POLITECNICO DI MILANO

Thesis of (Master of Science) in Computer Engineering

Department of Informatics

VIDEO CODEC IDENTIFICATION

Assistant: Paolo Bestagini

Supervisor: Assistant Professor Marco Tagliasacchi

Master Thesis:

Yiğit ÇağrıAKKAYA, 764625

Academic Year 2011-2012

Thanks to my mother Sebahat Akkaya and my father Ismail Coşkun Akkaya for providing me the best condintions in my whole life. I would like to thank my sister Müge Çağıl Akkaya for helping me always. I would like to present my gratitutes to Paolo Bestagini. I am grateful to Filip Petrovic, Nuno Lopes, Hasan Köse, Murat Ayhan, Sertaç Özdemir and Özlem Duyan for supporting me in any case.

Abstract

Video content is routinely captured and distributed in digital world. It is then common to encode video content multiple times for several purposes. Indeed, typically, a sequence is firstly encoded directly on the capturing device. Then it is re-encoded after any editing step. Moreover, an additional coding step may be applied when a sequence is uploaded on a sharing website. Identifying the used codec is a topic of interest for a forensic analyst. Indeed, it can be a useful hint to detect the device used to generate the content. In this thesis, we focus on the situation of video encoded twice, e.g., where the first coding step is applied by the generating device, and the second one by a sharing website. In this scenario we propose a codec identification method that aims to identify the codec used in the first coding step, by analyzing footprints left by this codec. The analysis is performed re-encoding the sequence with an additional coding step controlled by the analyst. By studying the correlation obtained between the input and the output of this additional coding step, we can infer with high accuracy which was the first codec used.

Contents

Chapter 1

Introduction

1.1 Problem Definition

Developing technology made digital contents to be used in every field of life. Smart-phones, tablets, video cameras etc. have started to become more and more inexpensive in recent years. Moreover, editing, copying and sharing the original audiovisual content has become less and less difficult though the help of improved network infrastructure and enhanced applications. Therefore, reliability in the digital world has decreased due to the ease with which people are able to alter original content. This has created distrust and, as a result of this distrust, digital videos and photographs are no longer assumed to be proof of evidence. In addition to this, difficulties in the detection of copyright infringements have arisen.

When the amount of academic research that addresses this issue is considered, one can see a significant increment in the amount of literature that is available. Furthermore, it is clear that a large part of existing research focuses on still images. There are some particular topics that academic literature intensely worked on still images, such as the possibility of validation, detection of alterations and recovery of the chain of processing steps. As a result, successful methods have been proposed as solutions for the afore mentioned problems. These solutions depend on the fact that most of the processing operations are irreversible. Therefore, these operations leave some traces in the output. These are referred to as *footprints* or *fingerprints*.

Despite the amount of research that has been performed in image forensics, academic literature pertaining to video forensics is currently not rich enough. Video contents are generally available in compressed formats. Strong compression rates on video content may cause a loss of existing fingerprints. Moreover, the compression algorithms that are preformed on video content is more sophisticated than those performed on still images. Thus, it is harder to perform forensic analysis on video content than it is on still images.

Although, academic literature lacks research done on video codec identification, some important researches has been performed in this field. In order to give a new impulse to academic literature, I focused on video codec identification.

In this thesis, I aimed to propose a video codec identification method when raw video content is coded twice. In real world circumstances, this can be thought of as uploading a video that is captured by a camera to a video sharing web site. The first coding may be applied by camera while capturing the video. Second one is applied while uploading the captured video to a web site such as YouTube. Syntax of the bitstream provides the information about the codec used in the second coding phase. Therefore, this thesis is concerned with the identification of the codec type used in the first coding phase.The proposed algorithm applies recompression to available video sequence by using different types of codec and compression parameters. Furthermore, it seeks similarities between the input and the output sequences of the additional coding step. In addition to these, a proper classification method has to be chosen in order to achieve a desirable true-positive rate.

In spite of the simplicity of the proposed method, experimental results show that identification is performed correctly on different video sequences. Test scenarios are executed step by step in order to determine noise tolerance of the proposed algorithm. Obtained results are quite promising in some cases.

1.2 Structure of the Thesis

The rest of this thesis is organized as follows. Chapter 2 provides an overview of video coding architecture. Moreover, it presents related works in academic literature. Chapter 3 defines the nature of the problem in detail and provides the algorithm that solves the problem. Chapter 4 contains the results obtained from experiments. Conclusion is done in Chapter 5. The appendices and bibliography can be found at the end of the thesis.

Chapter 2

State of the Art

2.1 Background

Video coding architectures inherits big part of its design from image coding architectures. The most well known image coding standard is JPEG and video coding standards adapt many of its coding principles [1]. As first step, JPEG starts with converting color images into a appropriate color space such as $YCbCr$(See Appendix A.2 for further details). After conversion, each color component is processed independently. The encoding process can be divided into three main step:

1. Non-overlapping $8x8$ blocks are formed. For sake of clarity, each block is represented as $X = [X(i,j)]$ where $0 \leq i \leq 7$ and $0 \leq j \leq 7$. Pixel values of each block are transformed by using a DCT into coefficients $Y(i,j)$.

2. Quantization levels $Y_q(i,j)$ are obtained from DCT coefficients $Y(i,j)$ by applying a uniform quantization with quantization step $\Delta(i,j)$. Quantization operation is shown in Equation 2.1.

$$Y_q(i,j) = sign(Y(i,j))Round(\frac{|Y(i,j)|}{\Delta(i,j)}) \tag{2.1}$$

At the decoder, the DCT coefficients are reconstructed by applying inverse inverse quantization operation defined in Equation 2.2.

$$Y_{iq}(i,j) = Y_q(i,j)\Delta(i,j) \tag{2.2}$$

3. Quantization levels $Y_q(i,j)$ are coded into a binary stream by using Huffman coding algorithm.

Since video coding architectures build on top of image coding tools, video coding architectures are more sophisticated than image coding architectures. Generally, video coding architectures (e.q. MPEG-x and H.26x families) not only use block-wise transform coding of the JPEG standards but also combines it with spatial and temporal prediction, in-loop filtering, interpolation etc.

Figure 2.1 presents simplified block diagram of a classical video coding architecture. Three main steps of JPEG standards can be seen easily. Moreover, a video coding architecture contains a prediction step. Prediction step, P, takes advantage of spatial or temporal correlation in the block. Predicted block is subtracted from original block and residual is encoded in similar way as JPEG standards apply.

Quantization, Q, is the main source of information loss. Therefore, this

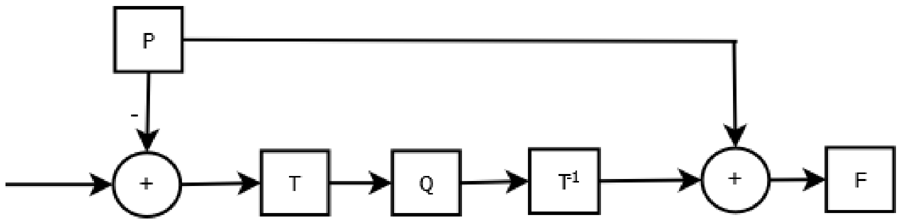


Figure 2.1: Simplified block diagram of a classical video codec. P, T, Q and F stand for prediction, orthonormal transform, quantization and in-loop filtering, respectively.

steps is non-invertible steps. Footprints left by lossy coding algorithm is caused by quantization step.

Due to block-wise operations, block artifact concept arises. The partitioning process depends on type of coding standard used. Therefore, block artifacts give an important clue about codec type.

In short, each codec may have a different coding architecture based on rate-distortion curve and computational constraints. Therefore, each codec leaves a different footprint that can be used by forensic analyst for several type of detection and validation purposes.

2.2 Literature Review

As stated in Chapter 1, the field of digital video forensics lacks research that compares to that available in the still image forensic area. However, many of the methods proposed for still images can be performed on video signals by considering each frame as a single image. Therefore, forensic analysis on still images are considered to be a good starting point of the literature review.

2.2.1 Image Forensics

Here, I give an overview of the existing research that has been performed on still image forensics. Forensic studies on still images can be summarized in three groups:

1. Camera Artifacts
2. Image Compression
3. Geometrical/Physical Inconsistencies

Camera Artifacts

Camera artifacts are formed during the acquisition process. Charged-Coupled Device(CCD)/Complementary Metal-Oxide-Semiconductor(CMOS) sensors create artifacts called Photo Response Non-Uniformity (PRNU) noise. PRNU basically describes the gain or ratio between optical power on a pixel versus the electrical signal output. Academic literature includes both digital camera identification [2] and image integrity methods that are [3] on PRNU.

Color Filter Array(CFA) is a mosaic of tiny color filters that are placed over the pixel sensors of a camera to form color information. In order to interpolate missing values, demosaicing algorithm is applied. Device pattern identification and tampering algorithms [4] use the benefits of pattern that are introduced in this phase.

Image Compression

Several methods are proposed based on image compression technique. Due to lossy nature of image compression, strategies leave certain footprints on the output signal. These footprints can be used to detect whether an image has been compressed [5]. If so, detection can be expanded on detection of encoder type [6], compression parameters [7] [8] and number of compression steps [25].

Geometrical/Physical Inconsistencies

The last group of image forensics focuses on inconsistencies in lighting, perspective, etc. of the image. This kind of technique presents a powerful approach for image integrity verification. The most significant research in this field is based on inconsistencies in scene illumination [9] and spotlight reflection in human eyes [10].

2.2.2 Video Forensics

2.2.2.1 Video Acquisition Analysis

The analysis of image acquisition device is one of the first problems in multimedia forensic studies. The basic goal is to provide information about the very first steps of the multimedia content. Several methods are proposed from different standpoints. Some of these approaches constructed a basis for video forensics. Studies that address video acquisition analysis can generally be divided in two groups:

1. Identification of Acquisition Device

Identification of Acquisition Device

In the first group, studies are focused on the identification of the device that captured the video content. Much of the research in this area involved the development of technology that can identify, whether a video is recorded by a camcorder or a modern mobile phone. Kurusowa et al. [11] were the first research group to propose a method that is based on camcorder footprints. They observed a fixed pattern noise of CCD chips and worked on this information.However, a small amount of academic literature research offered methods based on PRNU. In the beginning, it seemed that PRNU estimation of a camcorder is easier due to the large amount of frames. However, this is wrong in practice. First of all, the spatial resolution of a video content is not as high as that of a still image. The second problem is associated with video compression techniques. Frames are exposed to strong quantization and coding phases that causes more information loss than JPEG compression. The first body of work that relied on PRNU was presented by Chen et al. [12]. One challenging problem associated with video source identification concerns low resolution video contents. Van Houten et al. [13] [14] [15] proposed methods that investigates the nature of the problem in detail.

Detection of Reproduction

The second group is concerned with the copyright protection. We can sum up studies in this field under two topics. The first one is detection of re-acquisition. Methods that rely on active watermarking [16] [17] exist in academic literature for detecting a-d/d-a conversion and locating pirate position in cinema. Second topic can be considered as detection of copying. Bayrem et al. [18] considered the case of two similar video signatures which are not copies of each other.They proposed source device characteristics that were extracted from videos to construct copy detection technique.

2.2.2.2 Video Compression Forensics

This part of the academic literature survey played an important role in this thesis. Therefore, studies in this field are examined more precisely. I presented a peaky attitude about studies rely on double compression and codec identification. I examined research that had been performed in this field in three groups:

1. Video Coding Parameter Identification

2. Video Re-encoding

3. Network Footprints Identification

Video Coding Parameter Identification

A conventional video compression architecture has several parameters. This means increased number of degree of freedom. Therefore, methods proposed in literature are focused on estimating different coding parameters and syntax elements.

Some works have identified block size in a compressed video sequence [19]. By improving technology, it is possible to apply a de-blocking filter to reduce blocking artifacts. Therefore, traditional block detection algorithms fail.

As mentioned in 2.1, quantization is a non-invertible operation that leaves footprints. A histogram of transform coefficients has an obvious footprint. Instead of uniform distribution, histogram of transform coefficients has a comb-like distribution. The distribution can be expressed as:

$$p(Y_{iq};\Delta) = \sum_k \omega_k \delta(Y_{iq} - k\Delta) \tag{2.3}$$

Some methods focused on an estimation of the JPEG quality factor [20] [21], while another piece of research proposes the estimation of the whole quantization table[22]. Separate histograms are build from each DCT coefficient subband. Research indicates that it is possible to extract quantization steps by analyzing the power spectrum of every subband. Moreover, academic literature contains works that consider the case of MPEG-2 and

H.264/AVC coded video, respectively [23] [24]. In this case, histogram of DCT coefficients of prediction residuals are examined.

Video Re-encoding

Double compressed videos are passed through the quantization step twice. The histogram of DCT coefficients is affected by second quantization that has a different quantization step from the first one. Therefore, most of the methods are based on histogram of DCT coefficient.

One of the first pieces of research in this area was performed by Jan Lukǎs and Jessica Fridrich [25]. They tried to estimate primary quantization matrix in a double compressed image. Using this approach, they compressed one JPEG image twice with quantization matrices Q^1 and Q^2. They also stated that a DC coefficient is double compressed if, and only if, $Q^1 \neq Q^2$.

This research was primarily concerned with behavior of DCT coefficients. If DCT coefficients are quantized with quantization matrix Q^1, the value of the coefficient D_{ij} is expected to be a multiple of Q^1_{ij}, where i and j represents the index values in $8x8$ block in the image. After compression, the image is decompressed. Due to decompression, pixel values are rounden and truncated to 8-bit pixel values. As a result of this truncation, DCT coefficients are no longer multiples of Q^1 anymore. After second compression, a histogram is built for DCT coefficients and this histogram is examined further.

The researchers highlight two tasks, which are double compression detection and estimation of primary quantization matrix. They proposed a compatibility test ,which was mentioned in [26] and [27]. However, the method based on the compatibility test was computationally expensive and inefficient. Therefore, the authors proposed a new method that took advantage of *Neural Networks* (NN). The pseudo-code of the estimation is given below:

1. Extract the quantization matrix Q^2.

2. Get the histogram h_0 of absolute values of quantized DCT coefficient D^q_{ij}.

3. Remove the values $h_0(0)$ and $h_0(1)$, and normalize it.
4. Load the neural network corresponding to value of Q^2_{ij}.
5. Provide the normalized histogram h_0 as input to the neural network.
6. Output of neural network gives the estimated primary quantization step for D_{ij}.

An interesting piece of research that used double compression on video content, was based on block artifacts [28]. In the proposed method, they focused on $8x8$ block artifacts that were caused by MPEG compression. However, artifacts in P and B frames are more complicated due to motion compensation. The measurement of *Block Artifact Strength* (BAS) is influenced by by JPEG block artifact detection [21]. BAS of a frame is defined as a percentage of blocks that staisfy $|E + H - F - G| > |A + D - B - C|$ (See Figure 2.2). The researchers observed that I and P frames have higher BAS than B frames.

GOP was chosen as 12 and the number of B frames between I-P and

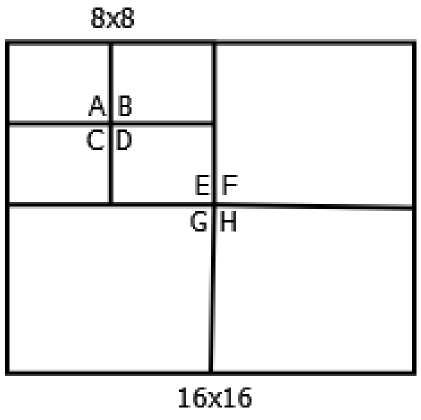


Figure 2.2: Block order for BAS calculation

P-P frame pairs was selected as 2 ($m = 12$ and $n = 3$. This created a GOP sequence of *IBBPBBPBBPBB*. Recompression was performed by erasing one to eleven frames from the MPEG compressed video. Frame removal and double compression are illustrated in Figure 2.3.

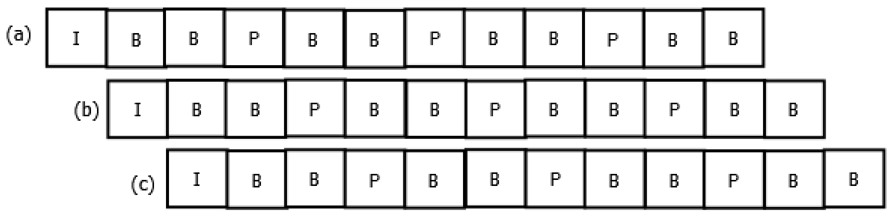


Figure 2.3: (a) Compression of original content with given GOP. (b) First frame of the result is removed and compressed with given GOP one more time. (b) First two frames of the result is removed and compressed with given GOP one more time.

Three groups were created from these eleven double MPEG compressed videos depending on their BAS. These three groups were used as features of the research. Two tampering methods were examined:

- Removal of video frames
- Use of different GOPs between the first and the second compression.

They obtained a periodic feature curves in the first case. It is observed that deleting x frames are similar to deleting y frames, when $m \equiv n$ (mod3). Moreover, results of the second case were promising. They found same inconsistencies in different test cases.

Wang and Farid [29] worked on detecting tampering in video by using double quantization.

Equation 2.4 represents quantization of DCT coefficient x in first compression phase. It should be noted that result of the operation is rounded.

$$x = [\frac{u}{q_1}] \tag{2.4}$$

Before compressing second time, the quantized DCT coefficients goes through de-quantization that is represented in Equation 2.5. It should be noted that result is multiple of q_1.

$$y = x * q_1 \tag{2.5}$$

In the last step, DCT coefficients are quantized one more time with a different quantization step.

$$z = [\frac{y}{q_2}] \tag{2.6}$$

In the lights of these three equations, Wang and Farid tried to model single compressed DCT coefficients by using Gaussian distribution.

$$P_{q_1}(y|x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y-xq_1)^2}{2\sigma^2}} \tag{2.7}$$

where xq_1 and σ are mean and standard deviation, respectively. Then, the observed marginal distribution of double compressed coefficients z is defined as in Equation 2.8.

$$P_{q_1}(z) = \sum_x P_{q_1}(x) P_{q_1}(z|x) \tag{2.8}$$

However, marginal probability of $P_{q_1}(c)$, used in Equation 2.8, is still unknown. Therefore, *Expectation Maximization* [30] algorithm to calculate $P_{q_1}(c)$. In detection mechanism, they used an *Euclidean Distance* based metric to find distance between expected coefficient and obtained coefficient. Obtained results were quite promising for the future researches.

Another double compression detection method is proposed by D. Liao et. al. [31] They worked on doubly compressed video format such as H.264/AVC and examined nonzero AC coefficients of DCT. The research is based on scalar quantization assumption of AVC(See Appendix). In the first step, video content is encoded with quantization step Q_1. Then, output is decoded with same quantization step. These operations are repeated one more time with a different quantization step, Q_2. However, compression is done without GOP. Therefore, compressed video has only I frames. In the classification phase, *Support Vector Machine* (SVM) with *Radial Basis Function* (RBF) is used as classifier. Features are probabilities of 20 AC coefficients of DCT, range from 10 to -10 excluding 0. The process of detection works as following:

1. Get total nonzero quantized coefficient by performing entropy decoding on I frames

2. Compute all the coefficients that stays in the range.

3. Calculate the probability of every obtained coefficient.

4. Give these 20 probability values to SVM.

There also exists an approach to identify codec by applying double compression [32]. This research built a basis for my thesis. In brief, the approach determines the codec by compressing the output video one more time. The proposed algorithm depends on the fact that quantization is an idempotent operation. In the other words, if we apply quantization to a value already quantized value, the output will be correlated with the input. As a matter of fact, it is possible to identify the adopted codec and its configuration by re-encoding the analyzed sequence a third time, with different codecs and parameter settings. Whenever the output sequence presents the highest correlation with the input video, it is possible to infer that the adopted coding set-up corresponds to that of the first compression

Network Footprints Identification

Video transmission over network is done on noisy channels in general. This causes loss of information on reconstructed video content. Therefore, some error neglecting algorithms proposed to overcome this problem. However, these methods leave a footprint on reconstructed content.

Some of the proposed methods works on transmission statistics. Transmission statistics are used to estimate channel distortion on the reconstructed content. An algorithm is presented relies on estimation of the packet loss impairment in the reconstructed content [33]. The problem in this approach is adaption of full-quality metrics. This means requirement of the original uncompressed video. Same authors also proposed another algorithm that focuses on effects of channel distortion on received video content[34]. They worked on three strategies. The first one makes a calculation of received video quality from network statistics. The second one analyzes the packet loss statistics to find the impacts in the spatial and temporal domain of the received video sequence. The third strategy evaluates effects of error propagation on the sequence.

Another approach proposed assumes there is no access to original video content. It evaluates effects of temporal and spatial error concealment [35]. This approach presents a good correlation with *Mean Opinion Score* (MOS).

A second class of strategies assumes that the transmitted video sequence has been decoded and that only the reconstructed pixels are available. There exists an approach which is build on top of [35]. The proposed solution only works on pixel values and identifies which slices of the video are lost[36]. Produced output is a value that presents a good quality with the *Mean Square Error* (MSE).

Chapter 3

Algorithm

3.1 Problem Formulation

In academic literature, video codec identification methods of research are really rare. Codecs leave a footprint on the resulting video as a result of their lossy coding nature. Nevertheless, characterizing the footprints of the codecs is not an easy task. First of all, several parameters are involved in the video coding process and this creates a large degree of freedoms. Another problem concerns the selection of the proper feature: the selected feature has to represent all codecs uniquely.

During the video coding process, important parameters can be listed as:

- Rate Control vs. Fixed Quantization Step: If rate control is selected, a bitrate has to be defined. Otherwise, the quantization step has to be defined.

- GOP: If GOP is defined as 0, the output video sequence consists of I frames. Another GOP value greater than 0 means that output video sequence contains I, P and B frames.

Changing these parameters affects the footprint left on the output sequence. Hence, there are several cases to be examined. At the end of the examination, the proposed classifier should present the desirable true-positive rate.

The feature selection problem is another important problem. The selected feature is supposed to represent every codec uniquely. Since sample video sequences are limited, feature space should not consist of many dimensions; otherwise, a problem that is referred to as *Curse of Dimensionality* arises. Therefore, the number of samples have to span feature space. On the other hand, small feature space may cause overlaps between two different codecs, which results in poor performance.

3.2 Idempotency Principle

In mathematics and computer science, idempotence is defined as a property of operations that does not change the initial result when applied multiple times. The concept of idempotency play an important role in closure operators in abstract algebra and functional programming. The term was first introduced by *Benjamin Peirce* as a concept in algebra.

A function is said to be idempotent depending on the concept:

1. A unary function is idempotent if $f(x) = f(f(x))$. Absolute value function can be given as an example of idempotent unary function.
2. A binary function is idempotent if it gives the same result when it is applied more than once. As an example, $\min(x, x) = \min(\min(x, x), x)$.
3. Given a binary operation, there may exist an idempotent element that does not change the result when it is used. For addition, the idempotent element is 0.

In video coding, idempotency arises when a video sequence is encoded several times using the same codec. Due to nature of lossy coding, the footprint of the codec is left on the resulting video. When the resulting video sequence is encoded one more time using the same codec, the same type of footprints are left one more time. Therefore, the difference between once and twice encoded video sequences is small. As a result, one can expect to obtain high PSNR values when these two video sequences are taken into account.

he idempotency principle is the reference point of this thesis. When a video sequence that is encoded with codec $c1$ is encoded with the same codec one more time, the PSNR values tend to cluster around one point. This would give us a clue about the codec used during the first encoding step.

3.3 Proposed Method

I proposed a solution to this problem based on idempotency principle. The simplified flow of double compression is represented in Figure 3.1. The

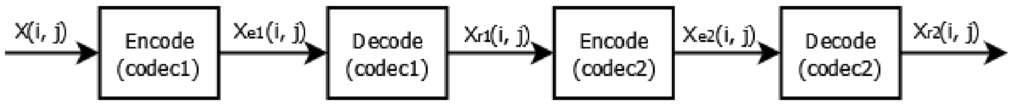


Figure 3.1: Simplified Block Diagram of Double Compression

original raw video $X(i,j)$ is encoded with codec *c1* where *c1*$\in$ {MPEG2, MPEG4, H.264/AVC, DIRAC} and output of the operation is represented as $X_{e1}(i,j)$. As stated in Equation 2.4, quantization operation leaves trace. Due to rounding operation, decoding operation cannot reconstruct the original video without information loss. Therefore, $X_{d1}(i,j)$ contains footprints of $c1$. The proposed method identifies $c1$ only having $X_{d1}(i,j)$ as input.

In order to identify $c1$, proposed algorithm encodes $X_{d1}(i,j)$ one more time with $c2$ where $c2 \in \{MPEG2, MPEG4, H.264/AVC, DIRAC\}$ as well. $c1$ may or may not be equal to $c2$. Output of the second encoding operation, $X_{e2}(i,j)$ is decoded and the resulting video sequence is represented as $X_{d2}(i,j)$.

Thus, two video sequences are obtained to analyze:

1. $X_{d1}(i,j)$ is the input video sequence.

2. $X_{d2}(i,j)$ is the output of the second compression step.

Now, the problem arisen is choosing the correct feature for further steps. PSNR (See Appendix C.3) is the ratio between maximum possible power of a signal and the power of the corrupting noise. Therefore, PSNR is used as quality metric of reconstruction of lossy codec. PSNR calculation is done as follows:

$$PSNR = 20\log(MAX_I) - 10\log(MSE) \tag{3.1}$$

where MAX_I and MSE are maximum possible pixel value in the original image and result of MSE (See Appendix C.1) calculation between original image and reconstructed image, respectively. Generally, possible maximum value in an 8-bit image is 255. If the original and the reconstructed images are totally same, result of MSE becomes 0. Therefore, logarithm operation tends to minus infinity. Consequently, PSNR value tends to infinity. Lower PSNR value means more difference between the original and the reconstructed image.

As mentioned in Section 3.2, encoding the original content with same codec has an idempotent effect. Hence, when $c1 = c2$, it is expected to obtain PSNR higher values whereas when $c1 \neq c2$, the expected PSNR values should be lower than values obtained in the first case. Consequently, PSNR is selected to create feature space. In order to create 2-dimensional feature space, two different PSNR calculations are needed. Thus, $X_{d1}(i,j)$ is subjected to two different second encoding phases with codecs $c2$ and $c3$. Results of the second encoding phases are represented as $X_{d2}(i,j)$ and $X_{d3}(i,j)$, respectively. Axises of 2-dimensional feature space are selected as results of PSNR calculations between $X_{d1}(i,j)$-$X_{d2}(i,j)$ and $X_{d1}(i,j)$-$X_{d3}(i,j)$, respectively.

The proposed algorithm consists of two phases:

1. Clustering

2. Classification

As stated above, it is tried to form clusters in 2-dimensional feature space. There are several parameters to be considered in order to obtain better clustering results. GOP is one of the most important ones. Since coding of I, P and B frames use different algorithms from each other, it is normal to have different PSNR values depending on type of frame. Variation of PSNR values creates a big problem in clustering phase. Besides, using either rate control or fixed quantization step is another point to be decided. Depending on decision, selected values effect the result of clustering. Another thing that effects clustering is selected type of codecs for second compression phase. Combination of these parameters create various possibilities to be tested.

In the classification phase, I used *Bayesian Classifier*. Let $\{\omega_1, ..., \omega_c\}$ be finite set of c categories and let $\{\alpha_1, ..., \alpha_c\}$ be finite set of possible outputs. The loss function $\lambda(\alpha_i|\omega_j)$ describes the loss incurred for taking output α_i when the category is ω_j. Let the feature vector x be a d-component vector valued and let $p(x|\omega_j)$ the state conditional probability density function for x. $P(\omega_j)$ describes the prior probabilty that the category is ω_j. Then, the posterior probability $P(\omega_j|x)$ can be computed from by Bayes formula:

$$P(\omega_j|x) = \frac{p(x|\omega_j)P(\omega_j)}{p(x)} \tag{3.2}$$

where the evidence is now

$$p(x) = \sum_{j=1}^{c} p(x|\omega_j)P(\omega_j) \tag{3.3}$$

Target of Bayes decision rule to minimize the overall risk, compute the conditional risk:

$$R(\alpha_i|x) = \sum_{j=1}^{c} \lambda(\alpha_i|\omega_j)P(\omega_j|x) \tag{3.4}$$

for $i = 1, ..., \alpha$ and then select the output α_i for which $R(\alpha_i|x)$ is minimum.

Chapter 4

Experimental Results

I tested the performance of the proposed algorithm on a dataset that consists of six video sequences at CIF (See Appendix A.3 for more information) resolution ($352x288$). Each of the original sequence was encoded with either MPEG-2, MPEG-4, AVC or DIRAC. As for the second coding pass, encoded sequences are re-encoded also with either MPEG-2, MPEG-4, AVC or DIRAC. This scenario is repeated with different encoding parameters.

Feature space has two dimensions. Second coding pass done twice with two different codecs. Thus, two PSNR values are obtained which represented as values on two axises of the feature space. For instance, original encoded video is re-encoded with AVC and MPEG-2 codecs. Two double encoded videos are obtained and PSNR calculation is done between the single encoded video and the double encoded videos. One axis of the feature space represents PSNR values associated AVC re-encoding whereas other axis represents PSNR values associated MPEG-2 re-encoding phase.

Performance of the proposed algorithm is presented in two sections. In the first section, clustering results with different parameters are presented. Each video sequence consists of 300 frames. Therefore, clustering is done based on 1800 encoded frames. In the second section, one video is selected as a test video and classification is done.

4.1 Clustering

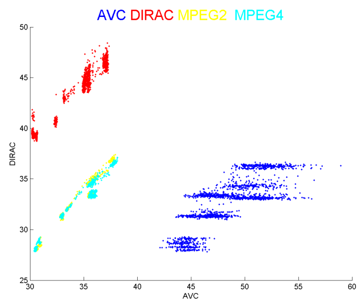
There are two main scenarios are considered. The first one uses different quantization parameters (QP) without introducing GOP. In this case, encoded video consists of only I frames. Since coding of P and B uses motion vectors and prediction, these frames are expected to effect result of clustering. Therefore, I excluded these frames in the first scenario. In the second scenario, I introduced also GOP in coding phases.

4.1.1 Fixed Quantization Parameter Without GOP

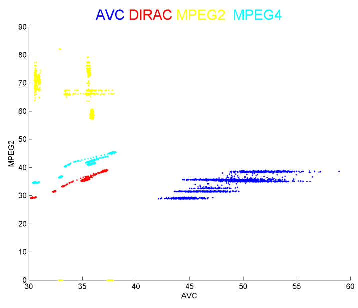
In the first scenario, fixed QP is considered without introducing GOP. As a start point, PSNR values between original raw video and single encoded video are examined. Three QPs are chosen which gives PSNR values between 30-40. These QPs are 7, 14 and 28 for MPEG-2 and MPEG4. QP of AVC is decided by Equation 4.1.

$$QP_{AVC} = 6\log_2(QP_{MPEG}) + 12 \tag{4.1}$$

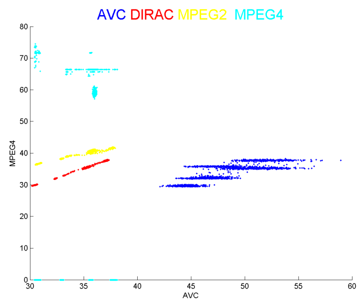
Due to quantization algorithm of MPEG standards, increasing QP means decreasing PSNR. In contrast, DIRAC quantization algorithm works reverse. Consequently, QP of DIRAC is selected as 5, 3 and 1, respectively. Selected QPs produce similar PSNR results at the end of the first coding pass. In the second encoding phase, encoded video is re-encoded with one of these QPs and a codec stated above. Clustering result can be seen at Figure 4.1 when QP is selected as 14 for both encoding passes.



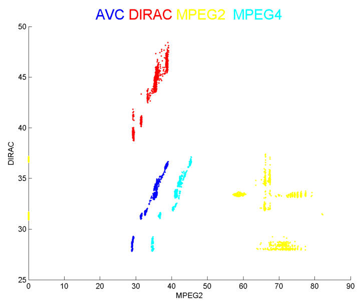
(a) AVC - DIRAC



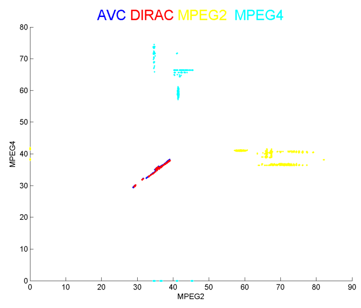
(b) AVC - MPEG-2



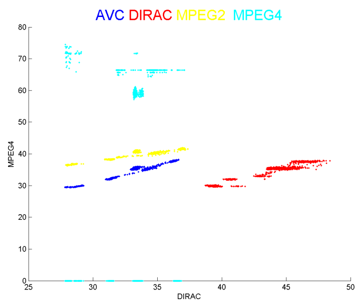
(c) AVC - MPEG-4



(d) MPEG-2 - DIRAC



(e) MPEG-2 - MPEG-4

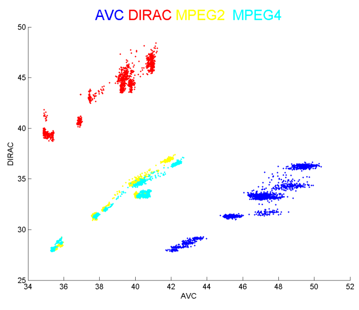


(f) DIRAC - MPEG-4

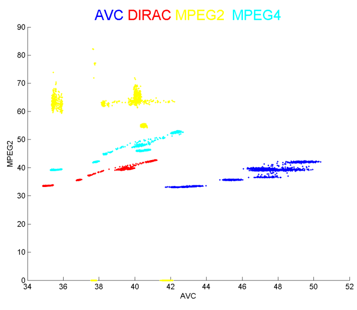
Figure 4.1: Clustering results when quantization parameter is selected 14 at both encoding passes

In Figure 4.1, axises of the plots represents the codec used in second encoding pass. Each color represents the type of codec used in the first encoding pass. For instance, AVC axis of each blue dot in Figure 4.1(a) represent PSNR value that is calculated by using single AVC encoded and double AVC encoded videos. Due to idempotency principle, it is expected to obtain high PSNR values when same codec used in second coding pass. All the plots in Figure 4.1 confirms the behavior of the idempotency principle.

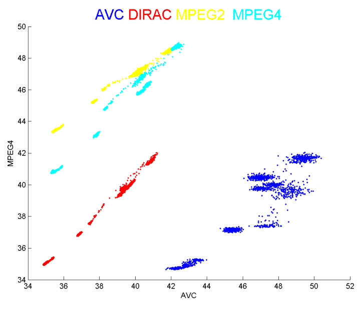
There are two more cases to be considered in this scenario. The question these cases is *"What would happen if I used a greater/less quantization parameter in the second encoding pass?"*. Since lower QP means better quality in MPEG standards, it is expected to have similar results in MPEG family when QP in the second pass is less. It is totally vice versa in the second question. Since DIRAC has different quantization parameter than MPEG standards, behavior of DIRAC is different. Due this fact, DIRAC is not used in classification phase. Figure 4.2 and Figure 4.3 shows clustering results in these two cases.



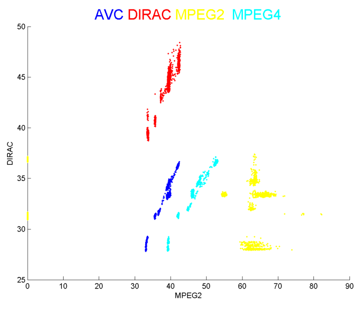
(a) AVC - DIRAC



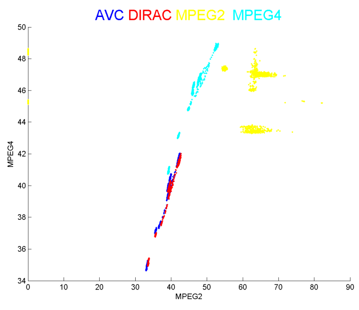
(b) AVC - MPEG-2



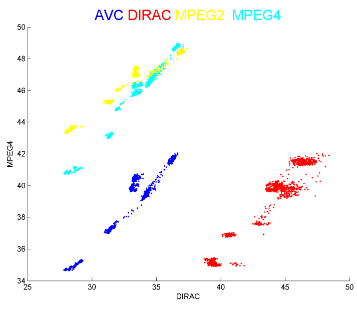
(c) AVC - MPEG-4



(d) MPEG-2 - DIRAC

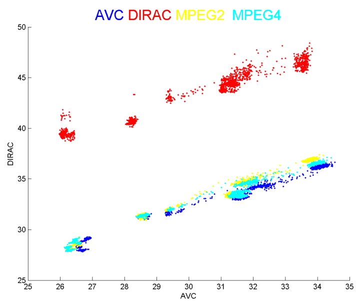


(e) MPEG-2 - MPEG-4

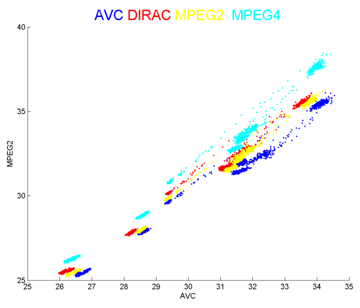


(f) DIRAC - MPEG-4

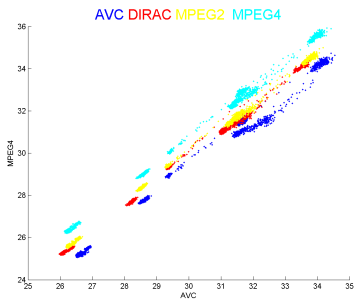
Figure 4.2: Clustering results when quantization parameter is selected 14 at the first encoding pass and 7 at the second encoding pass



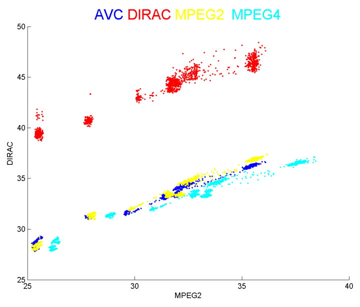
(a) AVC - DIRAC



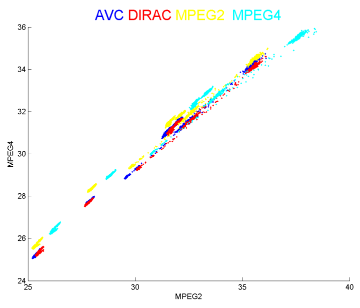
(b) AVC - MPEG-2



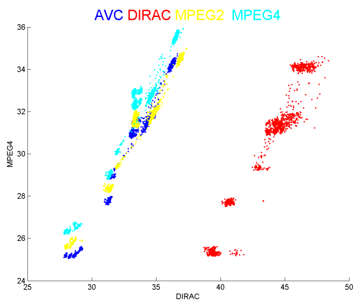
(c) AVC - MPEG-4



(d) MPEG-2 - DIRAC



(e) MPEG-2 - MPEG-4



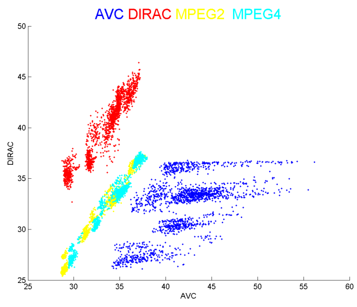
(f) DIRAC - MPEG-4

Figure 4.3: Clustering results when quantization parameter is selected 14 at the first encoding pass and 28 at the second encoding pass

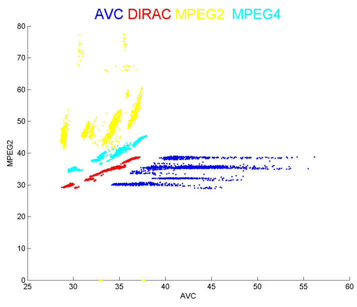
4.1.2 Fixed Quantization Parameter With GOP

When GOP is introduced, MPEG family uses P and B frames in encoding phase. Their size is lower than I frames and they contain less information. Especially, information they carry depends on motion in the video sequence. If there is no motion in the video sequence, information carried by P and B frames is almost nothing. This affects PSNR results. Hence, clusters are propagated into larger fields. Moreover, overlaps occur more often comparing to first case.

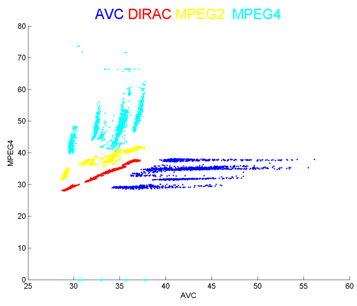
Same cases are tested by adding GOP parameter as 15. 15 is the value that *Mencoder* allows on videos for 25fps. The rest of the parameters used in the previous section remains same. The results are presented in Figure 4.4, Figure 4.5 and Figure 4.6.



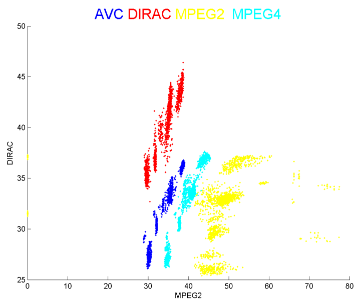
(a) AVC - DIRAC



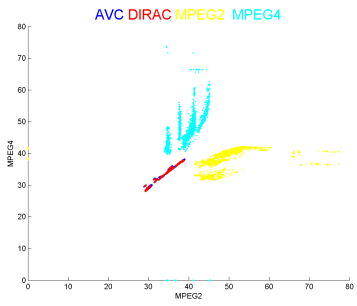
(b) AVC - MPEG-2



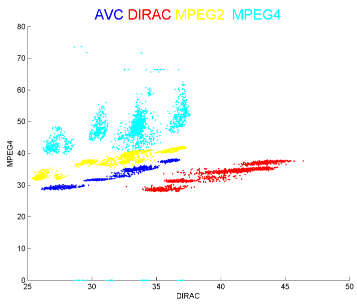
(c) AVC - MPEG-4



(d) MPEG-2 - DIRAC

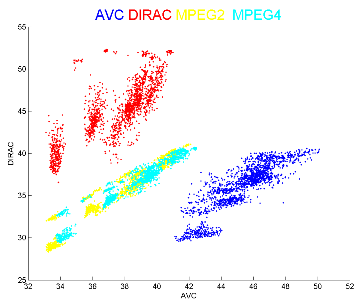


(e) MPEG-2 - MPEG-4

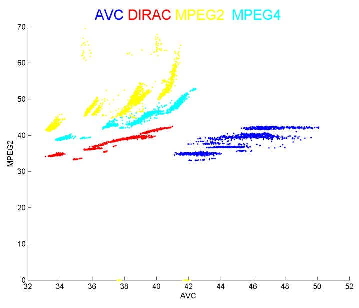


(f) DIRAC - MPEG-4

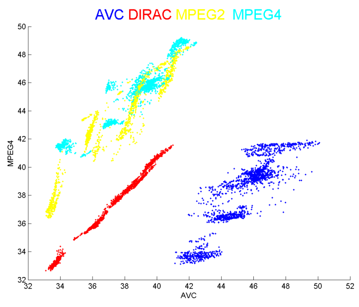
Figure 4.4: Clustering results when quantization parameter is selected 14 at both encoding passes



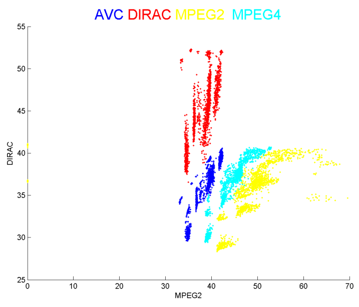
(a) AVC - DIRAC



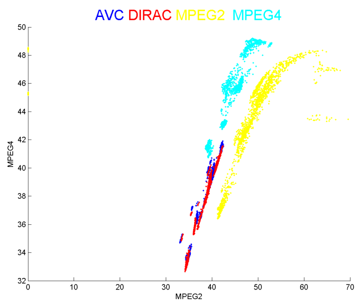
(b) AVC - MPEG-2



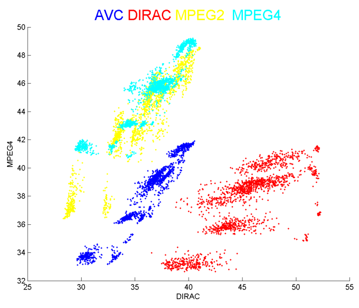
(c) AVC - MPEG-4



(d) MPEG-2 - DIRAC

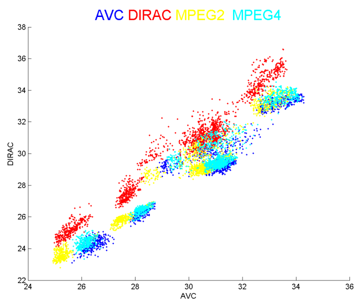


(e) MPEG-2 - MPEG-4

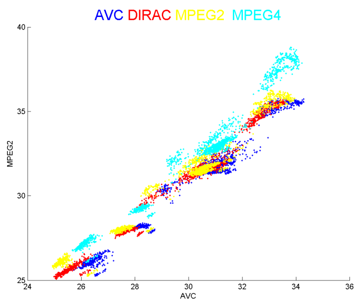


(f) DIRAC - MPEG-4

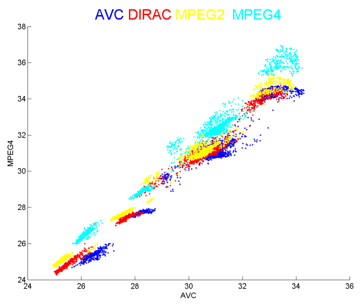
Figure 4.5: Clustering results when GOP is 15 and quantization parameter is selected 14 at the first encoding pass and 7 at the second encoding pass



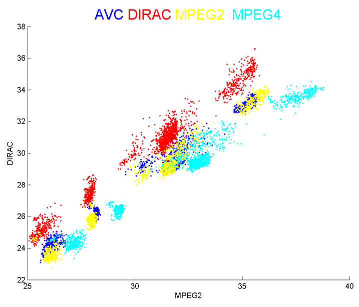
(a) AVC - DIRAC



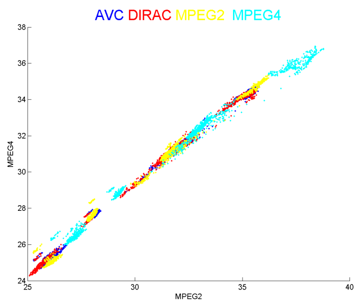
(b) AVC - MPEG-2



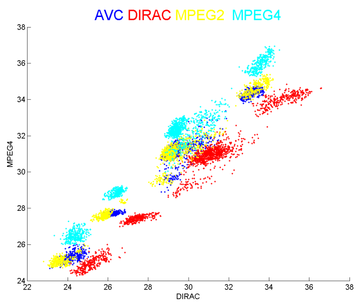
(c) AVC - MPEG-4



(d) MPEG-2 - DIRAC



(e) MPEG-2 - MPEG-4



(f) DIRAC - MPEG-4

Figure 4.6: Clustering results when GOP is 15 and quantization parameter is selected 14 at the first encoding pass and 28 at the second encoding pass

The obtained results were not promising. Moreover, trying different GOP values on each encoding pass causes inconsistent PSNR values. In the other words, PSNR calculation between I and P frame is totally meaningless, since I frame is encoded based on actual frame whereas P frame is encoded predictively. As a consequence, it is really difficult get clustered PSNR values.

4.2 Classification

The main concern of this thesis is detection of codec used in encoding phase of the video. Consequently, classification results are the main target of the study. As mentioned in the previous parts, it is really difficult to design that handles all degrees of freedom in the video coding. It is not only a smarter decision to focus on specific type of encoding process but also a realistic approach. As I presented in Section 4.1, focusing on encoding step with a GOP parameter defined as 0 is much more promising rather than encoding pass that defines a GOP.

Naive Bayesian Classifier is used as classification algorithm. As it can be seen from clustering results, clusters have an uniform distribution in a good clustered case such as QP is set as 14 in both coding passes. Naive Bayesian Classifier proposes good performance when data has a *Gaussian Distribution*. Therefore, Naive Bayesian Classifier is selected for the classification.

However, the designed classifier has to be tested on the sequences which are encoded by using different QP values. In this way, proposed algorithm can provide a benchmark. Test cases are gathered in two groups. The first group includes video sequences which are exposed double encoding. This group is named as *Noiseless Classification*. In the second group, output of the first encoding step pass through another coding step that causes noise in the input. This double encoded video sequence is encoded one more time to identify the codec used in the first encoding pass. This group is named as *Noisy Classification*. In this way, I can propose a benchmark about noise tolerance of the proposed algorithm.

4.2.1 Noiseless Classification

In this section, I present classification results of the classifier. Results are presented as confusion matrix. As a test case, classifier is selected as double encoded sequences which have QP value as 14 in both passes. Since it is assumed that parameters of the first coding pass are unknown, input video sequence is coded for the second time using the same parameters with the training phase. In order to propose a benchmark, various QP values are used in the first coding pass. Figure 4.7 shows the classification results when same parameters are used both for clustering and classification phases.

	AVC	DIRAC	MPEG-2	MPEG4
AVC	1800 (100%)	0	0	0
DIRAC	0	1153 (64%)	300 (16%)	347 (20%)
MPEG-2	0	0	1800(100%)	0
MPEG-4	0	600(34%)	300(16%)	900(50%)

(a) AVC - MPEG-2

	AVC	DIRAC	MPEG-2	MPEG4
AVC	1798(99.9%)	0	0	2(0.1%)
DIRAC	0	1179(66%)	321(18%)	300(16%)
MPEG-2	0	300(16%)	1500(84%)	0
MPEG-4	0	0	0	1800(100%)

(b) AVC - MPEG-4

	AVC	DIRAC	MPEG-2	MPEG4
AVC	0	1799(99.9%)	1(0.1%)	0
DIRAC	1800(100%)	0	0	0
MPEG-2	0	0	1800(100%)	0
MPEG-4	0	0	0	1800(100%)

(c) MPEG-2 - MPEG-4

Figure 4.7: Results of the classifier created by using 14 as QP value in both coding passes. Test sequences also have same configuration. Descriptive codecs represents the codecs used in the second coding pass.

As it can be seen from Figure 4.7, the best classification result is obtained when AVC and MPEG-4 is used for the second coding pass. As a result of this, these codecs are used for further tests.

The base of the thesis assumes that we don't know anything about first coding step. Type of codec is identified by applying second coding steps. In order to present more reliable results, first step is repeated with different QP values. Since classifier is trained by using 14 as QP in the second coding step, input is exposed to second coding step with QP of 14. In Figure 4.8 shows the classification results when QP of the first coding step is set to 7 and 28, respectively.

	AVC	DIRAC	MPEG-2	MPEG4
AVC	0	1734(96%)	66(4%)	0
DIRAC	0	1719(96%)	81(4%)	0
MPEG-2	0	1486(82%)	314(18%)	0
MPEG-4	0	1475(82%)	325(18%)	0

(a) QP of the first coding step is set to 7.

	AVC	DIRAC	MPEG-2	MPEG4
AVC	1631(90%)	169(10%)	0	0
DIRAC	0	1450(80%)	350(20%)	0
MPEG-2	0	0	1468(81%)	332(19%)
MPEG-4	0	0	1465(81%)	333(19%)

(b) QP of the first coding step is set to 28.

Figure 4.8: Results of the classifier when QP value used in the first coding pass is different than the second pass. In the second coding pass, QP is set to 14.

When QP of the first coding step is lower than the second one, the classification results are totally meaningless as expected. Due to higher QP values in the second coding step, footprints left from the first coding step are masked. Solution of this problem is using a lower QP value in the second coding pass. Figure 4.9 shows the results when 7 is selected as QP value in both coding steps. On the other hand, when the situation is vice versa, meaningful results are obtained.

	AVC	DIRAC	MPEG-2	MPEG4
AVC	1800(100%)	0	0	0
DIRAC	0	1204(67%)	296(16%)	300(17%)
MPEG-2	0	300(16%)	1500(84%)	0
MPEG-4	0	0	0	1800(100%)

Figure 4.9: Results of the classifier created by using 7 as QP value in both coding passes.

4.2.2 Noisy Classification

In this scenario, it is assumed that input video is exposed to noise. In order to create noise on input video, an indermediate coding step is applied. For instance, the video sequence that is encoded with AVC codec, is encoded with another codec one more time. Therefore, the footprints left from AVC codec are affected by the footprints left from intermediate codec. Classification is performed after this intermediate coding step. Consequently, a fall in the detection performance is expected. As a start point, the first test is done by using same QP in the all coding steps. Figure 4.10 shows classification results when QP is set as 14 in all coding steps. Axises of the feature space is selected as AVC and MPEG-4.

	AVC	DIRAC	MPEG-2	MPEG4
AVC	1726(96%)	0	0	74(4%)
DIRAC	1800(100%)	0	0	0
MPEG-2	1800(100%)	0	0	0
MPEG-4	1800(100%)	0	0	0

(a) Input video sequence is encoded with AVC in order to create noise.

	AVC	DIRAC	MPEG-2	MPEG4
AVC	1800(100%)	0	0	0
DIRAC	0	1482(83%)	318(17%)	0
MPEG-2	0	300(16%)	1500(84%)	0
MPEG-4	0	0	0	1800(100%)

(b) Input video sequence is encoded with DIRAC in order to create noise.

	AVC	DIRAC	MPEG-2	MPEG4
AVC	0	300(16%)	1500(84%)	0
DIRAC	300(16%)	1500(84%)	0	0
MPEG-2	0	300(16%)	1500(84%)	0
MPEG-4	0	299(16%)	1300(73%)	201(11%)

(c) Input video sequence is encoded with MPEG-2 in order to create noise.

	AVC	DIRAC	MPEG-2	MPEG4
AVC	0	0	0	1800(100%)
DIRAC	0	0	0	1800(100%)
MPEG-2	0	0	0	1800(100%)
MPEG-4	0	0	0	1800(100%)

(d) Input video sequence is encoded with MPEG-4 in order to create noise.

Figure 4.10: Classification results when noisy video sequences are used as input. QP value is set to 14 in all coding steps.

As it can be seen from Figure 4.10, intermediate coding step erase most of the footprints left from original codec. Therefore, classification performance decreases dramatically. Since coding algorithm of DIRAC is different than MPEG family, results are not affected.

The main reason of this dramatic decrease is using same QP value while creating the noise. In order to measure noise tolerance of the classifier, intermediate coding step is repeated with lower QP values. Figure 4.11 and Figure 4.12 show classification results when QP of intermediate coding step is set to 7 and 1, respectively.

	AVC	DIRAC	MPEG-2	MPEG4
AVC	1442(83%)	358(17%)	0	0
DIRAC	0	1480(83%)	320(17%)	0
MPEG-2	0	680(38%)	1120(62%)	0
MPEG-4	0	300(16%)	1500(84%)	0

(a) Input video sequence is encoded with AVC in order to create noise.

	AVC	DIRAC	MPEG-2	MPEG4
AVC	1800(100%)	0	0	0
DIRAC	0	1482(82%)	318(18%)	0
MPEG-2	0	300(16%)	1500(84%)	0
MPEG-4	0	0	0	1800(100%)

(b) Input video sequence is encoded with DIRAC in order to create noise.

	AVC	DIRAC	MPEG-2	MPEG4
AVC	0	1468(81%)	332(19%)	0
DIRAC	0	1469(81%)	331(19%)	0
MPEG-2	0	300(16%)	1500(84%)	0
MPEG-4	0	263(15%)	414(23%)	1123(62%)

(c) Input video sequence is encoded with MPEG-2 in order to create noise.

	AVC	DIRAC	MPEG-2	MPEG4
AVC	0	1207(67%)	593(33%)	0
DIRAC	0	1277(71%)	523(29%)	0
MPEG-2	0	0	1625(90%)	175(10%)
MPEG-4	0	0	600(33%)	1200(67%)

(d) Input video sequence is encoded with MPEG-4 in order to create noise.

Figure 4.11: Classification results when noisy video sequences are used as input. QP value is set to 14 in the first and third coding steps. In intermediate (noise) step, QP is selected as 7.

	AVC	DIRAC	MPEG-2	MPEG4
AVC	1800(100%)	0	0	0
DIRAC	0	1481(82%)	319(18%)	0
MPEG-2	0	300(16%)	1500(84%)	0
MPEG-4	0	0	0	1800(100%)

(a) Input video sequence is encoded with AVC in order to create noise.

	AVC	DIRAC	MPEG-2	MPEG4
AVC	1800(100%)	0	0	0
DIRAC	0	1482(82%)	318(18%)	0
MPEG-2	0	300(16%)	1500(84%)	0
MPEG-4	0	0	0	1800(100%)

(b) Input video sequence is encoded with DIRAC in order to create noise.

	AVC	DIRAC	MPEG-2	MPEG4
AVC	1796(99.9%)	4(0.1%)	0	0
DIRAC	0	1476(82%)	324(18%)	0
MPEG-2	0	300(16%)	1500(84%)	0
MPEG-4	0	0	0	1800(100%)

(c) Input video sequence is encoded with MPEG-2 in order to create noise.

	AVC	DIRAC	MPEG-2	MPEG4
AVC	1800(100%)	0	0	0
DIRAC	0	1482(82%)	318(18%)	0
MPEG-2	0	250(14%)	1550(86%)	0
MPEG-4	0	0	0	1800(100%)

(d) Input video sequence is encoded with MPEG-4 in order to create noise.

Figure 4.12: Classification results when noisy video sequences are used as input. QP value is set to 14 in the first and third coding steps. In intermediate (noise) step, QP is selected as 1.

While decreasing QP value for intermediate step, performance of the classifier is increasing as expected. As mentioned before, DIRAC codec does not cause any noise because of its coding algorithm.

Chapter 5

Conclusion

The presented thesis proposes a novel algorithm for identification of video coding standards. This method has firstly been analyzed considering sequences encoded once but available only as decoded sequences (i.e., the bitstream was unknown). Then it has been tested on the more interesting scenario of sequences double encoded, where the first codec is to be detected, and the second one acts as a masking codec (i.e., noise). The proposed method relies on the fact that an analyst can re-encode with several codecs a sequence previously encoded with a given unknown codec. In doing so, he can compute the input-output correlation (as PSNR between the sequences) that is characteristic of the first used codec.

The detector is based on a Bayesian classifier that works on these PSNR values. As a first step, it is trained on the input-output PSNR values obtained from a training dataset. Then it is used to detect the codec of test sequences. Depending on the codecs used in the analysis step by the analyst, results may change. For this reason, we performed an analysis to let the analyst chose beforehand the best analysis codec set.

Experimental results showed that when QP parameter of the analyst coding pass is greater than the first pass, identification rates are quite promising. However, when the situation is vice versa, the analyst codec masks the footprints of the first coding step. As it regards the masking codec, experimental results showed that when its QP value is equal or greater than that of the first coding step, classification results are negatively affected. Otherwise, obtained results were quite accurate.

The proposed algorithm focuses on video sequences that include only I frames. Since P and B frames are coded predictively, clustering results are affected negatively by these kind of frames. However, the knowledge of which frames are I frames is not a strict hypothesis, since it is possible to detect them using state of the art techniques. GOP parameter might be introduced as a next step.

As a last remark, it is worth to notice that even if classification is done using MPEG-family codec (DCT-based) as analysis codecs, also sequences encoded with DIRAC (wavelet-based) are correctly identified.

Bibliography

[1] Wallace, G.: The jpeg still picture compression standard. *IEEE Trans. Consum. Electron.*, 38(1) (1992), xviii-xxxiv.

[2] Lukǎs, J.; Fridrich, J.; Goljan, M.: Digital camera identification from sensor pattern noise. *IEEE Trans. Info. Forensics and Secur.*, 1(2) (2006), 205-214.

[3] Chen, M.; Fridrich, J. J.; Goljan, M.; Lukǎs, J.: Determining image origin and integrity using sensor noise, *IEEE Trans. on Info. Forensics Secur.*, 3(1) (2008), 74-90.

[4] Popescu, A.; Farid, H.: Exposing digital forgeries in color filter array interpolated images, *IEEE Trans. Signal Process.*, 53(10) (2005) 3948-3959.

[5] Fu, D.; Shi, Y. Q.; Su, W.: A generalized benfords law for jpeg coefficients and its applications in image forensics, in *Proc. of SPIE, Security, Steganography and Watermarking of Multimedia Contents IX*, vol. 6505, January 28-February 1, 2009, 39-48.

[6] Lin, W. S.; Tjoa, S. K.; Zhao, H. V.; Liu, K. J. R.: Digital image source coder forensics via intrinsic fingerprints, *IEEE Trans. Info. Forensics Secur.*, 4(3) (2009), 460-475.

[7] Fan, Z.; de Queiroz, R. L.: Maximum likelihood estimation of JPEG quantization table in the identification of bitmap compression history, in *ICIP*, 2000.

[8] Fan, Z.; de Queiroz, R. L.: Identification of bitmap compression history: JPEG detection and quantizer estimation, *IEEE Trans. Image Process.*,12(2) (2003), 230-235.

[9] Johnson, M. K.; Farid, H.: Exposing digital forgeries in complex lighting environments, *IEEE Trans. Info. Forensics Secur.*, 2(3-1) (2007), 450-461.

[10] Johnson, M. K.; Farid, H.: Exposing digital forgeries through specular highlights on the eye, in Information Hiding, *Lecture Notes in Computer Science, T. Furon, F. Cayre, G. J. DoÃ«rr, and P. Bas, eds., vol. 4567*, Springer, 2007, 311-325.

[11] Kurosawa, K.; Kuroki, K.; Saitoh, N.: Ccd fingerprint method identification of a video camera from video taped images, in *Proc. 1999 Int. Conf. on Image Processing, 1999. ICIP 99., vol. 3*, 1999, 537-540.

[12] Chen, M.; Fridrich, J.; Goljan, M.; LukÃ¡s, J.: Source digital camcorder identification using sensor photo response non-uniformity, in *Proc. SPIE*, 2007.

[13] Van Houten, W.; Geradts, Z. J. M. H.; Franke, K.; Veenman, C. J.: Verification of video source camera competition (camcom 2010), in *ICPR Contests, Lecture Notes in Computer Science, D.Ünay, Z. Çataltepe, and S. Aksoy, eds., vol. 6388. Springer*, 2010, 22-28.

[14] Van Houten, W.; Geradts, Z. J. M. H.: Using sensor noise to identify low resolution compressed videos from youtube, in *IWCF, Lecture Notes in Computer Science, Z. J. M. H. Geradts, K. Franke, and C. J. Veenman, eds., vol. 5718. Springer*, 2009, 104-115.

[15] van Houten, W.; Geradts, Z. J. M. H.: Source video camera identification for multiply compressed videos originating from youtube. *Digital Investigation*, 6(1-2) (2009), 48-60.

[16] Lee, M.-J.; Kim, K.-S.; Lee, H.-Y.; Oh, T.-W.; Suh, Y.-H.; Lee, H.-K.: Robust watermark detection against d-a/a-d conversion for digital cinema using local auto-correlation function, in *ICIP, IEEE*, 2008, 425-428.

[17] Lee, M.-J.; Kim, K.-S.; Lee, H.-K.: Digital cinema watermarking for estimating the position of the pirate. *IEEE Trans. Multimed.*, 12(7) (2010), 605-621.

[18] Bayram, S.; Sencar, H. T.; Memon, N. D.: Video copy detection based on source device characteristics: a complementary approach to content-based methods, in *Proc. 1st ACM SIGMM Int. Conf. on Multimedia Information Retrieva*l, MIR 2008, Vancouver, British Columbia, Canada, October 30-31, 2008, M. S. Lew, A. D. Bimbo, and E. M. Bakker, eds. ACM, 2008, 435-442.

[19] Li, H.; Forchhammer, S.: MPEG2 video parameter and no reference PSNR estimation, in *Picture Coding Symp.*, 2009. PCS 2009, May 2009, 1-4.

[20] Fan, Z.; de Queiroz, R. L.: Maximum likelihood estimation of JPEG quantization table in the identification of bitmap compression history, in *ICIP*, 2000.

[21] Fan, Z.; de Queiroz, R. L.: Identification of bitmap compression history: JPEG detection and quantizer estimation, *IEEE Trans. Image Process.*,12(2) (2003), 230-235.

[22] Ye, S.; Sun, Q.; Chang, E.-C.: Detecting digital image forgeries by measuring inconsistencies of blocking artifact, in *ICME, IEEE*, 2007, 12-15.

[23] Chen, Y.; Challapali, K. S.; Balakrishnan, M.: Extracting coding parameters from pre-coded MPEG-2 video, in *ICIP (2)*, 1998, 360-364.

[24] Tagliasacchi, M.; Tubaro, S.: Blind estimation of the QP parameter in H.264/AVC decoded video, in 2010 *11th Int. Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, April 2010, 1-4.

[25] Lukăs, J.; Fridrich, J.: Estimation of primary quantization matrix in double compressed jpeg images, in Proc. of *Digital Forensic Research Workshop*, 2003.

[26] Fridrich J., Goljan M. and Hogea D., Attacking the OutGuess, *Proc. Multimedia and Security Workshop at ACM Multimedia*, December 6, 2002.

[27] Fridrich J., Goljan M., Hogea D., Staganalysis of JPEG Images: Breaking the F5 Algorithm, *Proc.* 5th *International Workshop on Information Hiding*, October 2002.

[28] LukÃ¡s, J.; Fridrich, J.: Estimation of primary quantization matrix in double compressed jpeg images, in *Proc. of DFRWS*, 2003.

[29] Wang, W.; Farid, H.: Exposing digital forgeries in video by detecting double MPEG compression, in *MM&Sec, S. Voloshynovskiy, J. Dittmann, and J. J. Fridrich, eds.,ACM*, 2006, 37-47.

[30] A. Dempster, N. Laird and D. Rubin, *Maximum Likelihood From Incomplete Data via the EM Algorithm*, Journal of the Royal Statistical Society, 1977.

[31] D. Liao, R. Yang, H. Liu and J. Huang, Double H.264/AVC Compression Detection Using Quantized Nonzero AC Coefficients, *Media Watermarking Security and Forensics III*, 2011.

[32] Bestagini, P.; Allam, A.; Milani, S.; Tagliasacchi, M.; Tubaro, S.: Video codec identification, in *Proc. 37th Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP 2012)*, March 25-30, 2012, 2257-2260.

[33] Reibman, A. R.; Poole, D.: Characterizing packet-loss impairments in compressed video, in *ICIP (5), IEEE*, 2007, 77-80.

[34] Reibman, A. R.; Vaishampayan, V. A.; Sermadevi, Y.: Quality monitoring of video over a packet network. *IEEE Trans. Multimed.*, 6(2) (2004), 327-334.

[35] Naccari, M.; Tagliasacchi, M.; Tubaro, S.: No-reference video quality monitoring for H.264/AVC coded video. *IEEE Trans. Multimed.*,11(5) (2009), 932-946.

[36] Valenzise, G.; Magni, S.; Tagliasacchi, M.; Tubaro, S.: Estimating channel-induced distortion in H.264/AVC video without bitstream information, in *2010 Second Int. Workshop on Quality of Multimedia Experience (QoMEX)*, June 2010, 100-105.

[37] John Watkinson, *MPEG Handbook, Second Edition*, 2004.

[38] Ian E. G. Richardson, *H.264 and MPEG-4 Video Compression*, 2003.

[39] Wiegand, T.; Sullivan, Garry J.; Bjøntegaard, G. and Luthra, A., Overview of the H.264/AVC Video Coding Standard,*IEEE Transactions on Circuits and Systems for Video Technology, VOL. 13, NO. 7*, JULY 2003.

[40] M. Tun, K.K. Loo, J. Cosmas, Error-resilient performance of Dirac video codec over packet-erasure channel, *IEEE Trans. Broadcast.*, 53 (3) (September 2007), pp. 649-659.

[41] M. Tun, , K.K. Loo , J. Cosmas, Rate control algorithm based on quality factor optimization for Dirac video codec, *Signal Processing: Image Communication Volume 23, Issue 9*, October 2008, Pages 649-664.

[42] Bjarne Stroustrup *C++ Programming Language*, 1997.

Appendix A

Video Formation

Digital video may be considered as consisting of a series of still image frames. Each frame has spatio-temporal element (pixel) which is represented as a number or set of numbers. Value or values in pixels describe the brightness and color. Frames consist of two dimensional projection of three dimensional world. Therefore, each frame is represented as two dimensional array and every element of this array represents one pixel in the frame.

A.1 Sampling

There are two main sampling process in formation of the video. First one is spatial sampling which describes number of pixels in a frame. Choosing a coarse spatial sampling grid causes a low resolution image. In the lights of these information, spatial information is given by the relationship between neighboring pixels.

A video is captured by taking snapshots of the signal at periodic time intervals. Playing these captured signals back to back constructs motion. Therefore, higher sampling rate in temporal domain provides a smoother motion in the video. Temporal sampling rate in the video is called as *frame rate* of the video. Sampling between 25 or 30 frames per second is standart for television.

A.2 Color Formation

Colors in the images are represented by a number or a set of numbers. In a monochrome image, there is only one number per pixel which represents brightness or luminance. In order to display color images accurately, one of the color space models has to be applied.

The most well known color space model is *RGB*. Each pixel is represented by a vector keeps three numbers. Each value in this vector represents proportions of red, green and blue color components of the pixel. Therefore, every image can be considered as consisting of three different planes (red, green and blue planes). Color information in the image obtained by adding up these three color components. This is why *RGB Color Space* is called *additive* color space. Since human visual system works in a similar way to *RGB Color Space*, it is mostly used in computer graphics and digital video formation. Geometric interpretation of the *RGB* color space can be seen in Figure A.1.

RGBA Color Space is derived from *RGB*. Generally, *RGBA* is inter-

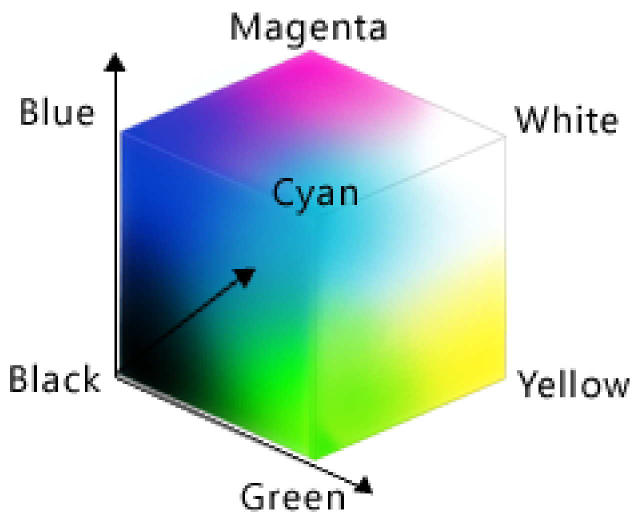


Figure A.1: Geometric Interpretation of RGB Color Space

preted as usage of *RGB* with extra information. The last component of

RGBA stands for *alpha* channel of the color space. When this component is 0, color becomes totally transparent.

YCbCr Model is another important color space used in video formation. As in *RGB*, *YCbCr* color space keeps a vector for each pixel consisting of three values. *Y* component stands for luminance. *Cb* and *Cr* represent blue-difference and red-difference of chrominance component. Idea behind the *YCBCR* model is representing a color image more efficiently by seperating luminance information from color information. *YUV* color space is a variation of *YCbCr*. *YCbCr* model is mostly used in digital photography. Geometric interpretation of *YCbCr* color space can be seen in Figure A.2.

Color spaces explained above are the most important ones used in digital video formation. *RGB* color space is used in this thesis. There are more color spaces which are designed for different purposes. *CMYK* is a subtractive color model that is generally used in color priting and also describes the

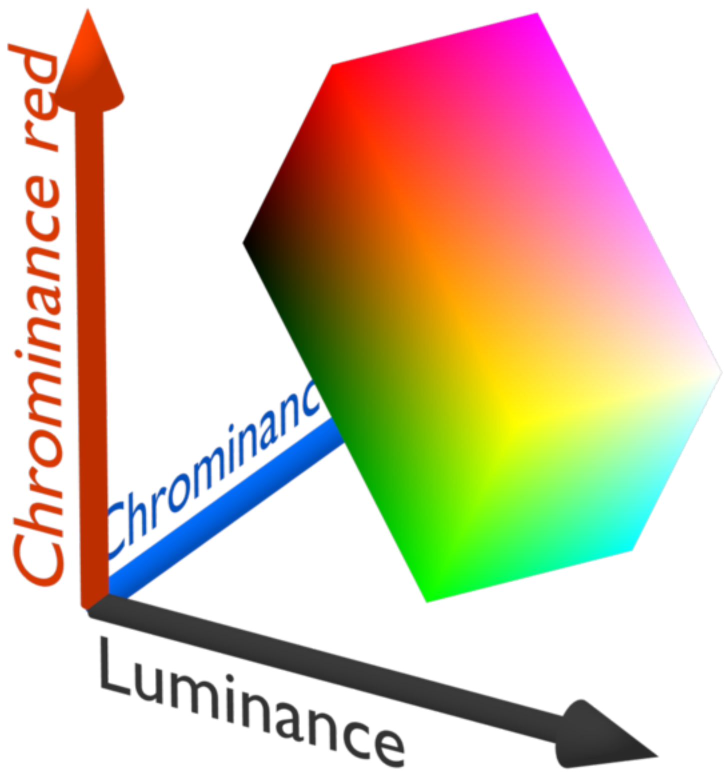


Figure A.2: Geometric Interpretation of YCbCr Color Space

process. *HSV* (or *HSI*) color space is often used by artists.

A.3 Video Formats

Two video formats are used while working on tests. First one is *Common Intermediate Format* (CIF). *CIF* is a format to standardize vertical and horizontal resolutions in pxels of *YCbCr* sequences. In this thesis, $288x352$ resolution is used for *CIF* standart.

Second one is *4CIF*. The only difference between them is resolution by convention. The resolution that is used to standardize in this case is $576x704$ which is simply four times bigger than *CIF*.

Appendix B

CODECS

A codec is a device or computer program capable of encoding or decoding a digital data stream or signal. *CODEC* word stands for *COder-DECoder* or, less commonly, *COmpressor-DECompressor*. Task of a codec is encoding a data stream or signal for transmission, storage or encryption, or decoding it for playback or editing. Codecs plays an important role on real time applications such as videoconferencing and streaming media. Moreover, some other applications like video editing applications take benefits of codec.

In digital videos, two type of codecs are used. First type is video codec which encodes and decodes video data stream. Other type is audio codec that works on audio stream. This thesis focuses on video codec identification. In digital video processing, the codec represents original video sequence by a *model* (an efficient coded representation that can be used to reconstruct an approximation of the video data). At the same time, ideally, the *model* has to represent the sequence using as few bits as possible. These two goals (compression efficiency and high quality) create a trade-off between each other[37].

A video encoder basically consists of three main functional units:

- Temporal Model: Attempts to reduce redundancy using time information by exploitingthe similarities between neighboring video frames.
- Spatial Model: Attempts to reduce redundancy by making use of similarities between neighbouring samples in the frame.

- Entropy Encoder: Attempts to reduce statistical redundancy and produce a compressed bit stream.

Five types of codecs used in this thesis. Two of them belong to *MPEG ISO*[38] standarts. First one is *MPEG-2* that suits for standards for digital television. Second codec belongs to MPEG ISO standarts is *MPEG-4* that suits of standarts for multimedia for the fixed and mobile web. Another codec is *MP4M* which is described as different implementation of MPEG-4. Fourth codec is *H.264/AVC*. *DIRAC* is the fifth codec. These codecs are explained in the below:

B.1 MPEG

MPEG is an acronym that stands for Moving Pictures Experts Group. MPEG was formed by Inetrnational Standards Organization (ISO) in order to specify standards for audio and video compression and transmission.

As stated above, a codec consists of two parts which are encoder and decoder. In some codecs, encoder is more complex than decoder which makes system asymmetrical. In asymmetrical coding, encoder has to be algorithmic whereas decoder just performs some fixed instructions. This approach presents a huge benefit in applications such as broadcasting. In broadcasting applications, there are small number of expensive encoders whereas number of cheap decoders are much more. It is obvious that benefits of asymmetrical coding is not great in point-to-point applications.

ISO standardized MPEG in a peculiar way. Instead of standardizing encoder, ISO applied this process to decoder. This means decoder shall interpret bitstream. A decoder that does an successfull interpretation of bitstream is called compliant. Every allowable bitstream has to be correctly interpreted by a compliant decoder. On the other hand, encoder produces a restricted subset of the possible codes.

Interesting part of MPEG is competition between encoders. In the fact that the bitstream is comliant, any coder design has to meet the standart.

However, some coder consturctions give better results than others. Therefore, different algorithms in encoder design provides diversity and competition. Since coders present different levels of and complexity, the user will have more options.

First standart released for audio and video compression was MPEG-1. MPEG-1 was designed to encode data into the bit rate of audio Compact Disc (CD). MPEG-1 applied a heavy downsampling on the images as well as considering picture rates of only 24-30Hz in order to deal with low bit rate. Quality of results was reasonable.

Next released standard was MPEG-2. Since MPEG-2 has been selected as the compression scheme for both Digital Video Broadcasting (DVB) and Digital Video/Versatile Disc (DVD), it has become important. Developments in scalability and multi-resolutional compression were ready by the time MPEG-2 was ready. Therefore, there have never been MPEG-3 release.

In order to achieve higher compression factors, MPEG-4 was designed. MPEG -4 uses more complex coding tools and than MPEG-2. Moreover, design of MPEG-4 is closer to computer graphics applications. It was expected for MPEG-4 to play an important role in internet and wireless delivery.

In 2001, International Telecommunication Union (ITU) Video Coding Experts Group(VCEG) joined with ISO MPEG to form Joint Video Team (JVT). This new team released a new stardant known as Advanced Video Coding (AVC), H.264 or MPEG-4 Part 10. AVC provided further refinements on MPEG-4 standarts.

B.1.1 MPEG-2

MPEG-2 is upgraded version of MPEG-1 by adding interlace capability, expanded range of picture sizes and bit rates. Since MPEG-2 is considered as an extension of MPEG-1, MPEG-2 decoders can handle MPEG-1 data easily. MPEG-1 bitstream can be thought as a subset of MPEG-2 bitstream. Therefore, MPEG-1 bitstream can be understood by MPEG-2 decoders.

MPEG-2 tries to solve too many applications with a single standard.

Consequently, MPEG-2 divided into *Profiles* and *Levels.* A Profile represents degree of complexity. On the other hand, a Level represents frame size or resolution. Not all levels are compatible with all Profiles. Although there is supposed to be 24 Profile-Level combination, some of them are not defined. Table B.1 shows Profile-Level pairs.

Bidirectional coding is not supported by *Simple Profile.* Consequently,

	Simple	Main	4:2:2	SNR	Spatial	High
High		4:2:0 1920x1152 90Mb/S				4:2:0 or 4:2:2 1920x1152 100 Mb/S
High 1440		4:2:0 1440x1152 60Mb/S			4:2:0 1440x1152 60Mb/S	4:2:0 or 4:2:2 1440x1152 60Mb/S
Main	4:2:0 720x576 15Mb/S NO B	4:2:0 720x576 15Mb/S	4:2:2 720x608 50Mb/S	4:2:0 720x576 15Mb/S		4:2:0 or 4:2:2 720x576 20Mb/S
Low		4:2:0 352x288 4Mb/S		4:2:0 352x288 4Mb/S		

Table B.1: Profiles and Levels in MPEG-2

output only contains I and P frames. This provides faster coding and decoding phase and works on simpler hardware.

The most important level of MPEG-2 is *Main Profile.* This level covers large proportion of uses. *Low Level* is designed for low resolution inputs which contains 352 pixels per line. Most of broadcast applications use Main Profile at Main Level. Main Profile at Main Level subset supports *Standard Definition Television* (SDTV). *High Level* maintains resolution with 16 : 9 format.

B.1.2 MPEG-4

MPEG-4 improves coding adding more functionalities to MPEG-2. MPEG-1 and MPEG-2 are focused to coding video frames. MPEG-4 also considers how shooting scene is created.

In MPEG-2, *macroblocks* used in motion compensation are regular fixed size. This brings some inefficiency in case of real moving objects. On the other hand, MPEG-4 codes moving objects as arbitrary shapes. This approach helps to code background independently from the objects on the foreground. Thus, motion information is stored in much reduced residual data.

The frame coding of MPEG-4 is known as *texture coding.* Texture coding approach loses less information coding of pixels, coefficients and vectors.

While an object moving, it changes its perspective. MPEG-4 defines this behavior by using a technique called *mesh coding.* The reconstruction of current frame is improved by warping another frame. Besides, MPEG-4 codes still frames using DCT or wavelets.

MPEG-4 adds up temporal scalability to spatial and noise scalability. This allows MPEG-4 quite forward from frame rates of film and television format. In addition, MPEG-4 still remains backward compatible. MPEG-4 also provides best frame quality for the available bit rate in network.

Another powerful feature of MPEG-4 is face and body animation. An image of a face (optionally body) can be animated to exhibit gestures to accompany speech in low bit rates by helps of special vectors.

Spatial compression is implemented based on DCT as in previous MPEG standards. Wavelet coding of stationary objects is newly introduced feature in MPEG-4, though. Since wavelet transformation decomposes an image into various resolutions, MPEG-4 becomes more advantageous in scalable systems.

B.1.3 AVC

Advanced Video Coding (AVC), or H.264, targets to compress moving frames in a form of 8-bit 4:2:0 coded pixel arrays. This approach brings more complexity with itself. On the other hand, AVC provides between two and two and a half times the compression factor of MPEG-2[39].

Macroblocks were transmitted only in a raster scan. This approach is fine when transmission is done on reliable channel. AVC is designed to work on imperfect channels. One of the most well known mechanisms is known

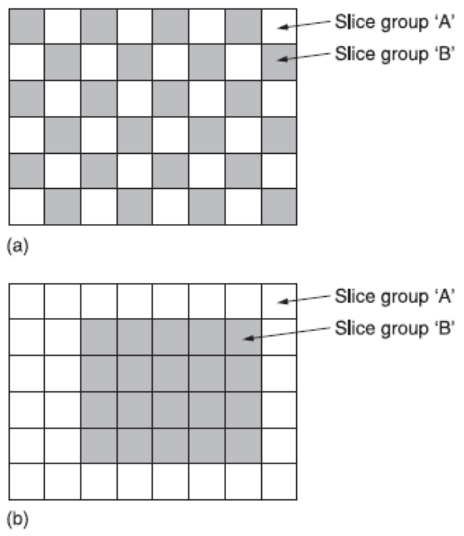


Figure B.1: (a) Chequerboarded Macroblocks (b) Important frame content is placed in one slice whereas background is in another.

as *Flexible Macroblock Ordering* (FMO). FMO divides frame into different regions along horizontal and vertical macroblock boundaries. Figure B.1(a) shows chequerboarded macroblocks. If the shaded macroblocks are transmitted in a different packet to unshaded ones, the loss of a packet causes a degraded frame instead of no frame. Figure B.1(b) presents another approach. In this approach, the important elements of the frame are located in one region whereas less important elements in another.

FMO contains *slice groups* that contain integer numbers of slices. Macroblocks inside the slice groups are transmitted in raster scan fashion. Location every macroblock in the correct place is task of decoder.

The bitstream of AVC is designed to be transmitted or recorded in various ways. This is another important advantage of AVC. *Video Coding Layer* (VCL) in a *Network Application Layer* (NAL) formats the date in a proper manner. This kind of output can be converted by AVC.

B.2 DIRAC

DIRAC is a video compression format developed by BBC Research team at the BBC. It is mainly developed for internet streaming to HDTV and electronic cinema. DIRAC supports both constant and variable bit rate operations. It also supports lossless and lossy coding. DIRAC uses *Wavelet Transformation* whereas MPEG family employs DCT. Architecture of wavelet coefficient coding can be seen in Figure B.2.

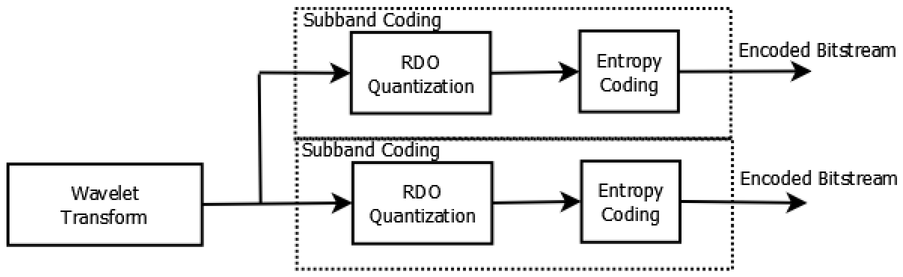


Figure B.2: Architecture of Wavelet Coding

As it can be seen from the Figure B.2, each wavelet subband is coded in turn. Both *Rate Distortion Quantization* (RDO) quantization and entropy coding of each band can depend on the coding of previously coded bands.

Dirac uses a parameter called QF to control the quality of the encoded frames. QF plays an important role since it is involved in the RDO processes of motion estimation and quantization as a Lagrangian multiplier, λ. The relation between λ and QF is given in Equation B.1:

$$\lambda = (10^{(10-QF)/2.5)})/16 \tag{B.1}$$

Since the QF controls the quality of the encoded video sequence by involving in the RDO processes of motion estimation mode decision and quantization, the accuracy of the motion estimation can be greatly reduced, especially for the lower-QF-encoding mode, which affects the subjective quality of the decoded video. So, the value of QF in Dirac should be set to at least 5 for low-quality encoding, even though Dirac allows the value of QF to range from 1 to 10.

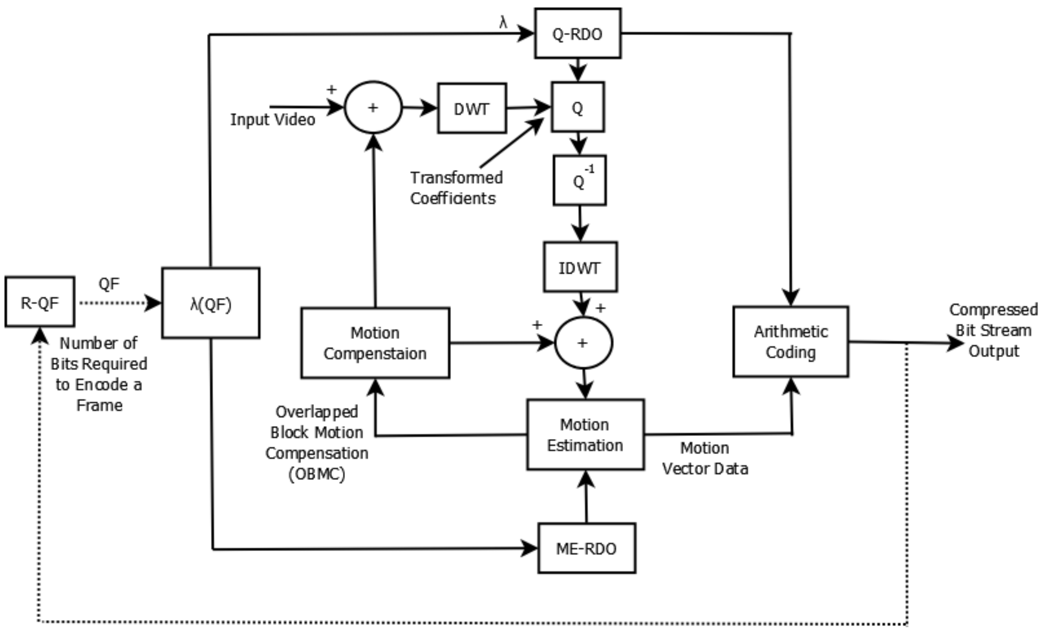


Figure B.3: Block diagram of Dirac encoder showing the proposed rate control idea using the R-QF model in blue colour.

Dirac defines three frame types. Intra-frames (I frames) are coded independently without reference to other frames in the sequence. Level 1 frames (L1 frames) and level 2 frames (L2 frames) are both inter-frames, which are coded with reference to other previously (and/or future) coded frames. The definition of the L1 and L2 frames are the same as with P and B frames in H.264, respectively. The encoder operates with standard GOP modes, whereby the number of L1 frames between I frames, and the separation between L1 frames, can be specified depending on the application. The detail explanation of the Dirac's GOP and intra-frames prediction structure can be found in [40]. General block diagram of dirac can be seen in Figure B.3.

Appendix C

Compression Quality Metrics

As mentioned above, there is a trade-off between compression efficiency and compression quality. This thesis focuses on fingerprints left by video compression. Therefore, our main concern is compression quality rather than compression efficiency. In order to define quality of the compression process, several computations metrics are defined.

C.1 Mean Square Error

Although *Mean Square Error (MSE)* is a well known error estimation method for statistics field, MSE is the root of the compression quality metrics. MSE of an estimator (predictor) means the difference between estimated (predicted) values and true values. Infact, MSE is a risk function that can be formed in several ways.

If $\hat{X}$ is a vector of n predictions and Y is the vector of the true values, then the MSE of predictor is:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2 \tag{C.1}$$

It can be easily derived that the MSE of an estimator $\hat{\sigma}$ with respect to the unknown parameter σ is:

$$MSE(\hat{\sigma}) = E[(\hat{\sigma} - \sigma)^2] \tag{C.2}$$

C.2 Signal To Noise Ratio

Signal to noise ratio (SNR) is measure to compare the level of desired signal to the level of noise. SNR is commonly quoted for electrical signals. However, it can be applied to any form of signal such as video and audio streams.

SNR ratio is by using power ratio. Definition is given below:

$$SNR = \frac{P_{signal}}{P_{noise}} \tag{C.3}$$

where P is average power. Power that is used to measure signal and noise must be equal in the system. If measurement is done across the same impedance, then SNR becomes:

$$SNR = \frac{P_{signal}}{P_{noise}} = (\frac{A_{signal}}{A_{noise}})^2 \tag{C.4}$$

where A is *root mean square (RMS) amplitude.* Generally, signals have a wide dynamic range, therefore, SNR is often described using logarithmic decibel scale.

$$SNR_{dB} = 10\log_{10}(\frac{P_{signal}}{P_{noise}}) = P_{signal,dB} - P_{noise_dB} \tag{C.5}$$

which is equivalently be written as

$$SNR_{dB} = 10\log_{10}(\frac{A_{signal}}{A_{noise}})^2 = 20\log_1 0(\frac{A_{signal}}{A_{noise}}) \tag{C.6}$$

C.3 Peak Signal To Noise Ratio

Peak Signal to Noise Ratio (PSNR) is derived from MSE and SNR. PSNR is the ratio between the maximum possible power of a signal and power of background noise. PSNR is generally used to measure of quality of codecs.

Let I be the an mxn monochrome image and $\hat{I}$ be the reconstruction of I. Then, MSE is:

$$MSE = \frac{1}{mn} \sum_{j=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - \hat{I}(i,j)]^2 \tag{C.7}$$

Then, PSNR is:

$$PSNR = 10 \log_{10}(\frac{max(I)^2}{MSE}) = 20 \log_{10}(\frac{max(I)}{\sqrt{MSE}}) \tag{C.8}$$

In the final form, PSNR can be simplified to form as:

$$PSNR = 20 \log_{10} max(I) - 10 log_{10}(MSE) \tag{C.9}$$

In this thesis, PSNR is used to measure compression quality.

Appendix D

Platforms and Tools

In the implementation phase, several tools are used. Video compression is done by using *ffmpeg* and *mencoder*. Chain compression test are done by using MATLAB. PSNR calculation is done by *C++* code.

D.1 FFmpeg

FFmpeg is the leading multimedia framework which is able to decode, encode, transcode, mux, demux, stream, filter and play pretty much anything that humans and machines have created. It supports the most obscure ancient formats up to the cutting edge. No matter if they were designed by some standards committee, the community or a corporation. It contains libavcodec, libavutil, libavformat, libavdevice, libswscale and libswresample which can be used by applications. As well as ffmpeg, ffserver, ffplay and ffprobe which can be used by end users for transcoding, streaming and playing.

The FFmpeg project tries to provide the best technically possible solution for developers of applications and end users alike. To achieve this, FFmpeg combines the best free software options available. FFmpeg keeps the dependencies on other libs low and maximizes code sharing between parts of itself.

FFmpeg provides various tools:

- ffmpeg: Command line tool to convert multimedia files between formats.

- ffserver: Multimedia streaming server for live broadcasts.
- ffplay: Simple media player based on SDL and the FFmpeg libraries.
- ffprobe: Simple multimedia stream analyzer.

and developer libraries:

- libavutil: Library containing functions for simplifying programming, including random number generators, data structures, mathematics routines, core multimedia utilities, and much more.
- libavcodec: Library containing decoders and encoders for audio/video codecs.
- libavformat: Library containing demuxers and muxers for multimedia container formats.
- libavdevice: Library containing input and output devices for grabbing from and rendering to many common multimedia input/output software frameworks, including Video4Linux, Video4Linux2, VfW, and ALSA.
- libavfilter: Library containing media filters.
- libswscale: Library performing highly optimized image scaling and color space/pixel format conversion operations.
- libswresample: Library performing highly optimized audio resampling, rematrixing and sample format conversion operations.

D.2 MEncoder

MEncoder (MPlayer's Movie Encoder) is a simple movie encoder, designed to encode MPlayer playable movies (e.g. AVI, DVD, VCD, MPG, MOV etc.) to other MPlayer playable formats. It can encode with various codecs, like MPEG-4 (DivX4) (one or two passes), libavcodec, PCM/MP3/VBR MP3 audio.

MEncoder features

- Encoding from the wide range of file formats and decoders of MPlayer.
- Encoding to all the codecs of FFmpeg's libavcodec.
- Video encoding from V4L compatible TV tuners.
- Encoding/multiplexing to interleaved AVI files with proper index.
- Creating files from external audio stream.
- 1, 2 or 3 pass encoding.
- VBR MP3 audio.
- PCM audio.
- Stream copying.
- Input A/V synchronizing (pts-based, can be disabled with -mc 0 option)
- Fps correction with -ofps option (useful when encoding 30000/1001 fps VOB to 24000/1001 fps AVI)
- Using our very powerful filter system (crop, expand, flip, postprocess, rotate, scale, RGB/YUV conversion)
- Can encode DVD/VOBsub and text subtitles into the output file.
- Can rip DVD subtitles to VOBsub format.

D.3 MATLAB

MATLAB (MATrix LABoratory) is a high-level language and interactive environment for numerical computation, visualization, and programming. MATLAB is used to analyze data, develop algorithms, and create models and applications. The language, tools, and built-in math functions enables to explore multiple approaches and reach a solution faster than with spreadsheets or traditional programming languages, such as C/C++ or Java.

MATLAB can be used for a range of applications, including signal processing and communications, image and video processing, control systems, test and measurement, computational finance, and computational biology. More than a million engineers and scientists in industry and academia use MATLAB, the language of technical computing.

D.4 C++

C++ is a statically typed, free-form, multi-paradigm, compiled, general-purpose programming language.C++ is developed by Bjarne Stroustrup who is a Danish computer scientist. Bjarne Stroustrup added object oriented approach, such as classes, and other enhancements to the C programming language.

C++ is used by hundreds of thousands of programmers in essentially every application domain. This use is supported by about a dozen independent implementations, hundreds of libraries, hundreds of textbooks, several technical journals, many conferences, and innumerable consultants. Training and education at a variety of levels are widely available.

Early applications tended to have a strong systems programming flavor. For example, several major operating systems have been written in C++ and many more have key parts done in C++. It is considered uncompromising low-level efficiency essential for C++. This allows to use C++ to write device drivers and other software that rely on direct manipulation of hardware under real-time constraints. In such code, predictability of performance is at least as important as raw speed. Often, so is compactness of the resulting system. C++ was designed so that every language feature is usable in code under severe time and space constraints[42].