for the MPEG standard defines a user as follows: "A user is any entity that interacts in the MPEG-21 environment or makes use of a digital item". The report also defines a digital item as "a structural digital object with standard representation, identification and metadata within the MPEG-21 framework. This entity is also the fundamental unit of distribution and transaction within this framework".

The desire of the standard is to enable a user to have seamless, universal and if need be, selective access to multimedia content on whatever device the user chooses. The MPEG-21 standard is made up of 7 key elements which include:

1. Digital item declaration (DID).
2. Digital item identification and description.
3. Content handling and usage.
4. Intellectual property management and protection.
5. Terminals and networks.
6. Content representation.
7. Event reporting.

3.3.1 DIGITAL ITEM ADAPTATION (DIA)

DIA developed out of the increasing need to provide UMA while at the same time providing interoperable adaptation of the multimedia content by the user. DIA is an important aspect of the MPEG-21 framework. Within this framework, multimedia content adaptation is based on a number of DIA tools. The DIA concept is best represented by the Figure below. The Digital item is representative of the bitstream of a multimedia content along with all relevant descriptions of that bitstream [5]. As shown in the Figure, the digital item is processed by the DIA engine as well as DIA tools, thereby generating as output, an adapted digital item. The DIA tools are introduced in the next section.

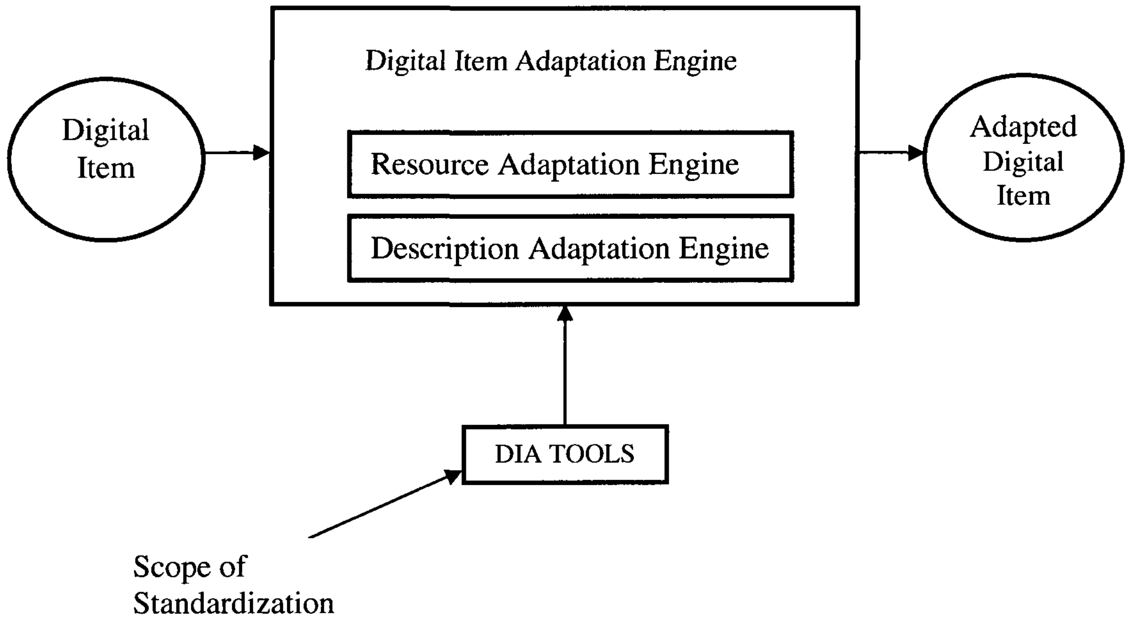

FIGURE 3.1: The approach of DIA Standardization [57].

3.3.2 DIA TOOLS

To support interoperability, MPEG specified a set of tools within the MPEG-21 DIA framework that supports multimedia content adaptation. These tools are designed to provide device independent and coding – format independent adaptation when needed [57]. DIA tools include the following:

1. Usage environment description tools.
2. DIA configuration tools.
3. Session Mobility.
4. Metadata adaptability.
5. Universal constraints description tools.
6. Terminal and network QoS.

7. Bitstream syntax description (BSD), bitstream syntax description language (BSDL) and generic bitstream syntax schema (gBS Schema).
8. Bitstream syntax description link.

These tools are displayed in Figure 3.2. For the purpose of this research work, the DIA tool of key importance is the bitstream syntax description tool set. This tool set is explained in more detail in subsequent sections.

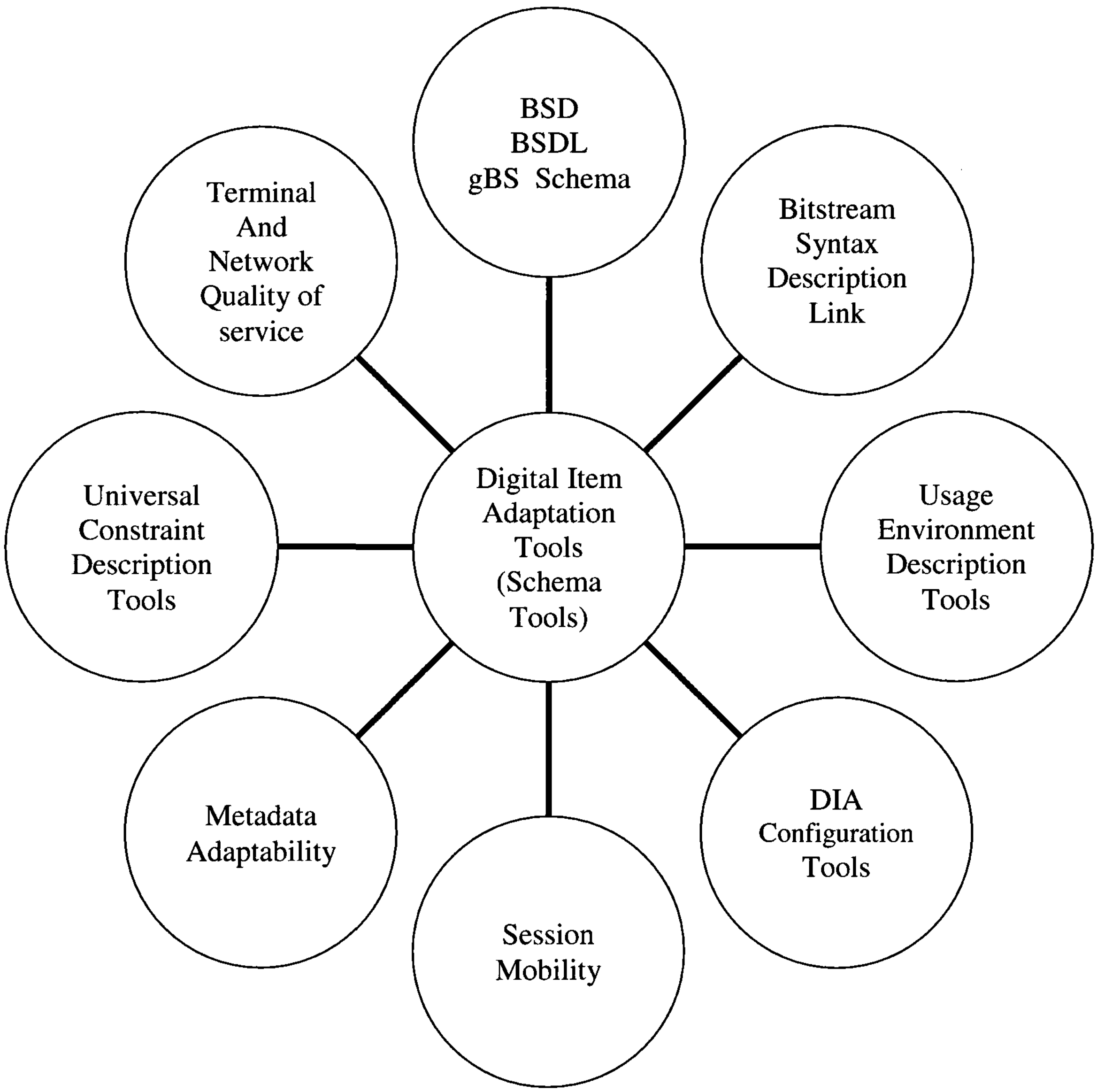

FIGURE 3.2 The DIA Tools.

3.3.2.1 BITSTREAM SYNTAX DESCRIPTION (BSD) TOOL SET

Adaptation of digital content has previously been based on scalable multimedia formats and transcoding. Due to scalability, it is possible to achieve several versions of any video content on the basis of quality, size and frame rate etc. from just one bitstream without requiring further computation by the usage device. Likewise, by using transcoding, various versions of a single content can be provided by using various video coding standards such that the user could choose the standard that best suits his need.

However, due to the large number of available video coding standards, it has become impossible to provide a single content in varying codecs while permitting each codec to render lower quality versions of the source in terms of frame rate, size etc. To solve this problem, MPEG-21 DIA provides a means of adapting multimedia content using the BSD tool set [57]. This tool set is composed of the bitstream syntax description, the bitstream syntax description language (BSDL) and the generic bitstream syntax schema (gBS Schema). This tool set can be used with any binary multimedia format and is also device independent.

In general, the syntax of the bitstream is described using the BSDL. The BSDL is an XML based language. The bitstream syntax generation system receives as input the binary bitstream and generates as output the BSD. This BSD is transformed on the basis of the XSLT which contains information provided by the user e.g. usage environment information, terminal capabilities, user preferences etc. The resulting transformed BSD is used by a generic processor to produce the adapted bitstream which must be conformant with the adaptation parameters provided by the XSLT stylesheet. Although the BSD is written based on the XML syntax, it may contain either parts of the actual bitstream data included within the XML elements or it may contain references to the bitstream data instead [57].

3.4 THE H.261 VIDEO CODER

The H.261 video encoder was adopted as a recommendation by ITU in 1990, prior to the standardization of MPEG-2 and the recommendation of H.262. It specifies a video encoding standard for video conferencing and video services for transmission over the Integrated Services Digital Network (ISDN) at p x 64kbps where values of p range between 1 and 28 [1, 4, 58]. This standard was primarily intended for use at video bitrates between 40kbps – 2Mbps [4, 58]. While the standard does not specify any particular encoder design, it does specify the syntax of the encoded bitstream and the decoder. Hence, the encoder should be designed in such a way as to be compatible with the standardized decoder design. To ensure that H.261 can operate in real time dual directional videoconferencing applications, the standard specifies a delay of less than 150ms [1, 4]. The H.261 encoder and decoder block diagrams are shown in the Figure 3.3.

Since H.261 is targeted for videophone and videoconferencing, motion compensation is optional and only defined in the forward direction. Although the standard provides for the use of both intra-coding and inter-coding, it leaves the decision open as to how the choice is made. The encoder bitstream is error protected by using forward error correction coding [4]. To enable a single standard to be used for both the 625 and the 525 line television standard, the H.261 standard supports both the Common Intermediate Format (CIF) and the quarter CIF (QCIF) picture formats [1, 4, 58]. The video coder also supports non interlaced pictures at an approximate frame rate of 29.97. The pictures are coded with one luminance and 2 color difference components, i.e. Y, C_B and C_R respectively. While the standard supports both CIF and QCIF picture scanning formats, the standard also specifies that all H.261 codecs must support QCIF while some can also support CIF [4]. The H.261 coder is capable of achieving a high coding efficiency because it can encode in groups of 30, 15, 10 or 7.5 fps as supported by CIF and QCIF thereby eliminating the need to individually encode and transmit whole frames [1].

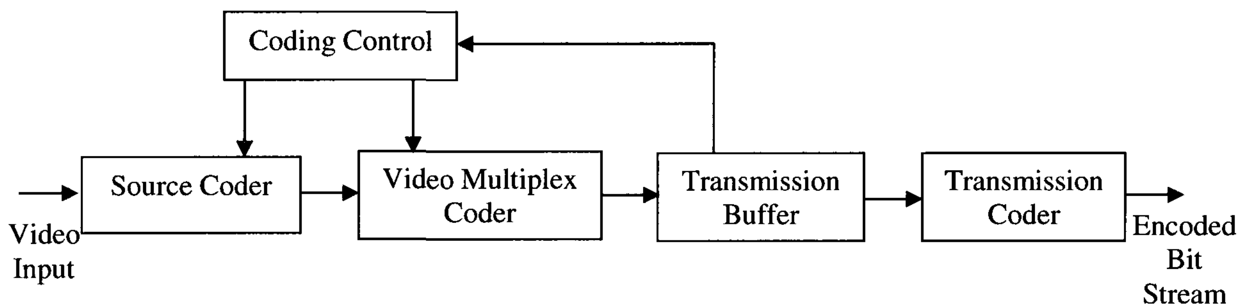

FIGURE 3.3 (a): ITU-T H.261 Video encoder block diagram.

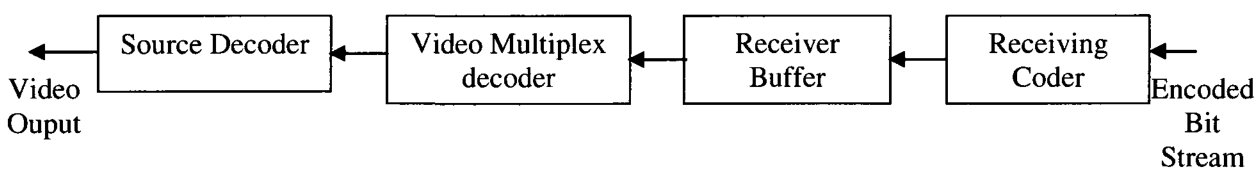

FIGURE 3.3 (b): ITU-T H.261 Video decoder block diagram.

Unlike images that only contain information in the spatial 2D domain, video signals contain information in both the spatial and temporal domains i.e. in 3 dimensions. Hence the H.261 video coder was modeled in both the spatial and the temporal domains [1]. A spatial encoder deals with blocks of 8 by 8 pixels while the temporal encoder deals with blocks of 16 by 16 pixels. Figure 3.4 shows the basic building blocks for the H.261 video compression system. Each building block has a specific function.

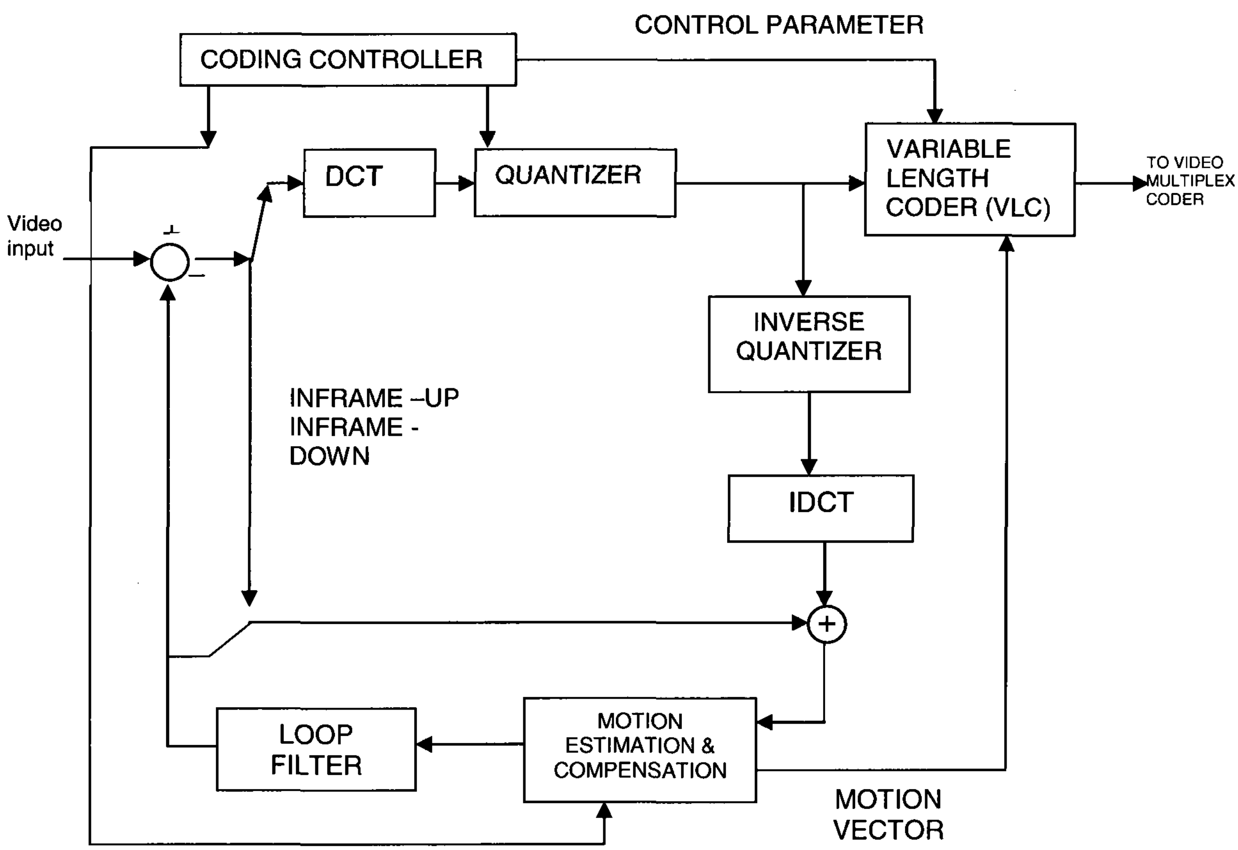

FIGURE 3.4: video compression system [1].

The system receives the source video as input. The spatial domain of the source is passed to a spatial operator which is a 2D linear transform. The most commonly used spatial operator for video compression is the 2D-DCT. Within the H.261 framework also, the spatial operator specified by standard is 2D-DCT. The DCT is a separable transform which operates on 8 by 8 pixel blocks, therefore, DCT is often the transform of choice because it is able to deliver real computations, is low in complexity and has relatively fast implementations [1, 2]. An overview of DCT is given in the next section. The outputs of the DCT process are transform coefficients which are sent on to the next block which is the quantizer.

The quantization operation is a lossy one which may significantly reduce bitrates. The quantization method employed is most often scalar, non-uniform quantization, where the step sizes are chosen based on the transform coefficient distribution. The next step is the variable length coding of the quantized transform coefficients. This is a lossless operation. These quantized transform coefficients are zigzag scanned using progressive coding. Following the zigzag scanning, entropy coding is done to generate a serial bitstream [1]. Specifically, the H.261 video compression standard uses a variable length Huffman code for this step in the video compression operation. The first two steps on the feedback loop are the inverse quantizer and the inverse DCT. The function of both steps is to reconstruct and store each frame. The frames stored by these blocks are used down the line by the motion estimator and the motion compensator block to generate the next prediction frame [1].

The H.261 video compression system provides for both intra frame coding and inter frame coding. Inter frame coding uses the feedback loop to generate a prediction error. The prediction error is calculated as the difference between the blocks of the current frame and the current prediction frame [1]. The motion estimation block operates on 16 by 16 pixel blocks and generates motion vectors for each 16 by 16 block. It is the responsibility of the motion compensation block on the other hand, to generate the prediction from the motion vectors and the previously reconstructed frame [1]. The intra frame mode plays two important roles [1]:

1. To ensure that an error is not continuously propagated i.e. error checking, the intra frame mode will spatially encode one current frame periodically, for example, every 15 frames.
2. When the inter frame mode is unable to meet its performance threshold, the intra frame mode will spatially encode a block.

3.5 DISCRETE COSINE TRAMSFORM (DCT)

DCT was first proposed in 1974 by Ahmed et al. [59]. Since its discovery, it has played a large role in the development of compression techniques. It is commonly thought of as Fourier-Cosine series performed in the discrete time domain. DCT is based on the decomposition of a source signal into smaller M by M blocks or M by M by M blocks for 2D or 3D coding respectively. Each of these blocks is then individually processed.

DCT derives its name from the fact that in its computation, rows of M by M transform matrices are obtained as functions of cosines. DCT is closely related to Discrete Fourier Transforms (DFT). In fact, DCT can be obtained from DFT by mirroring a 1D M-point sequence to create a 2M-point sequence, where the first M-points of the generated 2M-point DFT sequence is the DCT of that sequence. The DCT of an M by M pixel array is given as

$$F(u,v) = \frac{2}{M} C_u C_v \sum_{i=0}^{M-1} \sum_{j=0}^{M-1} f(i,j) \cos\left(\frac{(2i+1)u\pi}{2M}\right) \times \cos\left(\frac{(2j+1)v\pi}{2M}\right)$$

EQ 3.1

The corresponding inverse DCT (IDCT) is given as

$$f(i,j) = C_u C_v \sum_{u=0}^{M-1} \sum_{v=0}^{M-1} F(u,v) \cos\left(\frac{(2i+1)u\pi}{2M}\right) \times \cos\left(\frac{(2j+1)v\pi}{2M}\right)$$

EQ 3.2

Where $C_u = \frac{1}{\sqrt{2}}$ $for\ u = 0, C_u = 1\ Otherwise\ and$

$C_v = \frac{1}{2}$ $for\ v = 0, C_v = 1\ Otherwise.$

i and j represent the horizontal and vertical indices of the M by M spatial array while u and v represent the horizontal and vertical indices of the M by M coefficient array.

Overtime, a lot of international compression standards have been established based on DCT. In 1992, the Joint Pictures Expert Group (JPEG) was established as an international compression standard for image compression. The JPEG encoder and decoder are DCT based. In video coding also, MPEG video coding standards are also largely based on DCT e.g. H.261, MPEG-2, H.263 etc. DCT has been widely used in video coding because it is a unitary transform. Unitary transforms are used because they have an inverse operation which produces a good reconstruction of the input signal. In general, DCT is preferred to other transform coding techniques because it has a better energy compaction or coding gain than others in the transform domain [1, 60]. This is due to the fact that DCT takes advantage of the high degree of correlation that exists between adjacent pixels in a source signal.

Although DCT has garnered widespread approval and usage, it is not without shortcomings. The fact that a source needs to be sub-divided into M by M blocks often has the drawback of blocking artifacts in the reconstructed signal. Blocking artifacts occur due to the fact that correlation between blocks still exists across block boundaries. Blocking artifacts are especially more pronounced at low bitrates [61]. Another problem with DCT is that only the spatial correlation that exists between pixels in a block is considered while the correlation that exists between pixels in one block and the pixels of its neighboring blocks are ignored. Despite these shortcomings, DCT is still widely used because of its simplicity and performance.

3.6 METHODOLOGY AND IMPLEMENTATION

The proposed multiple description coding technique is composed of five distinct implementation stages. The first stage is made up of the video encoding process. The raw video signal is encoded using an H.261 encoder. The output of the first stage is an encoded bitstream. The next stage involves a detailed XML description of the encoded bitstream. This detailed description is known as the generic bitstream syntax description (gBSD). The third stage involves the proposed technique for splitting the compressed bitstream in such a way as to generate multiple descriptions. The output of this stage is two or more transformed gBSDs.

The next stage is the processing of the transformed gBSDs using the DIA engine. This processing results in the creation of binary files representing each description. The final stage involves decoding the binary files using an H.261 decoder and the corresponding post-processing phase. A detailed discussion of each implementation stage is given below. Figure 3.5 gives a general overview of the components of the proposed methodology.

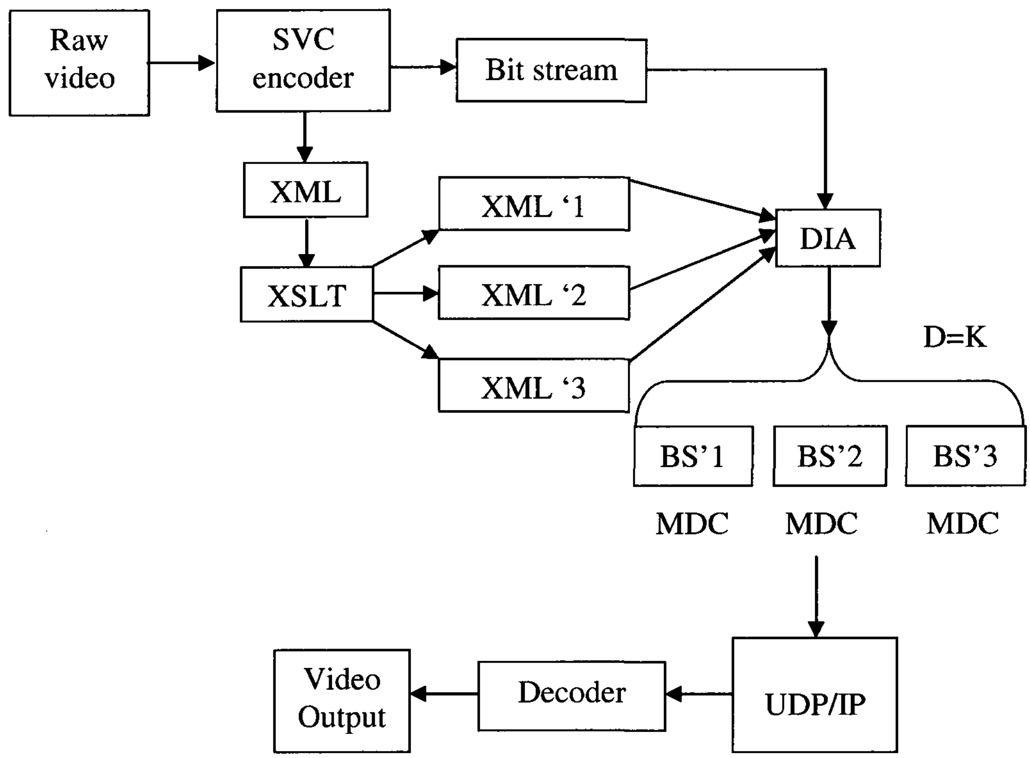

FIGURE 3.5: An Overview of the components of the MDC technique.

While network testing was not implemented, to enable the efficient transmission of the descriptions generated by our proposed technique, these descriptions should be transmitted using the user datagram protocol (UDP) as opposed to the transmission control protocol (TCP). TCP is a connection oriented and reliable protocol. It is said to be connection oriented because it ensures that both end processes have established a connection prior to the transfer of data. In order to ensure reliability, TCP ensures that transmitted packets are received error free. In the event of errors or losses, TCP enables re-transmission of the affected packets. Due to this feature, TCP is most often used for transmission over the internet. Other advantages of TCP include in-order delivery of transmitted packets, congestion control etc. UDP on the other hand is an unreliable, connectionless protocol. It is connection free since it can send information without ensuring that an end to end connection has been established. UDP uses a checksum technique in which either the UDP along with the IP header are delivered error free else they are lost [62]. Therefore, UDP provides no guarantee that transmitted packets would be received. However, due to the fact that UDP provides no in-order delivery, congestion control nor buffering, it has a low protocol overhead making it attractive for the transmission of time sensitive video communication [62].

We propose the use of UDP for the transmission of the MDC descriptions for several reasons including the fact that there are no transmission delays due to re-transmissions as is experienced in TCP. Also, UDP is an attractive protocol due to the fact that it has low protocol processing overhead [62]. Although it achieves this by sacrificing in-order delivery and re-transmissions, these are not required in MDC since, MDC descriptions are all of equal importance and can be received in any order by the decoder without any adverse effect on the reconstruction quality. Also, MDC is designed to be able to deal with unreliable transmission conditions, hence re-transmissions are not necessary. UDP is also preferred for the transmission of descriptions because TCP generally has a low performance over wireless networks [63]. Likewise, the UDP checksum does not pose a problem in MDC, specifically because a description in MDC is either received error free otherwise it is lost.

3.6.1 THE VIDEO ENCODING PROCESS

A raw video file was processed using an H.261 encoder as specified in Section 3.4. The H.261 video encoder supports both the CIF and the QCIF picture formats, therefore, the raw video could be in either format. The encoder received as input the raw video file and as specified by the video coding standard, each block generated sixty four coefficients composed of one DC coefficient and sixty three AC coefficients after the DCT transformation process. Generally, H.261 supports both inter and intra prediction coding. However, for the purpose of this work of research, the implementation of the H.261 video codec only supports intra coding. Intra coding is a special case of prediction coding where the predictor value is zero. Intra coding was chosen over inter coding not only because of computational simplicity, but because of the advantage of limited error propagation and easy transmission error concealment. In the case of intra coding, it is important to note that the value of the DC coefficients is always much larger in comparison with the AC coefficients.

Mismatch is not an issue in the proposed technique. Mismatch is a condition introduced when a signal used for prediction at the encoder is unavailable at the decoder. Due to the fact that this H.261 encoder implementation supports only intra coding, motion estimation was not used. Therefore, the problem of mismatch between the encoder and decoder, which is commonly related to motion estimated multiple description coders is avoided.

3.6.1.1 Video Multiplex Coder

The encoded bitstream was sent as an input to the video multiplex coder of the H.261 coder. The function of the video multiplex coder is to arrange the bitstream into a well defined hierarchy of layers. This video data hierarchy ensures that the video multiplex decoder can decode the video data with little or no ambiguity. The video multiplex coder of the H.261 coder defines a hierarchical data structure with four identified layers which are: Picture layer, Group of Blocks layer (GOB), Macroblock layer (MB) and Block layer [1, 4, 58]. Each layer has associativity with the preceding layer in that each layer is developed from the preceding layer and contains the payload associated with that particular layer as well as a header describing the parameters that are used to generate the bitstream [1].

The encoded bitstream was arranged from the bottom up as follows: Each block was composed of 64 coefficients. Each macroblock consisted of 4 luminance blocks and two sub-sampled color difference blocks. Each group of block consisted of thirty three macroblocks. Since the H.261 coder supports both CIF and QCIF, each picture layer in the case of CIF pictures was composed of twelve groups of blocks while in the case of QCIF pictures was composed of three groups of blocks. The structure of each layer is provided in appendix A.

3.6.2 GENERATION OF THE GENERIC BITSTREAM SYNTAX DESCRIPTION (gBSD)

At this point of the implementation process, the hierarchical, encoded bitstream has been obtained. The next step in the implementation process was generating an XML based description of this hierarchical bitstream on the bit level, thereby obtaining a generic bitstream syntax description. This process was implemented in C++. Adhering to the formatting rules of XML documents, a generic gBSD writer was used to write a standard gBSD header for the XML references to be used. This header contained all the namespaces that were to be used in the XML document. Next, a header specific to the compressed H.261 bitstream was written into the output XML document. This header contained information such as the classification alias for the bitstream, the address unit and the address mode etc.

Using the hierarchical breakdown of the intra coded H.261 bitstream provided by the video multiplex coder, the bitstream was described in terms of the position of each layer within the hierarchy. Based on the gBS Schema element, as shown in Figure 3.6, the hierarchical structure of gBSDs and gBSDUnits with their associated syntactical labels were used to describe the starting point of the elements of each layer and the corresponding binary length of each element within the bitstream.

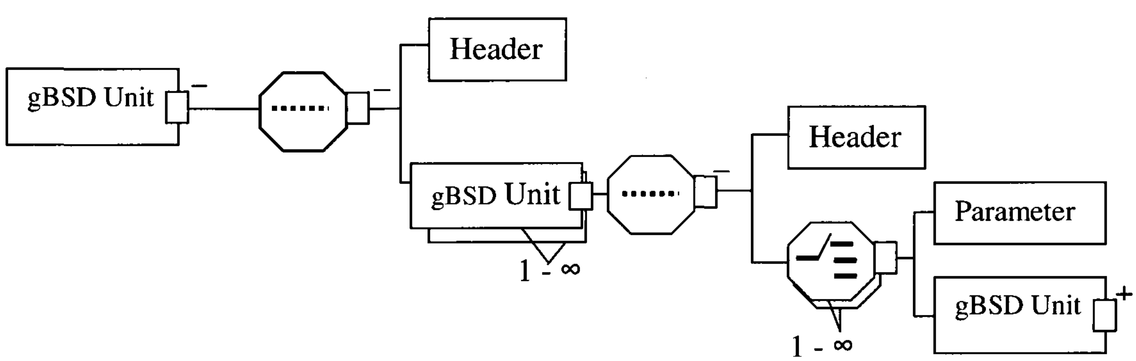

FIGURE 3.6: A gBS Schema Element Hierarchy.

To effectively describe all the coefficients of the block layer, parameter elements were used as opposed to gBSDUnit elements. In addition to the start and length attributes used by the gBSDUnit elements, the parameter elements also provided information about how each element within the block was coded using either variable length coding or fixed length escape codes. To provide this information, additional level, run and escape attributes were used. Instead of the syntactical label attribute used by gBSDUnit elements, a name attribute was used with parameter elements. Furthermore, to differentiate the DC coefficient from the AC coefficients within each block, a marker attribute was introduced. The length and start attributes of both the gBSDUnit and parameter elements were generally used in such a way that based on the length of the current node, the start value of the subsequent node was updated by the exact length value. The result of this stage of processing was a detailed XML based gBSD of the encoded bitstream. An excerpt of this description showing the first block of the first macroblock in the first group of blocks is shown in Figure 3.7.


```
<?xml version="1.0" encoding="UTF-8"?>
<dia:DIA                          xmlns:dia="urn:mpeg:mpeg21:2003:01-DIA-NS"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
      <dia:Description xsi:type="gBSDType">
<Header>
      <ClassificationAlias                                  alias=":H261"
href="urn:mpeg:mpeg4:graphics:cs:syntacticalLabels"/>
      <DefaultValues      addressUnit="bit"      addressMode="Absolute"
globalAddressInfo="Media/foreman.cmp"/>
</Header>
<gBSDUnit syntacticalLabel=":H261:PSC" start="0" length="20"/>
<gBSDUnit syntacticalLabel=":H261:TR" start="20" length="5"/>
<gBSDUnit syntacticalLabel=":H261:Ptype" start="25" length="6"/>
<gBSDUnit syntacticalLabel=":H261:PEI" start="31" length="1"/>
<gBSDUnit syntacticalLabel=":H261:GBSC" start="32" length="16"/>
<gBSDUnit syntacticalLabel=":H261:iGBID" start="48" length="4"/>
<gBSDUnit syntacticalLabel=":H261:mGQuant" start="52" length="5">
<Header>
<DefaultValues addressMode="Absolute"/>
```

```
</Header>
<Parameter    name=":H261:mGQuant[iGBID]"    start="52"    length="5"
gQuant="5"/>
</gBSDUnit>
<gBSDUnit syntacticalLabel=":H261:GEI" start="57" length="1"/>
<gBSDUnit syntacticalLabel=":H261:MBA" start="58" length="1"/>
<gBSDUnit syntacticalLabel=":H261:MTYPE" start="59" length="4"/>
<gBSDUnit   syntacticalLabel=":H261:Blocks"   start="63"   length="344"
GBID="1" MBID="1" BLOCKID="1">
<Header>
<DefaultValues addressMode="Absolute"/>
</Header>
<Parameter name=":H261:coeff" start="63" level="85" run="0" escape="0"
length="8" marker="DC"/>
<Parameter name=":H261:coeff" start="71" level="4" run="0" escape="0"
length="8" marker="AC"/>
<Parameter name=":H261:coeff" start="79" level="1" run="0" escape="0"
length="3" marker="AC"/>
<Parameter name=":H261:coeff" start="82" level="4" run="0" escape="0"
length="8" marker="AC"/>
<Parameter name=":H261:coeff" start="90" level="-3" run="0" escape="0"
length="6" marker="AC"/>
<Parameter name=":H261:coeff" start="96" level="-2" run="0" escape="0"
length="5" marker="AC"/>
<Parameter name=":H261:coeff" start="101" level="2" run="0" escape="0"
length="5" marker="AC"/>
<Parameter name=":H261:coeff" start="106" level="1" run="0" escape="0"
length="3" marker="AC"/>
<Parameter name=":H261:coeff" start="109" level="-1" run="0" escape="0"
length="3" marker="AC"/>
<Parameter name=":H261:coeff" start="112" level="8" run="0" escape="0"
length="13" marker="AC"/>
<Parameter    name=":H261:coeff"    start="125"    level="18"    run="0"
escape="000001" length="20" marker="AC"/>
<Parameter name=":H261:coeff" start="145" level="4" run="0" escape="0"
length="8" marker="AC"/>
<Parameter name=":H261:coeff" start="153" level="1" run="0" escape="0"
length="3" marker="AC"/>
<Parameter name=":H261:coeff" start="156" level="-1" run="0" escape="0"
length="3" marker="AC"/>
<Parameter name=":H261:coeff" start="159" level="-3" run="0" escape="0"
length="6" marker="AC"/>
<Parameter name=":H261:coeff" start="165" level="4" run="0" escape="0"
length="8" marker="AC"/>
. . . . .
</gBSDUnit>
<gBSDUnit syntacticalLabel=":H261:EOB" start="407" length="2"/>
```

Figure 3.7: An excerpt of the gBSD showing block 1 of macroblock 1 in group of blocks 1.

3.6.3 GENERATION OF THE TRANSFORMED GENERIC BITSTREAM SYNTAX DESCRIPTION (gBSD)

This was implemented as a three step process. The most important aspect of this process was the choice of a suitable splitting technique. This was paramount to the development of the multiple descriptions needed to fulfill the requirements of multiple description coding. Once the splitting technique had been chosen, the gBSD had to be transformed using a suitable transformation language on the basis of the transformation constraints specified by the splitting technique. The final step of this process involved parsing the gBSD to generate the transformed gBSD.

3.6.3.1 The Bitstream Splitting Process

Multiple description coding is based on the premise that it is possible to split a single source into two or more individually good descriptions, such that if losses occur during transmission, the source can still be reconstructed provided at least one description is received error free. Each additional description received improves the quality of the reconstruction. Necessarily, each description must have some redundancy information without which it would be impossible to reconstruct the source. Generally, each description is often a low quality version of the original. In this work of research, a splitting technique that fulfills all these requirements is proposed.

This splitting technique takes advantage of the DCT blocking used in H.261. During the H.261 encoding process, each video frame had to be sub-divided into 8 by 8 blocks which were discrete cosine transformed and quantized. Assuming that N descriptions are desired, the proposed technique indicates that each DCT component other than the DC component must be divided into the number of descriptions required.

Therefore, provided that two descriptions were desired, each AC coefficient was divided into two. Several techniques for splitting these coefficients were tested. Some of the techniques tested include odd and even splitting, odd and even splitting with zero padding, uniformly splitting both DC and AC coefficients and uniformly splitting only AC coefficients. The first technique tested was the odd and even splitting. However, due to the low quality reconstruction achieved, the odd and even splitting with zero padding was implemented so as to provide a possible improvement in quality. The quality that resulted was also not satisfactory. Largely, these techniques were rejected because of their performance using our proposed technique. Finally, to create two descriptions of equal importance with similar sizes, it was observed that each DCT component would need to be split into almost equal parts. However, rather than simply splitting each AC coefficient down the middle and rounding the resulting value, which would lead to a significant loss in quality of the reconstructed source, a more dynamic splitting technique was chosen. Given an AC value of 21, this value was split into two. In one description, the resulting value was floored such that the resulting value was 10, while in the second description, the ceiling of resulting value was obtained which equaled 11. By utilizing a splitting technique of this nature, the resulting central reconstruction of the source would in this case result in the pre-processing value of 21, thereby losing little or no information. This process resulted in two dependent descriptions, each containing about half the information of the original source. If any one of these two descriptions is received error free, a low quality version of the original source can be reconstructed.

3.6.3.2 The XSLT Transformation Process

Having proposed a splitting technique, the next step was to design a transformation process that could implement this splitting technique. The transformation language used is XSLT. The key transformation constraint identified by the proposed splitting technique required to obtain two descriptions is that the value of each AC coefficient must be split into two. Therefore, an XSLT code that satisfied this constraint was generated based on the gBSD of the compressed bitstream. The structure of the gBSD used was designed to enable this transformation process to be carried out with relative ease. The value of each DC and AC coefficient within the gBSD was stored using a level attribute. These two values were differentiated using a marker attribute. Therefore, the value stored in each level attribute with a corresponding 'AC' marker attribute was split into two as specified by the transformation constraint.

```
<xsl:template name="half_level_value" match="gbsd:gBSDUnit[@level]">
    <xsl:copy>
      <xsl:variable name="Updatelevel">
        <xsl:call-template name="half_level">
          <xsl:with-param name="a" select="2"/>
          <xsl:with-param name="level" select="@level"/>
          <xsl:with-param name="index" select="1"/>
        </xsl:call-template>
      </xsl:variable>
      <xsl:apply-templates select="@*"/>
      <xsl:attribute name="level">
        <xsl:value-of select="$Updatelevel"/>
      </xsl:attribute>
      <xsl:apply-templates select="node()"/>
    </xsl:copy>
  </xsl:template>
  <xsl:template name="half_level">
    <xsl:param name="nodes"/>
    <xsl:param name="index" select="1"/>
    <xsl:param name="a" select="2"/>
    <xsl:param name="level" select="@level"/>
    <xsl:variable name="Calclevel">
      <xsl:choose>
        <xsl:when test="$level and @marker='AC'">
          <xsl:value-of select="floor or ceiling(($level)div($a))"/>
        </xsl:when>
        <xsl:otherwise>
```

```
        <xsl:if test="$level and @marker='DC'">
          <xsl:value-of select="$level"/>
        </xsl:if>
      </xsl:otherwise>
    </xsl:choose>
  </xsl:variable>
  <xsl:value-of select="$Calclevel"/>
</xsl:template>
```

Figure 3.8: XSLT code excerpt showing the coefficient splitting process for two descriptions

Once each AC coefficient was split into the required number of descriptions, it became necessary to re-encode the newly calculated coefficient values. To prevent any errors, particularly since all the processing was done at the bit level, the binary DC coefficient values were also re-encoded. As specified by the H.261 standard, the DC and AC coefficients had to be encoded in different ways. This made it necessary to use two separate lookup tables for each coefficient type. Due to the detailed description of the binary stream provided by the gBSD, we were able to distinguish between the DC and the AC coefficients as previously stated using a marker attribute. The forward DC re-encoding was done simply using 8 bit codes as specified by the standard. However, the forward re-encoding for AC coefficients was done using the variable length code and the escape code specified by the H.261 standard.

The forward re-encoding of the AC coefficients was not as easily implemented as the DC encoding. The gBSD specified within each parameter element which AC levels were previously encoded using the variable length codes and which ones were encoded using the fixed length escape codes. However, after splitting each AC value by the number of descriptions desired and either obtaining the ceiling or floor of the resulting coefficient value, the value of each AC coefficient was reduced by at least half its previous value in the situation where two descriptions were desired. Three situations resulted: first, some level values were reduced to zero. Next, some level values previously encoded using escape codes could now be coded using the variable length codes. Finally, some level values which prior to processing were encoded using the variable length table, needed to be re-encoded as escape codes.

Intuitively, for the first situation, where the value of an AC coefficient after processing resulted in zero, the value of the run attribute of that particular coefficient was increased by one, and provided that the subsequent level attribute value was not zero, the run attribute for the ensuing AC coefficient was updated by the exact value of its preceding sibling. This preceding sibling with the zero level attribute was subsequently removed from the gBSD. Due to the fact that the value of all AC coeffients were split, some values that were previously encoded using escape codes no longer required a 20 bit escape code and were re-encoded using variable length codes. Finally, the forward run length coding that was required to deal with situations were post processing level values resulted in zero which led to new level/ run combinations that had to be encoded using escape codes. This was a direct result of the fact that these level/ run combinations were no longer within the variable length codes.

These three conditions resulted in significant bit savings. For example, where escape codes were re-encoded using variable length codes, this resulted in at least a gain of 6 bits. Also, where level values resulted in zero, they were subsequently removed thereby leading to bit savings. Even under conditions where zero level values resulted in the conversion of variable length codes into escape codes, there was still an overall reduction in the number of bits required to encode the resulting level/ run sequence. To quantitatively measure the exact number of bits saved after processing, it was necessary to provide a means for updating the values of the start and length attributes within the gBSD after forward re-encoding had been implemented. Therefore, within the XSLT code, provisions was made to update the length in the gBSD such that it reflected the number of bits used to encode the recalculated level value while simultaneously changing the start values to reflect the new length.

3.6.3.3 The gBS Description Transformation Process

At this point, the gBSD of the bitstream had been obtained and the XSLT code implementing all the facets of the splitting constraint had been developed. The next phase of the proposed multiple description coding technique involved parsing the gBSD and the XSLT code using a suitable parser.

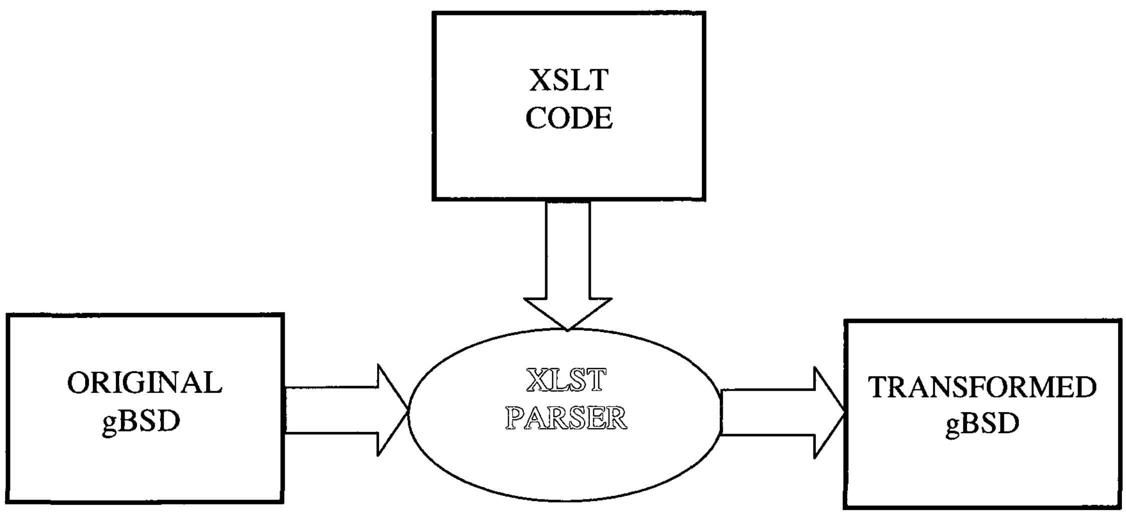

FIGURE 3.9: The processing of the original gBSD to the transformed gBSD

As shown in the Figure above, the parser received as input both the XML description of the bitstream as well as the XSLT code that specified what updates the parser had to implement within the XML description. The output of this stage of processing was another XML document that implemented all the splitting constraints. Assuming that the number of required descriptions (N) is two, the output XML document satisfied the following constraints:

1. It divided the level of each AC component by two. For the first description, the resulting coefficient values were floored while for the second description, the ceiling of the resulting coefficient values was obtained.
2. There was a removal of all AC level components whose post processing values equaled zero.
3. There was forward DC re-encoding and
4. There was forward AC re-encoding, including escape - to - variable length coding and variable length - to - escape coding.

This resulting XML document that satisfied all these constraints is known as the transformed gBSD. An excerpt of the original gBSD prior to transformation and the resulting transformed gBSDs for both descriptions are shown in Figures 3.10 and 3.11 (a) and (b) respectively.

```
<gBSDUnit   syntacticalLabel=":H261:Blocks"   start="1225"   length="84"
GBID="1" MBID="1" BLOCKID="4">
<Header>
<DefaultValues addressMode="Absolute"/>
</Header>
<Parameter   name=":H261:coeff"   start="1225"   level="79"   run="0"
escape="0" length="8" marker="DC"/>
<Parameter   name=":H261:coeff"   start="1233"   level="-2"   run="0"
escape="0" length="5" marker="AC"/>
<Parameter   name=":H261:coeff"   start="1238"   level="-2"   run="0"
escape="0" length="5" marker="AC"/>
```

```
<Parameter   name=":H261:coeff"   start="1243"   level="-2"   run="1"
escape="0" length="7" marker="AC"/>
<Parameter   name=":H261:coeff"   start="1250"   level="-1"   run="0"
escape="0" length="3" marker="AC"/>
<Parameter   name=":H261:coeff"   start="1253"   level="-2"   run="0"
escape="0" length="5" marker="AC"/>
<Parameter   name=":H261:coeff"   start="1258"   level="-2"   run="0"
escape="0" length="5" marker="AC"/>
<Parameter   name=":H261:coeff"   start="1263"   level="-2"   run="0"
escape="0" length="5" marker="AC"/>
<Parameter   name=":H261:coeff"   start="1268"   level="-1"   run="0"
escape="0" length="3" marker="AC"/>
<Parameter   name=":H261:coeff"   start="1271"   level="-1"   run="0"
escape="0" length="3" marker="AC"/>
<Parameter   name=":H261:coeff"   start="1274"   level="-2"   run="0"
escape="0" length="5" marker="AC"/>
<Parameter   name=":H261:coeff"   start="1279"   level="-1"   run="0"
escape="0" length="3" marker="AC"/>
<Parameter   name=":H261:coeff"   start="1282"   level="-2"   run="0"
escape="0" length="5" marker="AC"/>
<Parameter   name=":H261:coeff"   start="1287"   level="-1"   run="0"
escape="0" length="3" marker="AC"/>
<Parameter   name=":H261:coeff"   start="1290"   level="-1"   run="0"
escape="0" length="3" marker="AC"/>
<Parameter   name=":H261:coeff"   start="1293"   level="-1"   run="0"
escape="0" length="3" marker="AC"/>
<Parameter   name=":H261:coeff"   start="1296"   level="-1"   run="0"
escape="0" length="3" marker="AC"/>
<Parameter   name=":H261:coeff"   start="1299"   level="-1"   run="0"
escape="0" length="3" marker="AC"/>
<Parameter   name=":H261:coeff"   start="1302"   level="-1"   run="7"
escape="0" length="7" marker="AC"/>
</gBSDUnit>
<gBSDUnit syntacticalLabel=":H261:EOB" start="1309" length="2"/>
```


Figure 3.10: An excerpt of the gBSD showing a block prior to transformation.


```
<gBSDUnit syntacticalLabel=":H261:Blocks" start="1225" length="84"
GBID="1" MBID="1" BLOCKID="4">
<Header>
<DefaultValues addressMode="Absolute"></DefaultValues>
</Header>
<Parameter name=":H261:coeff" start="1225" level="79" run="0"
escape="0" length="8" marker="DC">
<value xsi:type=":bt:b8">01001111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1233" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1238" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
```

```
<Parameter name=":H261:coeff" start="1243" level="-1" run="1"
escape="0" length="4" marker="AC">
<value xsi:type=":bt:b4">0111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1253" level="-1" run="1"
escape="0" length="4" marker="AC">
<value xsi:type=":bt:b4">0111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1258" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1263" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1274" level="-1" run="2"
escape="0" length="5" marker="AC">
<value xsi:type=":bt:b5">01011</value>
</Parameter>
<Parameter name=":H261:coeff" start="1282" level="-1" run="1"
escape="0" length="4" marker="AC">
<value xsi:type=":bt:b4">0111</value>
</Parameter>
</gBSDUnit>
<gBSDUnitsyntacticalLabel=":H261:EOB"start="1309"length="2"></gBSDUnit>
```

Figure 3.11 (a): **An excerpt of the gBSD showing the block after transformation in the first description (ceiling).**

```
<gBSDUnit syntacticalLabel=":H261:Blocks" start="1225" length="84"
GBID="1" MBID="1" BLOCKID="4">
<Header>
<DefaultValues addressMode="Absolute"></DefaultValues>
</Header>
<Parameter name=":H261:coeff" start="1225" level="79" run="0"
escape="0" length="8" marker="DC">
<value xsi:type=":bt:b8">01001111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1233" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1238" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1243" level="-1" run="1"
escape="0" length="4" marker="AC">
<value xsi:type=":bt:b4">0111</value>
</Parameter>
```

```
<Parameter name=":H261:coeff" start="1250" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1253" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1258" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1263" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1268" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1271" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1274" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1279" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1282" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1287" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1290" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1293" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1296" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
<Parameter name=":H261:coeff" start="1299" level="-1" run="0"
escape="0" length="3" marker="AC">
<value xsi:type=":bt:b3">111</value>
</Parameter>
```

```
<Parameter name=":H261:coeff" start="1302" level="-1" run="7"
escape="0" length="7" marker="AC">
<value xsi:type=":bt:b7">0001001</value>
</Parameter>
</gBSDUnit>
<gBSDUnitsyntacticalLabel=":H261:EOB"start="1309"length="2"></gBSDUnit>
```

Figure 3.11 (b): An excerpt of the gBSD showing the block after transformation in the second description (floor).

3.6.4 THE DIA ADAPTATION PROCESS

The aim of this phase was to generate two or more adapted bitstreams based on the transformed gBSD. This was implemented using the DIA engine. The DIA engine received as input the encoded bitstream and the transformed gBS descriptions. The bitstream generated during the encoding process was used as a template. The header information for each frame was largely left untouched and copied over to the adapted bitstreams. However, the quantizer for the group of blocks had to be modified based on the number of descriptions desired. For instance, where the number of descriptions desired was two, the quantizer value was doubled. This reflected the fact that the AC coefficient values had been split into two.

The DIA engine was used to parse through the transformed gBSDs and all the sections of the bitstream corresponding to gBSDUnits were copied directly from the encoded bitstream into the adapted bitstream. The most vital changes in the bitstreams for the proposed technique were done at the block level where the AC coefficients were split. Therefore at the block level, all the sections of the bitstreams corresponding to parameter elements were replaced by the new binary values for each coefficient. These binary values were obtained from the parameter elements of the transformed gBSDs and were copied into the adapted bitstreams. This resulted in two or more new bitstreams which are known as the adapted bitstreams. The user of the adapted bitstream is the video decoder.

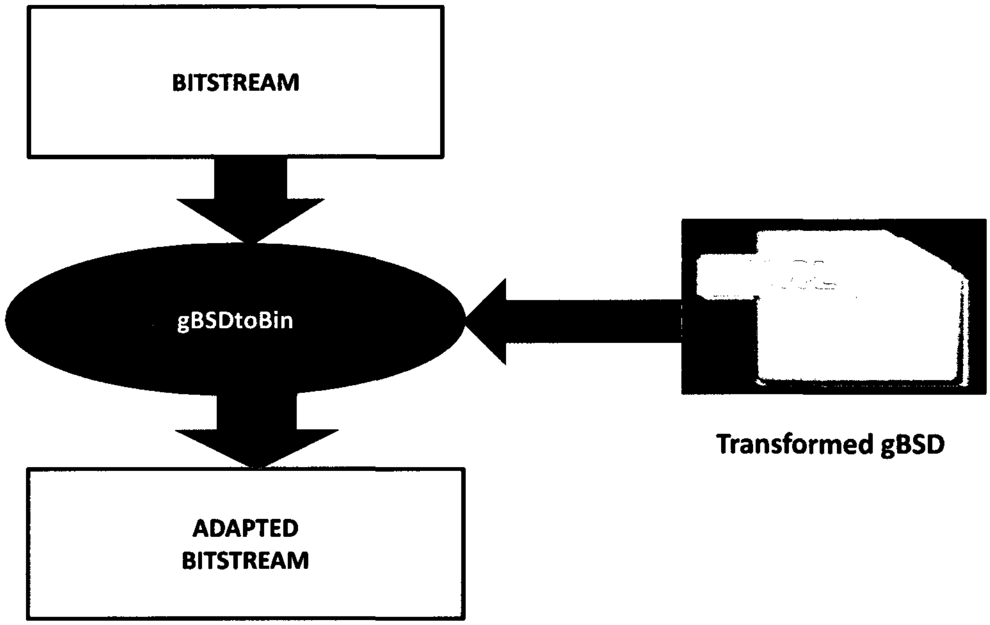

FIGURE 3.12: Adapted bitstream generation using a gBSDtoBin parser.

3.6.5 THE VIDEO DECODING PROCESS

This was the final processing stage needed to reconstruct the encoded video. Therefore using an H.261 decoder, the adapted bitstreams were decoded. To decode the side descriptions, each side decoder received the adapted bitstream as input. The bitstream was first passed into a video multiplex decoder. This was used to decode the video multiplex arrangement. This was necessary to identify the hierarchical layers within the bitstream from the picture layer to the block layer. The decoders received the resulting bitstreams and processed them through the inverse quantizer. The inverse quantization of the bitstreams had to be modified to accommodate the splitting of the block coefficients. This was done prior to passing the XML descriptions through the DIA engine. The quantization value used to encode the video was stored in a parameter element in the picture header. A parameter element was used as opposed to a gBSDUnit element to allow the quantization value to be modified as needed, based on the number of descriptions desired i.e. N. Therefore, in the XSLT code, when N=2, and the original quantization value used to encode the video was 5, the quantization value used to decode the split bitstream was double the encoder quantization value i.e. 10. The inverse quantization stage was followed by inverse DCT, thereby generating the reconstructed side descriptions.

```
<gBSDUnit syntacticalLabel=":H261:mGQuant" start="52" length="5">
<Header>
<DefaultValues addressMode="Absolute"/>
</Header>
<Parameter    name=":H261:mGQuant[iGBID]"    start="52"    length="5"
gQuant="5"/>
</gBSDUnit>
```

Figure 3.13: An excerpt of the gBSD showing the quantizer prior to transformation

```
<gBSDUnit syntacticalLabel=":H261:mGQuant" start="52" length="5">
<Header>
<DefaultValues addressMode="Absolute"></DefaultValues>
</Header>
<Parameter    name=":H261:mGQuant[iGBID]"    start="52"    length="5"
gQuant="10">
<value xsi:type=":bt:b5">01010</value>
</Parameter>
</gBSDUnit>
```

Figure 3.14: An excerpt of the gBSD showing the quantizer after transformation.

The approach taken by the proposed technique to generate the central description from the received side descriptions is different from the approach generally used in multiple description decoders. More often than not, in the two description MD coder for example, the two correlated side bitstreams are individually decoded to form the side descriptions. At the central decoder, the two bitstreams are jointly decoded to generate the central description which is a combination of the two side descriptions. However, in the proposed technique, the two side description bitstreams are also individually decoded for both side descriptions, while at the central decoder, the side description bitstreams are decoded and the result summed up to obtain the central reconstruction.

CHAPTER 4

EXPERIMENTATION AND RESULTS

4.1 INTRODUCTION

A video signal is a three dimensional sequence of frames. Each frame within this sequence is composed of an array of pixels. Each pixel is composed of 8 bits of information. Therefore, the array of pixels representing a three dimensional sequence of frames will require a significant amount of bandwidth. The aim of multiple description coding is to represent a video sequence for transmission in a manner that ensures that a side description which is a coarse representation of the original image is received after transmission with the least amount of redundancy and inherent distortion possible while at the same time utilizing the least amount of bandwidth during transmission.

The proposed multiple description video coding technique while adhering to the basic requirements of multiple description coders, is more focused towards enabling easy adaptability and universal multimedia access within the framework of MPEG-21. The goal of this proposed technique is content generation that can withstand content delivery across error prone networks while simultaneously maximizing the user experience by adapting the multimedia content to fit the usage environment. Based on the characteristics of the transmitting network in terms of bandwidth, the choice of the level of decomposition can be determined i.e. a decision about how many times to split a video source can be made. However, testing the network performance of this proposed technique is outside the scope of this research.

This implementation provides a generic format for the description of the encoded bitstream. This allows for a more format independent adaptation of the bitstream that does not require the adaptation engine to have a full knowledge of the specific BS Schema. The use of the gBSD of the digital item enables the DIA engine to adapt the bitstream without a need to know what kind of media is being processed. The DIA engine simply generates the adapted bitstream.

In this Chapter, the results of the proposed technique are presented and explained. The performance of this multiple description video coding technique is examined using a few digital video quality assessment methods. The advantages and drawbacks of the technique are presented, along with the applicability of the proposed technique.

4.2 IMPLEMENTATION

Based on the methodology presented in Chapter three, the proposed multiple description video coding technique was implemented. In the video multiplex hierarchical structure, the highest layer is the picture layer which is followed by the group of blocks layer. The third layer in the hierarchy is the macroblock layer. Each macroblock is composed of four luminance blocks and two color difference blocks. Video standards use the YUV or YCbCr systems to represent a video sample. The luminance information is represented by Y while the two chrominance color differences are represented by Cb and Cr. However, for the purpose of this work of research, the video encoding/decoding process was implemented with a focus majorly on the luminance information of the video sample using only the four luminance blocks in each macroblock.

The proposed MDC technique was first implemented for the standard two descriptions MDC case. The splitting of the bitstream using the gBSD and the XSLT transformation constraints resulted in two new transformed gBSDs which contained the binary values of each of the split AC coefficients. This transformed gBSDs were adapted using the DIA engine resulting in two new bitstreams which were representative of the two side descriptions. These bitstreams produced the side descriptions which were decoded by the side decoders. When either of these bitstreams is received at the decoder, a coarse reconstruction of the original bitstream can be obtained. Figures 4.1 (a) and (b) show the first frame from the reconstructed side descriptions of the foreman video sequence. Figure 4.2 on the other hand shows the first frame from the reconstructed central description of the foreman video sequence formed when both descriptions from the two description MDC are received.

FIGURE 4.1 (a): First Side Reconstruction of the two description MDC (ceiling).

FIGURE 4.1 (b): First Side Reconstruction of the two description MDC (floor).

FIGURE 4.2: Central Reconstruction of the two description MDC.

This was then generalized and implemented to develop more descriptions. The proposed multiple description technique was implemented for the cases of three, four and five side descriptions. The resulting reconstruction for each of these implementations is shown in Figures 4.3 (a) and (b), Figures 4.4 (a) and (b) and Figures 4.5 (a) and (b).

FIGURE 4.3 (a): First Side Reconstruction of the three description MDC (ceiling).

FIGURE 4.3 (b): First Side Reconstruction of the three description MDC (floor).

FIGURE 4.4 (a): First Side Reconstruction of the four description MDC (ceiling).

FIGURE 4.4 (b): First Side Reconstruction of the four description MDC (floor).

FIGURE 4.5 (a): First Side Reconstruction of the five description MDC (ceiling).

FIGURE 4.5 (b): First Side Reconstruction of the five description MDC (floor).

4.2 VIDEO QUALITY ASSESSMENT TECHNIQUES

Overtime, several techniques have been proposed to measure the quality of digital video. These assessment techniques are required to measure the quality of the reconstruction against the original video. During the encoding/decoding process as well as during other implementation processing steps required to generate coarse side descriptions of the original video source, distortion is introduced by noise or blocking artifacts. These often results in the degradation of the video sequence. This degradation may result in perceptible visual distortions which affect the quality of the reconstructed video. Quality assessment techniques are used as a yardstick to measure this difference in quality. As expected, there is typically a tradeoff between the efficiency of any multiple description coding technique and the reconstruction quality of either the side descriptions or the central description.

Typically, video quality assessment techniques can either be objective or subjective in nature. Objective quality assessment techniques make use of both quantitative and mathematical methods to measure the quality of videos while subjective quality assessment techniques are based on the human visual perception of quality. While there have been several techniques implemented in both classes, no one quality assessment technique is self sufficient for assessing the quality of a reconstructed video. Therefore, a full assessment of the quality of a video is often obtained using a combination of two or more techniques across both objective and subjective categories of assessment techniques.

4.2.1 SUBJECTIVE ASSESSMENT OF THE QUALITY OF VIDEO SEQUENCES

This class of quality assessment techniques is based on the human visual system. Usually, the quality of the video samples is determined based on the amount of degradation perceived by the human eye. Although the human eye is unable to distinguish one frame from a sequence of frames in fast progression, it is very sensitive to distortions observed in the quality of the video sequence. This quality assessment technique is referred to as subjective because it relies on the quality of the video as observed by the human visual system, the result of which is seldom the same when a single video sequence is observed independently by two or more individuals. To make this quality assessment technique more objective in nature, a group of observers can be used to view and rate the quality of the video sequence using a set of numbers. The average of these numbers taken across all the observers could then be used to measure the quality of the video. The human visual system is generally more sensitive to luminance than it is to chrominance [64]. For this reason, while the chrominance components can be represented at a low spatial resolution, the luminance components must be represented at a higher spatial resolution. Within the framework of multiple description coding, each side description must necessarily be dependent for it to be individually good [14], therefore a level of redundancy must be included during the processing of each side description. As a rule of thumb, the higher the amount of redundancy introduced into each side description during multiple description coding, the higher the quality of the reconstruction. However, the amount of redundancy introduced should not be significantly higher than what is required to achieve a coarse reconstruction of the original video sequence. Therefore, it is necessary to consider the visual perception of quality when designing a multiple description coding algorithm.

During the implementation phase of this work of research, the proposed multiple description technique was implemented with a focus on the luminance components of the video. Therefore, the reconstructed video had to be represented at a high enough resolution. In terms of the level of redundancy introduced, the higher the number of descriptions achieved from a single source, the lower the amount of redundancy contained in each of the resulting descriptions. However, the value of the DC coefficient in the side descriptions had to be left unchanged from that of the central description because the information contained in the DC coefficient is essential to ensure that the reconstruction produced by each side decoder is of an acceptable quality, albeit a coarse representation of the original source. This proposed multiple description coding algorithm was implemented on the bit level. Therefore, it was very sensitive to errors since an error in only one bit of the bitstream or the losses due to the processing of coefficient values had a significant effect on the reconstruction and introduced an unacceptable amount of degradation.

Applying this subjective quality assessment technique to the results generated by the proposed MPEG-21 based multiple description technique shows two individually good reconstructed side descriptions for the two description implementation as shown in Figure 4.1 although the quality achieved by flooring the resulting values as opposed to obtaining the ceiling values was higher. Therefore, comparing the reconstruction obtained from the central decoder which is shown in Figure 4.2 to the reconstructions shown in Figure 4.1, it is evident that the side description is a coarse representation of the central description, although the side reconstructions are also individually good. Both reconstructions show the distortion introduced by blocking artifacts which was as a result of the fact that the DCT based H.261 video coding standard was employed for the encoding/decoding process. However, it can be observed that the blocking artifacts are more pronounced in the side reconstructions than they are in the central reconstruction. In terms of the richness of the details of the foreman video, the performance of the side and the central reconstruction of the proposed technique were in keeping with the levels of redundancy introduced. Although the proposed technique handled the original video sequence on the bit level, and made significant changes to most of the bitstream, there were no significant distortions introduced due to any bit error. As expected, due to the fact that for the two description case, the AC coefficients in the reconstructed side descriptions only had approximately half the luminance information contained in the central description reconstruction, the side description was darker than the central description reconstruction. When Figures 4.1, Figure 4.2 and Figures 4.3 are compared, the perceived difference in luminance as observed visually is not indicative of the actual difference in luminance that can be quantitatively measured.

4.2.2 OBJECTIVE ASSESSMENT OF THE QUALITY OF VIDEO SEQUENCES

This class of quality assessment techniques is the converse of the subjective assessment techniques because while the later is dependent on the human visual perception, the objective assessment techniques avoid human based quality assessment. Rather, objective quality assessment techniques are based on quantitative mathematical methods that measure the amount of distortion or loss of information present in a reconstruction in comparison with the original video source. This class of techniques includes mean square error (MSE), mean absolute error (MAE) and peak signal to noise ratio (PSNR) etc.

For a video frame A and the reconstructed video frame $\hat{A}$, with frame size of N x M, pixel indices $1 \leq i \leq M$ and $1 \leq j \leq N$, and given that the number of bits per pixel is n, the equations for MSE, MAE and PSNR are as follows [1]:

$$MSE = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} [A\,(i,j) - \hat{A}\,(i,j)]^2 \qquad EQ-4.1$$

$$MAE = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} [A\,(i,j) - \hat{A}\,(i,j)] \qquad EQ-4.2$$

$$PSNR = 20\, log_{10} \left(\frac{2^n}{MSE^{1/2}} \right) \qquad EQ-4.3$$

The most commonly used of these techniques is PSNR. It is computed using the MSE. The MSE is used to calculate the distance between the original video frame and the reconstructed video frame. While these objective quality assessment techniques are useful, they often do not give a true representation of the quality of a video. Ultimately, the qualities of videos are best determined by the human observer [1]. Therefore, the PSNR value may not always reflect the perceived visual quality of a human observer. This discrepancy between the observed quality and the calculated quality of videos may be due to the fact that the computation of the PSNR does not take into consideration the sensitivity of the eye to motion and contrasts within the video sequence. Also, there may be inherent distortions within the video frame that the eye may not perceive, but which a quantitative technique like PSNR would detect. Since PSNR is more sensitive to distortions and impairments within a video source, it is possible for the PSNR value to indicate a significant level of distortion in the quality of the video which may be visually pleasing to the observer.

The quality metric for the proposed MPEG-21 based multiple description coding technique was computed for the reconstructed side and central descriptions where the number of descriptions was two. The MSE and PSNR values obtained are as shown in Table 4.1. The MSE and PSNR values were also computed for the side description reconstructions obtained when the foreman video sample was decomposed into 2, 3, 4 and 5 descriptions. These values are as shown in Tables 4.2 and 4.3.

TABLE 4.1: The MSE and PSNR values for the proposed two description MD Coder

FOREMAN VIDEO SAMPLE	MSE	PSNR
SIDE DESCRIPTION RECONSTRUCTION-floor	55.98	30.65
SIDE DESCRIPTION RECONSTRUCTION-ceiling	216.36	24.77
CENTRAL DESCRIPTION RECONSTRUCTION	37.88	32.35

It was observed that the PSNR achieved for the side description in the two description multiple description coder was significantly lower than the value achieved for the central reconstruction. Also, the PSNR values for the side descriptions differed in that flooring the values and keeping them lower resulted in better PSNR overall than obtaining the ceiling of the coefficient values. These results were due to the fact that coefficient values in the side description were significantly reduced compared to the value of the coefficients in the central description. The splitting of these values and especially obtaining the ceiling values as opposed to the floor values led to a loss of information especially around block edges. During the process of splitting the AC coefficient values, non integer coefficient values were either floored or the ceiling of the values obtained which also resulted in loss of information in the side descriptions.

However, at the central decoder, when both side descriptions are received, the value of the PSNR is significantly increased due to the fact that the AC coefficients from both descriptions are summed up, thereby increasing the amount of information available to the decoder and approximately, exact reconstructions were obtained. This was significant because the addition of information in the ceiling side description was compensated for by the removal of information in the floor side descriptions. Therefore the distortion that was introduced by dividing the coefficient values during the splitting process was compensated for in the central reconstruction.

TABLE 4.2: The MSE values from the side reconstructions of the proposed MD Coder at different levels of decomposition

FOREMAN VIDEO SAMPLE	MSE- Ceiling	MSE-Floor
2 Descriptions	216.56	55.98
3 Descriptions	325.10	139.70
4 Descriptions	433.04	314.72
5 Descriptions	654.25	557.92

TABLE 4.3: The PSNR values from the side reconstructions of the proposed MD Coder at different levels of decomposition

FOREMAN VIDEO SAMPLE	PSNR- Ceiling	PSNR-Floor	Average PSNR
2 Descriptions	24.77	30.65	27.71
3 Descriptions	23.01	26.68	25.46
4 Descriptions	21.77	23.15	22.46
5 Descriptions	19.97	20.67	20.39

A comparison of the PSNR values achieved when the foreman video was split into two to five descriptions shows the decrease in the quality of the reconstructed side descriptions achieved. It can therefore be concluded that although the proposed technique allows a single source to be split into as many descriptions as desired, the more the level of decomposition, the lower the quality of the reconstructed side descriptions. Intuitively, this is to be expected because while the side reconstructions at a level of decomposition of two, contains approximately half the information of the source, the side reconstructions at the level of decomposition of three only contains approximately one third the information of the source.

The quality of the central description is not as adversely affected by splitting a source into an increasing number of descriptions. This is due to the fact that provided all the descriptions are received, the central decoder reconstruction of the source will always result in a sum of the individual AC coefficients, thereby multiplying the information available to the central decoder by the total number of descriptions received. However, the distortion due to processing would also increase the higher the number of descriptions to be processed. This ultimately only resulted in minute decrease in the PSNR values of the central reconstruction.

Wang et al. [65] have proposed a comparable multiple description coding technique. Their proposed technique is multiple description coding based on the odd and even splitting of DCT coefficients coupled with zero padding. For this technique, the H.264 video coder was used for the compression process. However, rather than using the 2D-DCT for the transformation stage, 1D-DCT was used to transform each column within the video frame. To add a level of redundancy, zero padding was implemented in the DCT domain. In general, zero padding in the time domain leads to interpolation in the frequency domain. Likewise, the converse is also true: zero padding in the frequency domain leads to interpolation in the time domain. Therefore, padding zeros along the vertical axis in this technique led to an increase in sample size. After the 1D-DCT process, the IDCT processing was done. To obtain two new descriptions using the proposed quincunx sub-sampling technique, pixels in the even row and even column comprised one description while the other description was composed of pixels in the odd row and odd column. This sub-sampling led to the creation of two independently decodable side descriptions. At the central decoder the DCT process was re-implemented to get rid of the zeros used to pad the side descriptions. The resulting sample was processed using IDCT thereby generating the central reconstruction.

When only one description is received, this proposed technique uses prediction to generate the missing description from the received description. The advantages of this implementation include better coding efficiency since the H.264 video coding standard was used. Also, the zero padding process leads to an increase in quality for the side reconstructions. However, this increase in quality of the side reconstruction is sometimes accompanied by an associated decrease in the quality of the central reconstruction especially at high bitrates. This is due to the fact that zero padding leads to a decrease in bit rate since more zeros are also added by the quantization process. Ultimately, less non-zero coefficients were available to be coded. Therefore, it can be concluded that there is a trade off in quality between the central and the side reconstructions for this technique, particularly at higher bitrates.

To provide a suitable comparison between this zero padding technique and the MPEG-21 based MDC technique proposed by this work of research, the result generated when both techniques were applied to a QCIF foreman video sample at a fixed frame rate of 30 frames per second are compared. In general, better results were expected from the proposed zero padding technique since a more efficient video coding standard was used, also because the 1D-DCT was used as opposed to the 2D-DCT used in our implementation. However, this was not necessarily the case.

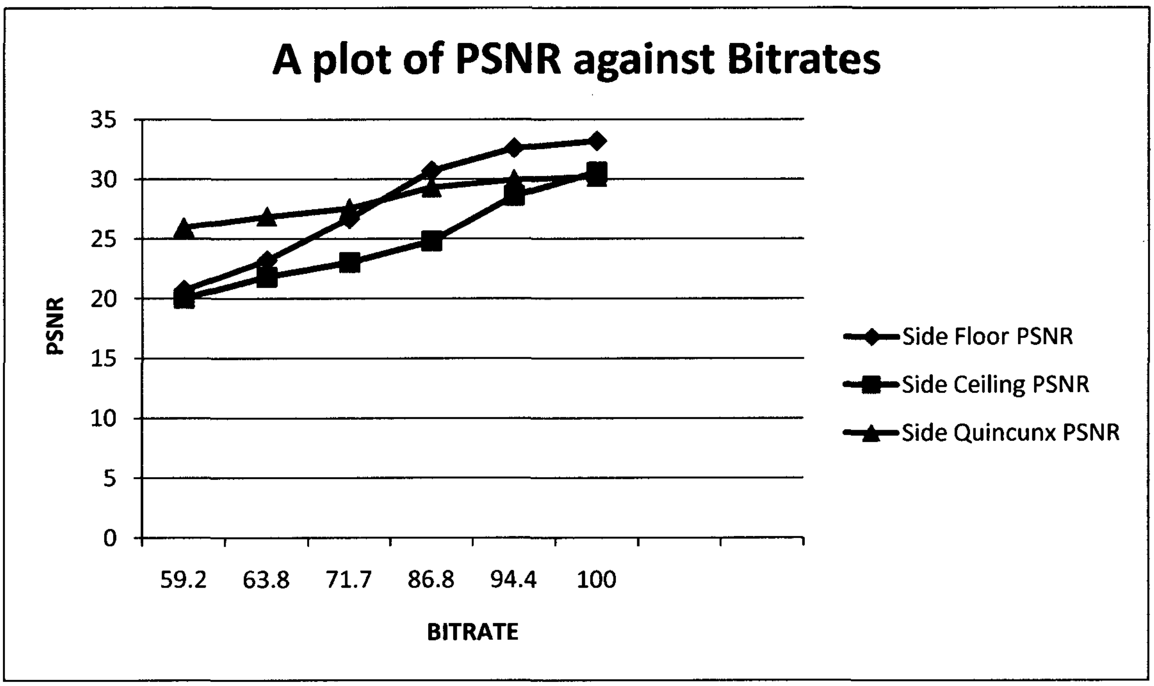

Figure 4.6: Quality comparison between the zero padding MDC technique and the proposed MPEG-21 based MDC technique.

As shown in the Figure above, their quincunx side description generally outperformed our ceiling side description especially at low bitrates. Likewise, their implementation outperformed our floor description at low bitrates. At high bitrates however, our floor description outperformed the proposed 1D-DCT MDC implementation. There was comparable performance of both central description reconstructions when fewer zeros were used to pad the descriptions. However, as more zeros were used, there was a decrease in the central quality of the zero padding technique in comparison to the MPEG-21 based MDC technique proposed by this work of research. This results show that even though an H.261 video coder was used as opposed to the H.264 video coder, the 2D-DCT was used rather than the 1D-DCT implementation and our splitting process was implemented outside the encoder, the technique proposed by this work of research still provided some comparable results which would improve provided that a video coder with better coding efficiency such as H.264 was used. It should be noted however, that our ceiling side description was generally not comparable, particularly at low bitrates. While providing some comparable quality results, our propose technique also have the added advantage of being able to generate any number of descriptions from a single bitstream without any need to re encode the bitstream.

4.3 ACHIEVABLE CODING EFFICIENCY

The goal of video compression technologies is to remove redundancies from a source, thereby reducing the bitrate used to encode that source. The degree to which the encoding process reduces the number of bits required to represent a signal is known as coding efficiency [1]. Typically, compression of a video source is achieved during the encoding process; nevertheless, in the implemented multiple description coding technique, the side descriptions had to be further compressed to decrease the size of the side descriptions in comparison to the central description. This compression was achieved within the gBSD by specifying a splitting constraint in the XSLT code that not only splits the bitstream as needed, but does this in a way that the number of bits required to represent the new bitstream is significantly reduced.

As described in Section 3.6.3.2, three processing stages during the splitting process resulted in significant bit savings. Whenever the value of the AC coefficient obtained after the splitting and processing evaluated to zero, the value of its associated run attribute was updated by one. Given that the next AC coefficient following this AC coefficient was not zero, the associated run value of this coefficient was updated by the run value of the preceding coefficient. However, in a situation where all the coefficients following a zero level AC coefficient value were also zero level coefficients, then these zero level coefficients were eliminated and the end of block element was set into position. This resulted in significant bit savings. Next, owing to the fact that coefficient values were split and rounded, escape codes which previously required 20 bits needed to be re-encoded using variable length codes with a maximum length of 14 bits. The converse was also true. The process of updating the run value of some coefficients necessitated the re-encoding of some coefficients by switching variable length codes with escape codes. Although this required using a 20 bit code to represent this new level/ run combination, the bits already saved by eliminating zero level coefficients, ultimately resulted in using a reduced number of bits to represent a set of coefficients.

To calculate the number of bits required to represent the new side description, the length attributes for each AC coefficient within the transformed gBSDs were updated with the new bit values. Ulitmately, the start values were also recalculated thereby obtaining the total number of bits used to represent the side descriptions.

TABLE 4.4: The Coding Efficiency achieved by the floor side reconstruction of the proposed MD Coder at different levels of decomposition

FOREMAN VIDEO SAMPLE	ENCODED BITRATE (bits)	DECODED BITRATE (bits)	CODING EFFICIENCY (%)
Original sample	125 600	-	-
2 Descriptions	125 600	86841	144.63
3 Descriptions	125 600	71708	175.15
4 Descriptions	125 600	63777	196.94
5 Descriptions	125 600	59217	212.10

TABLE 4.5: The Coding Efficiency achieved by the ceiling side reconstruction of the proposed MD Coder at different levels of decomposition

FOREMAN VIDEO SAMPLE	ENCODED BITRATE (bits)	DECODED BITRATE (bits)	CODING EFFICIENCY (%)
Original sample	125 600	-	-
2 Descriptions	125 600	87043	144.30
3 Descriptions	125 600	71888	174.72
4 Descriptions	125 600	63914	196.51
5 Descriptions	125 600	59386	211.50

Finally, coding efficiency is defined as the ratio of the encoded bitrate with the decoded bitrate [1].

$$Coding\ Efficiency = (Compression\ Ratio)^{-1} \qquad EQ-4.4[1]$$

$$= {Encoded\ Bitrate}/{Decoded\ Bitrate} \qquad EQ-4.5\ [1]$$

Therefore, Tables 4.4 and 4.5 show the encoded bitrate, the decoded bitrate, as well as the resulting coding efficiency achieved for both the ceiling side description and the floor side description at varying levels of decomposition using the foreman video sample. As shown in the Figure below, the bitrate used to encode both the ceiling and the floor side descriptions indicate that both descriptions are balanced in size.

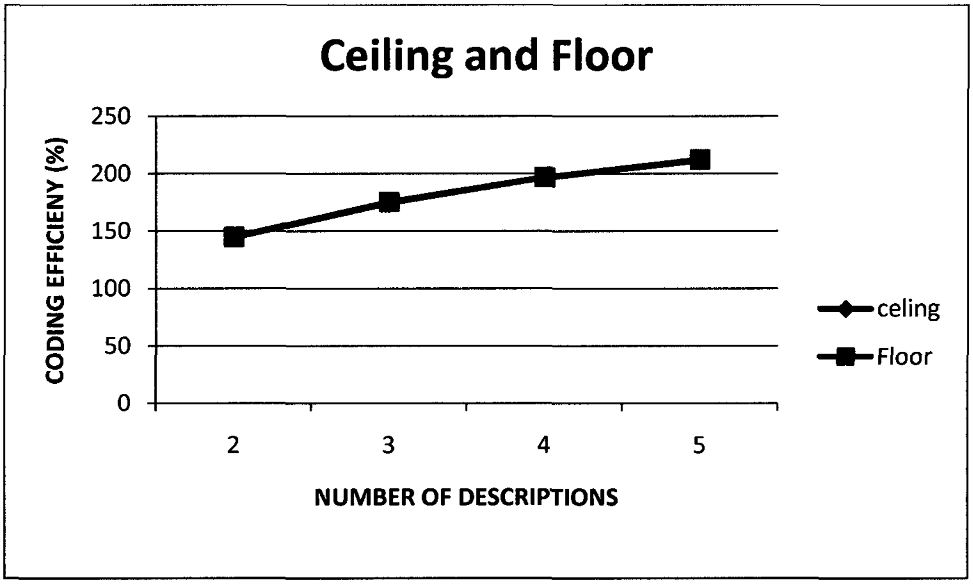

Figure 4.7: Coding Efficiency Performance Showing Balanced Side Reconstructions.

It can be observed that the higher the number of times a single source is decomposed, the higher the coding efficiency achieved. Similarly, the higher the number of times a single source is decomposed, the lower the quality of the side reconstruction achieved. However, Figures 4.8 and 4.9 show that there is a significant decrease in the number of bits required to represent the side descriptions resulting from splitting a source into three compared to the number of bits needed to represent the side descriptions resulting from splitting a source into two. Therefore, there is an associated increase in coding efficiency. As the number of side descriptions generated increases from three through five, the gain in coding efficiency is not as significant as observed previously.

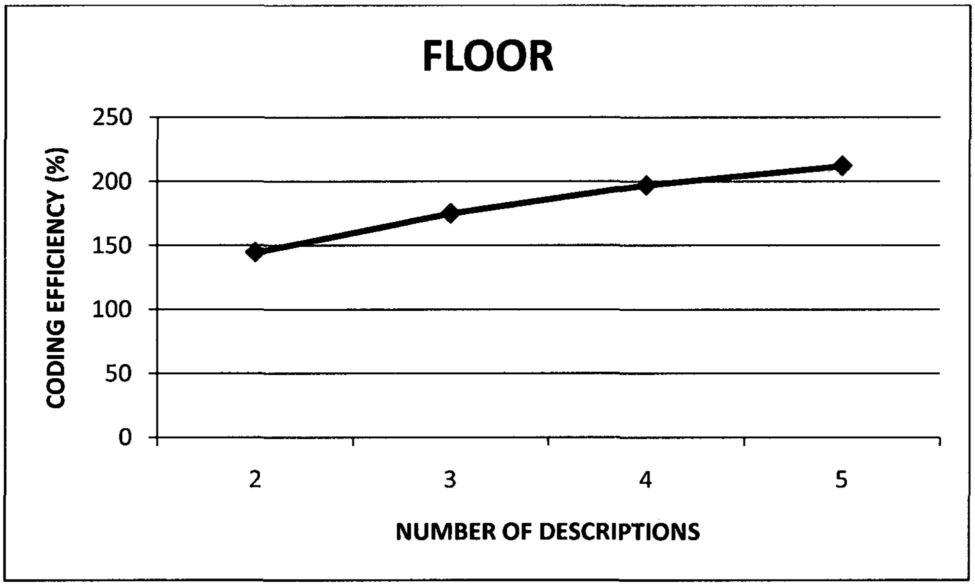

Figure 4.8: Coding Efficiency Performance for the Floored Side Reconstructions.

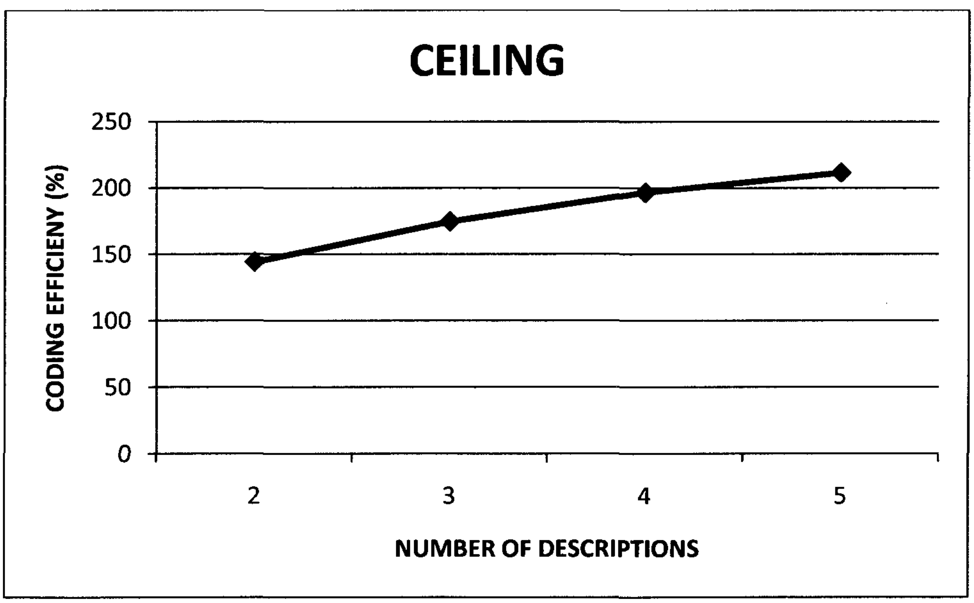

Figure 4.9: Coding Efficiency Performance for the Ceiling Side Reconstructions.

Furthermore, Figure 4.10 shows that as the number of side descriptions generated increases from two to five, the loss in quality of the side description achieved is significant.

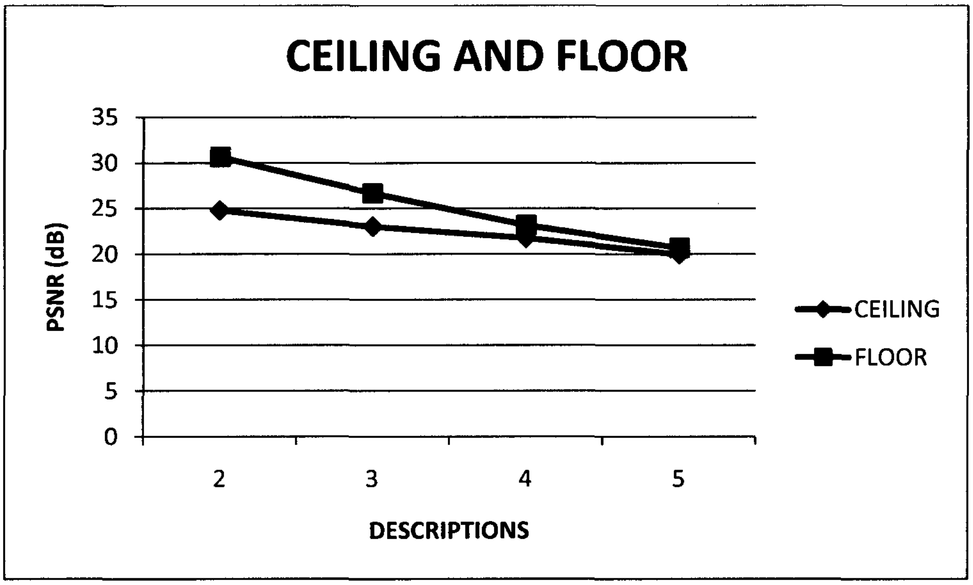

Figure 4.10 Quality Performances for the both Side Description Reconstructions.

Consequently, it can be concluded that although there are significant advantages to be achieved by splitting a source into two or more descriptions, as the number of descriptions increases, there is a decrease in the advantages to be obtained from further splitting the source to generate more descriptions given that only one of the side descriptions is received. However, as the number of description received increases, there is significant gain to be achieved by further splitting a source. This is further bolstered by the quality of reconstruction achieved when one or more descriptions were dropped. These results are shown in Table 4.6.

While actual network testing of the proposed MDC technique was not implemented, the quality achieved by the proposed technique when one or more of the descriptions were not received at the central decoder was simulated. The results of this simulation are as shown in Table 4.6. In the three description implementation for example, considering that one of the three descriptions is lost, and two of these are received; provided that one of the descriptions received is a floor description while the second description received is a ceiling description, a quality of 28.57 dB was achieved, while when all three descriptions were received, a quality of 32.30 dB was achieved.

TABLE 4.6: The Quality achieved by the central reconstructions of the proposed MDC when only some of the descriptions are received.

	1 Description Received	2 Description Received	3 Description Received	4 Description Received	5 Description Received
2 Descriptions	30.65 ------------- 24.77	32.35	N/A	N/A	N/A
3 Descriptions	26.68 ------------- 23.01	28.57 ------------ 1F & 1C	32.30	N/A	N/A
4 Descriptions	23.15 ------------ 21.77	25.68 ------------ 1F & 1C	28.80 ------------- 1F & 2C	32.09	N/A
5 Descriptions	20.67 ------------- 19.97	23.22 ------------ 1F & 1C	26.30 ------------- 1F & 2C	29.02 ------------- 2F & 2C	31.93

For the five description implementation however, when only one floor and one ceiling description were received by the decoder, a quality of 23.22 dB was achieved. Provided that three of the five descriptions were received, and two of those descriptions were ceiling descriptions while one was a floor description, a quality of 26.30 was also achieved. Likewise, where four of the five descriptions were received, given that two of the received descriptions were ceiling descriptions while the other two were floor descriptions, a PSNR of 29.02 dB was achieved by the reconstruction. Finally, when all five descriptions were received, a PSNR of 31.93 was achieved. As expected, the achieved results show that the larger the number of descriptions received, the more improved the quality of the reconstruction.

Regardless, there is a tradeoff between quality and coding efficiency. For example, given that a server is requested to send information across two different networks. The first network is a reliable one while the other is unreliable. To ensure that the information is received after transmitting across the reliable network, the server may only need to split the information into two descriptions. However, to ensure that the information is received after transmitting across the unreliable network, the server may need to split the information into four to five descriptions. Therefore, it can be observed that based on the particular application, a determination should be made about the number of descriptions to be generated.

The choice about the number of descriptions to generate in this proposed technique is based not only on network characteristics but also on user preferences. The user may desire multimedia content based on their particular device capability or the application for which the media content is desired. For example, a user who requires video strictly for surveillance purposes may not have stringent quality requirements but rather may be more concerned about ensuring that at least a coarse reconstruction is possible at the decoder whereas, for a user, such as a consumer of cable TV, while it is likewise important that the media content is received, such a user will have higher quality requirements. Hence, the media content may need to be split into two independent descriptions thereby, ensuring the reconstruction quality at the decoder. On the other hand, mobile clients may require lower resolution reconstructions on the basis of their receiving devices particularly, since there is a limit on the permitted quality of multimedia supported by mobile devices e.g. a quality of 352 x 288 (CIF) or higher is not generally allowed on mobile devices [43]. Although network testing was not implemented, in order to make a decision about the number of descriptions to generate based on network characteristics, it is possible to estimate the available network bandwidth by monitoring the size of packets sent across a given network as well as the amount of transmission time associated with such packets [42]. To make information about user preferences and network conditions readily available to the intermediate node used by our proposed implementation, this information can be stored in a usage environment description (UED) [42].

4.4 APPLICABILITY OF THE PROPOSED MPEG-21 BASED MULTIPLE DESCRIPTION CODING TECHNIQUE

Multiple description coding is not a newly discovered technology and has been implemented in several ways using several techniques as discussed in Chapter two. However, MDC has not often been implemented using MPEG-21 adaptation tools. This may be as a result of the fact that the MPEG-21 technology is still relatively new and the full range of its capabilities is yet to be discovered. Usually, the process of dividing the source into two or more descriptions is implemented during the encoding process. Invariably, every time new descriptions are to be developed, the source has to be re-encoded. While this technique has been largely successful and has yielded good quality side descriptions, it is cost and time intensive and cannot be implemented dynamically. Due to the increased need to provide universal multimedia access in an adaptable way, alternatives to the established MDC techniques have become essential.

The proposed MDC technique is relevant because it provides an adaptable and innovative way to achieve post encoding distribution within the frame work of MDC. The aim of this work of research is to enable video content to be adapted to fit the network and usage environment. Several solutions have been explored by other researchers. One of these solutions is based on the service provider making available a single media content using various coding standards, and at varying quality ranges and packet sizes. Thereby, based on the network and the usage environment characteristics, the appropriate version of the media content can be requested from the service provider. Another solution previously explored is instantaneous transcoding. Once again, the burden of implementing this solution rests squarely on the shoulders of the service provider. Instantaneous transcoding requires that an intermediate node capable of implementing the encoding/decoding process be located between the service provider and the users. Depending on the user's request, this intermediate node encodes the content using the desired coding standard, at the required quality and size. While these solutions have received some attention, they have not been widely successful due to their large bandwidth requirement along with their cumbersome and cost intensive implementation. The proposed MPEG-21 based MDC technique provides a viable alternative to these previously explored solutions while simultaneously reducing the cost and bandwidth required to implement this solution.

This technique has succeeded in providing an adaptive means for splitting the encoded bitstream outside of the encoding process thereby reducing the cost and time required to implement the multiple description coding technique. While there are advantages to our implementation, there is also the associated drawback of a decrease in quality compared to other MDC techniques were the splitting process was implemented within the encoder. However, the ability to split the encoded bitstream outside of the encoding process is an advantage that opens up a whole vista of possibilities for post encoding distribution of video content. The proposed technique also provides a significant advantage in environments where there are more stringent limitations on bandwidth availability.

Chen et al. [35] have proposed the use of a channel status detection technique to obtain channel status information for use in roaming. Based on the channel status information received, a decision is made about how data is transmitted. Using this proposed technique, in cases where network conditions are optimal, MDC is not utilized and data transmission is done over a single channel. However, if the channel is less than optimal, MDC along with multipath transport is utilized. Due to the fact that multipath transport is used here, the actual quality of each channel is not required; rather, the average or relative quality of both channels is used to determine how to reliably transmit the data.

For the proposed MPEG-21 based MDC technique, an active probing technique similar to the technique proposed in [35] can be used by the server to obtain information about the number of descriptions to generate from a single source based on the network conditions. To obtain network capabilities and conditions over the client network, the intermediate node can send out individual probe description sizes for a set of MDC splits e.g. a set of descriptions can be sent from 2, 3, 4 or 5 levels of decomposition. These descriptions can be sent at intervals of half a second or one second over this network. These probe descriptions would contain information such as departure time stamp as well as the sequence number of each particular description within the transmitted set. To obtain an accurate picture of the quality of the client's network, it is essential that the clocks of the intermediate node and the client are synchronized. Provided that these clocks are synchronized and the client receives the probe descriptions, an acknowledgement for each description received is sent back to the intermediate node. Based on the number of acknowledgements received, the intermediate note can obtain the packet loss rate (PLR). Likewise, based on the round trip time (RTT), the associated delay on the client's network can be accurately obtained. By associating an RTT / PLR information pair to each level of decomposition, the information received by probing each client's network can be used to make an informed decision about the number of times the original source needs to be split to enable the content to be transmitted between the two communicating networks. For example, if a minimum / maximum RTT of "a / b" and a minimum / maximum PLR of "b / c" is associated with splitting the source into three descriptions, when the probe information received from the client network indicates an RTT and PLR pair within this range, the source is split into three descriptions and transmitted over this client's network. At the new size for the side descriptions, the receiving host can accept all the side descriptions and reconstruct the source at little or no loss in quality compared to if the entire information had been sent in a single packet.

This ability of the proposed splitting technique to adapt itself to the current network characteristics makes it suitable for use in static downloadable media such as Video-on-Demand and TV-on-Demand. Media on demand is a system that allows the streaming of media content to a receiving device or the downloading of the media content on to the receiving device. In the case of media streaming, the media can be instantaneously viewed while the downloadable media content requires the media content to be stored on the receiving device for future viewing. Due to the large bandwidth requirement of the current implementations of media on demand, this technology has only garnered widespread usage in cable TV. This is because cable TV service providers can make use of the large bandwidth available on the cable system to transmit the media content to users. However, for consumers of satellite TV, the associated bandwidth limitations have made the implementation of media on demand relatively impossible. Nevertheless, using the proposed MPEG-21 based MDC splitting technique, the associated bandwidth limitations can be overcome to enable the implementation of media on demand even within satellite TV. Although the proposed technique is not perfect and there are some limitations on the achievable quality of the reconstructed side descriptions, its ability to provide adaptation of a video source is an advantage that can be used in several applications.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

The aim from the onset of this work of research was to create a splitting technique for use in multiple description video coding. This splitting technique had to provide an adaptive way to implement multiple description coding in a way that promotes universal multimedia access. We propose an MPEG-21 based splitting technique for use in MDC. This proposed technique employs the strong adaptation tools of MPEG-21 DIA. To implement the proposed technique, we utilized the H.261 video coding standard to encode the video sequence. This encoding was done using the intra coding prediction mode of the H.261 video coding standard. The intra-frame coding mode was used to implement a special case of predictive multiple description coding where the predictor value is zero. This afforded limited error transmission from one frame to another as well as computational simplicity because it eliminated the use of motion compensated prediction.

The basic idea behind this technique was to describe the encoded bitstream in a very detailed way using the generic bitstream syntax description tool of MPEG-21. This description was produced alongside the encoded bitstream using an XML gBSD writer. This proposed technique was set apart from other implementations of multiple description coding because we were able to introduce a source splitting technique that is not implemented within the encoding phase, thereby reducing the computational complexity and increasing the flexibility and adaptability of the implementation.

Key to the implementation of this proposed technique was specifying the constraints for splitting and re-encoding the source using XSLT. This proved to be quite challenging due to the many limitations of XSLT. Owing to the fact that XSLT is a declarative and not a procedural language, most of the rules of traditional programming languages are not applicable. Ultimately, these constraints were successfully implemented within the XSLT framework. This XSLT code was used to transform the XML based gBSD thereby generating the transformed gBSDs. Using the bitstream generated by the encoding process along with the transformed gBSDs as input, a DIA engine was used to generate the adapted bitstreams that were representative of the side descriptions.

The performance of this proposed technique was evaluated using the Peak-Signal-to-Noise-Ratio as well as a measure of the coding efficiency achieved. As the level of decomposition increased, there was an associated decrease in the quality of the reconstructed side descriptions and an associated increase in the achieved coding efficiency. The proposed technique achieved comparable qualities to some of the qualities attained by the zero padding MDC design where the splitting was done within the encoder, although in some cases it did not perform as well. However, due to the quality of the reconstructed side descriptions accomplished, coupled with the decrease in computational complexity and the increased flexibility achieved, the proposed technique is a viable alternative and in some cases, a suitable replacement for previously implemented multiple description coders. Also, the fact that most implemented multiple description coders cannot be implemented in an adaptable way based on the current network conditions makes the proposed technique suitable for post encoding distribution of video content.

5.2 FUTURE WORK

Bringing the strong adaptive capabilities of MPEG–21 to bear on the error resilience capabilities of MDC is a combination that has great potential within the video coding framework. In order to achieve better overall results using this technique, a more suitable splitting technique with higher performance capabilities at lower redundancies should be researched.

While it was possible to use XSLT to implement the splitting constraints for the gBSD, it's short comings greatly hampered and limited the scope of the possible splitting techniques that could be implemented. Using a $C^{\#}$ XML parser provides a possible alternative to XSLT for use in the transformation of XML descriptions and should be considered as a possible replacement for XSLT within the proposed technique.

To implement the encoding phase of this research, the H.261 video coding standard was used. Although this provided the structural hierarchy within the encoded bitstream desired for the implementation, the use of other video coding standards could be researched to harness their stronger compression efficiency. Furthermore, the use of a DCT based video coder coupled with the block based transformation used by the proposed technique made the issue of blocking artifacts an important consideration in the reconstruction quality of the side distortion. Therefore the use of non-DCT based video coders should also be explored.

REFERENCES

[1] A. Bovik. *Handbook of Image and Video Processing*, 2nd Edition. Burlington, MA. : Elsevier Academic Press, 2005

[2] A. Norkin, A. Gotchev, K. Egiazarian and J. Astola. "Low-Complexity Multiple Description Coding of Video Based on 3D Block Transforms," *EURASIP Journal on Embedded Systems*, vol.2007, Article ID 38631, January 2007

[3] V. Goyal. "Multiple Description Coding: Compression Meets the Network," *IEEE Signal Processing Magazine*, vol.18, pp.74-93, September 2001

[4] International Telecommunication Union ITU-T Recommendation H.261, Video Codec for Audiovisual Services at P x 64 Kbits, March 1993

[5] R. Iqbal, S. Shirmohammadi and C. Joslin. "MPEG-21 Based Temporal Adaptation of Live H.264 Video," In *Proceedings of the 8th IEEE International Symposium on Multimedia*, December 2006, pp.457-464

[6] ISO/IEC 21000-7:2004, Information Technology – Multimedia Framework – Part 7: Digital Item Adaptation

[7] H. Kosch. *Distributed Multimedia Database Technologies Supported by MPEG-7 and MPEG-21*. Boca Raton, Fla.: CRC Press, 2004

[8] N.S. Jayant. "Subsampling of a DPCM Speech Channel to Provide Two 'Self – Contained' Half–Rate Channels," *Bell Systems Technical Journal*, vol.60, no.4, pp.501-509, April 1981

[9] A. Gersho. "The Channel Splitting Problem and Modulo-PCM Coding," *Bell Labs Memo for Record*, October 1979

[10] S.E. Miller. "Fail-Safe Transmission without Standby Facilities", *Bell Labs, Technical Report*, TM80 – 136-2, August 1980

[11] Y. Wang, A.R. Reibman, and S. Lin. "Multiple Description Coding for Video Delivery," In *Proceedings of the IEEE*, vol.93, no.1, January 2005, pp.57-70

[12] R. Ma and F. Labeau. "Error-Resilient Multiple Description Coding," *IEEE Transactions on Signal Processing*, vol.56, no.8, pp.3996-4007, August 2008

[13] T. Cover and J. Thomas. *Elements of Information Theory.* NewYork, NY: John Wiley and Sons, 1991

[14] A.A. El Gamal and T. Cover. "Achievable Rates for Multiple Descriptions," *IEEE. Transactions on Information Theory*, vol.28, pp.851-857, November 1982

[15] M. Khansari, A. Zakauddin, W.-Y. Chan, E. Dubois and P. Mermelstein. "Approaches to Layered Coding for Dual-Rate Wireless Video Transmission," In *Proceedings of IEEE International Conference on Image Processing*, vol.1, November 1994, pp.258-262

[16] H. Radha, Y. Chen, K. Parthasarathy, and R. Cohen. "Scalable Internet Video Using MPEG-4," *Signal Processing: Image Communication*, vol.15, no.1-2, pp.95-126, September 1999

[17] U. Horn, K. Stuhlmuller, M. Link, and B. Girod. "Robust Internet Video Transmission Based on Scalable Coding and Unequal Error Protection," *Signal Processing: Image Communication*, vol.15, no.1-2, pp.77-94, September 1999

[18] J. Chakareski, S. Han, and B. Girod. "Layered Coding vs. Multiple Descriptions for Video Streaming Over Multiple Paths," In *Proceedings of the 11th Association for Computing Machinery International Conference on Multimedia*, November 2003, pp.422-431

[19] A. Reibman, H. Jafarkhani, Y. Wang, M. Orchard, and R. Puri. "Multiple Description Coding for Video Using Motion Compensated Prediction," In *Proceedings of IEEE International Conference on Image Processing*, vol.3, October 1999, pp.837-841

[20] D. Comas, R. Singh, and A. Ortega. "Rate-Distortion Optimization in a Robust Video Transmission Based on Unbalanced Multiple Description Coding," In *Proceedings of IEEE 4th Workshop on Multimedia Signal Processing*, October 2001, pp.581-586

[21] Y. Wang and S. Lin. "Error Resilient Video Coding Using Multiple Description Motion Compensation," In *Proceedings of IEEE 4th Workshop on Multimedia Signal Processing*, October 2001, pp.441-447

[22] Y.-C. Lee and Y. Altunbasak. "A Collaborative Multiple Description Transform Coding and Statistical Error Concealment Method for Error Resilient Video Streaming Over Noisy Channels," In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, vol.2, May 2002, pp.2077-2080

[23] Y. Wang, M.T. Orchard, V. Vaishampayan, and A.R. Reibman. "Multiple Description Coding Using Pairwise Correlating Transforms," *IEEE Transactions on Image Processing*, vol.10, pp.351-366, March 2001

[24] A. Ingle and V.A. Vaishampayan. "DPCM System Design for Diversity Systems with Application to Packetized Speech," *IEEE Transactions on Speech and Audio Processing,* vol.3, no.1, pp.48–58, January 1995

[25] D. Chung and Y. Wang. "Multiple Description Image Coding Using Signal Decomposition and Reconstruction Based on Lapped Orthogonal Transforms," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.9, no.6, pp.895–908, September 1999

[26] I. V. Bajic and J. W. Woods. "Domain-Based Multiple Description Coding of Images and Video," *IEEE Transactions on Image Processing*, vol.12, no.10, pp.1211–1225, October 2003

[27] D. Chung and Y. Wang. "Lapped Orthogonal Transforms Designed for Error Resilient Image Coding," In *Proceedings of the International Conference on Image Processing*, vol.3, September 2000, pp.368-371

[28] Y. Wang and D. Chung. "Non-hierarchical Signal Decomposition and Maximally Smooth Reconstruction for Wireless Video Transmission," In *Mobile Multimedia Communications*, D. Goodman and D. Raychaudhuri, Eds. New York: Plenum, 1997, pp. 285–292

[29] A. Albanese, J. Blomer, J. Edmonds, M. Luby and M. Sudan. "Priority Encoding Transmission," *IEEE Transactions on Information Theory*, vol.42, no.6, part1, pp.1737–1744, November 1996

[30] G. Davis and J. Danskin. "Joint Source and Channel Coding for Image Transmission Over Lossy Packet Networks," In *Proceedings of the Society of Photo-Optical Instrumentation Engineers Conference on Wavelet Applications to Digital Image Processing XIX*, vol.2847, 1996, pp.376–387

[31] A. Norkin, A. Gotchev, K. Egiazarian, and J. Astola. "A Low-Complexity Multiple Description Video Coder Based on 3D-Transforms," In *Proceedings of the 14th European Signal Processing Conference*, vol.2007, no.8, January 2007, pp.39-50

[32] T. Tillo and G. Olmo. "Data-Dependent Pre- and Post-Processing Multiple Description Coding of Images," *IEEE Transactions on Image Processing*, vol.16, no.5, pp.1269-1280, May 2007

[33] C. Su, J. Yoa and H. Chen. "H.264/AVC-Based Multiple Description Coding Scheme," In *Proceedings of the 14th IEEE International Conference on Image Processing*, vol.4, September 2007- October 2007, pp.265-268

[34] F. Verdicchio, A. Munteanu, A. Gavrilescu, J. Cornelis and P. Schelkens. "Embedded Multiple Description Coding of Video," *IEEE Transactions on Image Processing*, vol.15, no.10, pp.3114-3130, October 2006

[35] C-M. Chen, Y-C. Chen and C-W. Lin. "Seamless Roaming in Wireless Networks for Video Streaming," *IEEE International Symposium on Circuits and Systems*, vol.4, pp.3255-3258, May 2005

[36] J.R. Smith, R. Mohan and C. Li. "Scalable Multimedia Delivery for Pervasive Computing," Presented at the Association for Computing Machinery Multimedia Conference, Orlando, FL., 1999

[37] A. Fox and E.A. Brewer. "Reducing WWW Latency and Bandwidth Requirements by Real Timer Distillation," Presented at the International WWW Conference, Paris, France, 1996

[38] J. Xin C.-W. Lin and M.-T. Sun. "Digital Video Transcoding," In *Proceedings of IEEE*, vol.93, no.1, January 2005, pp.84-97

[39] K. Taehyun and M.H. Ammar. "Optimal Quality Adaptation for Scalable Encoded Video," *IEEE Journal on Selected Areas in Communications*, vol.23, pp.344-356, 2005

[40] S. Devillers, C. Timmerer, J. Heuer and H. Hellwagner. "Bitstream Syntax Description-Based Adaptation in Streaming and Constrained Environments," *IEEE Transactions on Multimedia*, vol.7, Issue 3, pp.463-470, June 2005

[41] D. Negru, T. Ahmed and A. Mehaoua. "NGL03-5: Ambient Middleware Framework for Efficient Service Provision to Users Anywhere Around the Globe," *IEEE Global Telecommunications Conference*, November 27 2006 – December 1 2006, pp.1-5

[42] M. Xu, J. Li, L. Chia, Y. Hu, B. Lee, D. Rajan and J.S. Jin. "Event on Demand with MPEG-21 Video Adaptation System," In *Proceedings of the 14th Annual ACM International Conference on Multimedia*, October 2006, pp.921-930

[43] T. Huang. "Multimedia Descriptions and Adaptation Delivery of Media Content Over Wireless," *International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, August 2008, pp.695-698

[44] T.Y. Huang and C. Zheng. "Prioritized MPEG-4 Audio Visual Objects Streaming Over the DiffServ," *Journal of Electronic Science and Technology of China*, pp.314-320, 2005

[45] J. Apostolopoulos. "Reliable Video Communication Over Lossy Packet Networks Using Multiple State Encoding and Path Diversity," In *Proceedings on Visual Communications and Image Processing*, 2001, pp.392–409

[46] H. Man, R.L. de Queiroz, and M.J.T. Smith. "Three-dimensional Subband Coding Techniques for Wireless Video Communications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol.12, no.6, pp.386-397, 2002

[47] J. Kim, R.M. Mersereau, and Y. Altunbasak. "Error-resilient Image and Video Transmission Over the Internet Using Unequal Error Protection," *IEEE Transactions on Image Processing*, vol.12, no.2, pp.121-131, 2003

[48] S. Somasundaram and K.P. Subbalakshmi. "3-D Multiple Description Video Coding for Packet Switched Networks," In *Proceedings of IEEE International Conference on Multimedia and Expo*, vol.1, July 2003, pp.589-592

[49] M. Yu, Z. Wenqin, G. Jiang, and Z. Yin. "An Approach to 3D Scalable Multiple Description Video Coding with Content Delivery Networks," In *Proceedings of IEEE International Workshop on VLSI Design and Video Technology*, May 2005, pp.191-194

[50] S. Saponara, L. Fanucci, and P. Terreni. "Low-power VLSI Architectures for 3D Discrete Cosine Transform (DCT)," In *Proceedings of the 46th IEEE International Midwest Symposium on Circuits and Systems*, vol.3, December 2003, pp.1567-1570

[51] A. Burg, R. Keller, J. Wassner, N. Felber, and W. Fitchtner. "A 3D-DCT Real Time Video Compression System for Low Complexity Single-Chip VLSI Implementation," In *Proceedings of the Mobile Multimedia Conference*, November 2000, pp.1B-5-1

[52] M. Bakr and A.E. Salama. "Implementation of 3D-DCT Based Video Encoder/Decoder System," In *Proceedings of the 45th IEEE Midwest Symposium on Circuits and Systems*, vol.2, August 2002, pp.13-16

[53] ITU-T, Video coding for low bitrate communication. ITU-T Recommendation, Draft on H.263v2, 1999

[54] D. Hunter, J. Rafter, J. Fawcett, E.van der Vlist, D. Ayers, J. Duckett, A. Watt and L. Mckinnon. *Beginning XML*. Indianapolis, IN.: Wrox/Wiley Publishers, 2007

[55] M. Kay. *XSLT Programmer's Reference*, 2nd Edition. Birmingham, UK.: Wrox Press, 2001

[56] E. Lenz. *XSLT- Pocket Reference*, 1st Edition. Sebastopol, CA.: O'Reilly Media, 2005

[57] I.S. Burnett, F. Pereira, R. Van de Walle and R. Koenen. *The MPEG-21 Book*, 1st Edition. Chichester, England; Hoboken, NJ.: Wiley, 2006

[58] C. Fogg, D.J. LeGall, J.L. Mitchell and W.B. Pennebaker. MPEG Video Compression Standard, 1st Edition. Norwell, MA.: Kluwer Academic Publishers, 1996

[59] N. Ahmed, T. Natarajan, and K. R. Rao. "Discrete Cosine Transform", *IEEE Transactions on Computers*, January 1974, pp.90-93

[60] K. Sayood. *Introduction to Data Compression*, 3rd Edition. San Francisco, CA.: Kaufmann Publishers, 2006

[61] R. Choupani, S. Wong and M. Tolun. "Scalable Video Coding: A Technical Report," Computer Engineering Laboratory, Delft University of Technology, Netherlands

[62] L.A. Larzon, M. Degermark and S. Pink. "Efficient Use of Wireless Bandwidth for Multimedia Application," *IEEE International Workshop on Mobile Multimedia Communications*, November 1999, pp.187-193

[63] C. Parsa and J.J. Garcia-Luna-Aceves. "Improving TCP Congestion Over Internets with Heterogeneous Transmission Media," In *Proceedings of the 7th International Conference on Network Protocols*, October 31 1999 – November 3 1999, pp.213-221

[64] V. Bhaskaran and K. Konstantinides. *Image and Video Coding standards – Algorithms and Architectures.* 2nd Edition. Norwell, MA.: Kluwer Academic Publishers, 1997

[65] D. Wang, N. Canagarajah, D. Redmill and D. Bull. "Multiple Description Video Coding Based on Zero Padding," In *Proceedings of the International symposium on Circuit and Systems*, vol.2, May 2004, pp.205-208

APPENDIX A

A.1 PICTURE LAYER

The first layer in the hierarchy is the picture layer. This layer starts with the picture header. When a picture is dropped, the corresponding picture header is not transmitted [4, 58]. The picture layer is composed of five components. The Picture Start Code (PSC) is a 20 bit word. After this comes the Temporal Reference (TR). This is a 5 bit number which can have 32 possible values. It is calculated by adding one to the value of the last transmitted TR and then also adding the number of the pictures that were not transmitted at 29.97 Hz. It is important to note that the calculation of this value is done using only the five least significant bits (LSB) [4, 58]. The next component in the picture layer is the Type Information (PTYPE). This component uses 6 bits to describe the various picture options. Each of the 6 bits has a corresponding option. The PTYPE bits and the associated options are as shown in the table below.

TABLE A.1: PTYPE Bits and Option.

BIT NUMBER	OPTION	"0" BIT	"1" BIT
Bit 1	Split screen indicator	Off	On
Bit 2	Document camera indicator	Off	On
Bit 3	Freeze picture release	Off	On
Bit 4	Source format	QCIF	CIF
Bit 5	Optional still image mode HI_RES defined in Annex D	On	Off
Bit 6	Spare		

The next component of the picture layer is the Extra Insertion Information (PEI). This component has a single bit. If the bit is set, it indicates that the next component which is the Spare Information (PSPARE) has a byte of information. The standard specifies that the decoders are to be designed in such a way that it disregards PSPARE if PEI is set to 1 [4, 58].

A.2 GROUP OF BLOCK LAYER (GOB)

The next layer after the picture layer is the GOB layer. Depending on the picture resolution, this layer can be composed of 3 or 12 macroblocks. Each GOB is made up of one twelfth of the CIF or one third of the QCIF of the picture area [4]. The GOB layer is composed of a GOB header followed by the data for the macroblock layer. First, we have the GOB Start Code (GOBSC). It is a 16 bit word. Next to that is the Group of Numbers (GN). It is made up of 4 bits that is a binary representation of the position of the group of blocks. The next component of this layer is the Quantizer Information represented by the acronym GQUANT. It is a 5 bit fixed length codeword which is used to indicate the exact quantizer being used by a GOB. The next two components of the GOB layer are the Spare Information (GEI) and the Spare Information (GSPARE) which serve the same function at this layer as the PEI and PSPARE for the picture layer [4, 58].

A.3 MACROBLOCK LAYER (MB)

The next layer is the macroblock layer (MB). It is composed of the MB header along with the data for the block layer. Each GOB is composed of 33 macroblocks. The first component of this layer is the Macroblock Address (MBA). This is a variable codeword that indicates the position of a macroblock in a GOB [1, 4, 58]. The first MBA is the address of the first transmitted GOB. The value of all subsequent MBA's are obtained by calculating the difference between the actual addresses of each macroblock and the last macroblock that was transmitted. The standard comes with a VLC table for macroblock addressing [4]. The next component of this layer is the Type Information (MTYPE). The MTYPE gives information about which of the 10 available encoding modes provided in the standard is currently used. It is also a variable length codeword.

The next component of the MB header is the Quantizer (MQUANT). MQUANT is only included in the MB header if it is indicated in the MTYPE. It is represented by a fixed length 5 bit codeword. Following MQUANT is the Motion Vector Data (MVD). Similar to MQUANT, MVD is only included in the MD header if it is indicated in MTYPE [4, 58]. The MVD is represented by a VLC of up to 11 bits in length which describes the differential displacement. The final component in the MB header is the Coded Block Pattern (CBP). It is a variable length code word indicating the blocks in the macroblock which have at least one transmitted transform coefficient and the variable length code can have a length as long as 9 bits [1]. The CBP is only included in the MB header if it is indicated by MTYPE.

TABLE A.2: VLC Table for MTYPE

Prediction	MQUANT	MVD	CBP	TCOEFF	VLC
Intra				x	0001
Intra	x			x	000 001
Inter			x	x	1
Inter	x		x	x	000 1
Inter + MC		x			0000 0000 1
Inter + MC		x	x	x	0000 0001
Inter + MC	x	x	x	x	0000 0000 01
Inter + MC + FIL		x			001
Inter + MC + FIL		x	x	x	01
Inter + MC + FIL	x	x	x	x	0000 01

NOTES

1 "x" means that the item is present in the macroblock

2 It is possible to apply the filter in a non-motion compensated macroblock by declaring it as MC + FIL but with a zero vector.

A.4 BLOCK LAYER

The final layer in the H.261 hierarchy is the Block layer. This is composed of the codewords for transform coefficients (TCOEFF) followed by the end of block marker (EOB). If MTYPE indicates INTRA, it implies that transform coefficient data is present in all six blocks of a macroblock. Otherwise in all other cases, MTYPE and CBP signal which blocks actually have coefficient data transmitted for them [4]. Each block in the macroblock is composed of 64 transform coefficients i.e. in each 8 by 8 block. The first level in the intra block is coded with a fixed length codeword of 8 bits [4, 58]. All other levels i.e. the various combinations of RUN and LEVEL are encoded with a 20 bit word.

20 bit word => 6 bits (ESCAPE), 6 bits (RUN), 8 bits (LEVEL)

The last bit in the code is an ‘s’ which denotes the sign of the level where ‘0’ indicates a positive and ‘1’ indicates a negative [4]. The Figure below shows how each GOB for either the CIF or QCIF picture format translates into macroblocks; and how each macroblock translates into individual blocks. Finally the Figure shows the structure of a block in a macroblock.

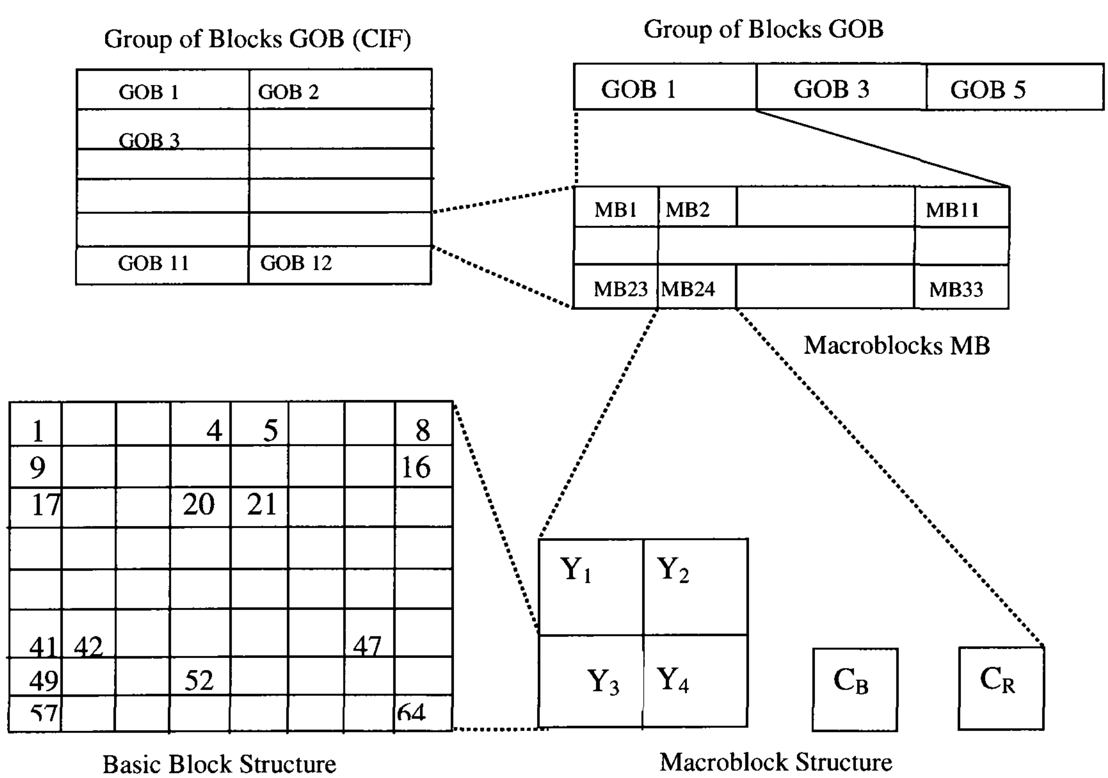

FIGURE A.1: Structure of the H.261 hierarchy [1].