UNIVERSITY OF CALIFORNIA, SAN DIEGO

Optimization and Learning based Video Coding

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Electrical Engineering

by

Cheolhong An

Committee in charge:

Professor Truong Q. Nguyen, Chair
Professor Serge J. Belongie
Professor Pamela C. Cosman
Professor William S. Hodgkiss
Professor Gert Lanckriet

2008

The dissertation of Cheolhong An is approved, and it is acceptable in quality and form for publication on microfilm & electronically:

_____

_____

_____

_____

_____

Chair

University of California, San Diego

2008

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

My decision to comeback school to pursue a Ph.D. degree is the most successful until now. Sincerely, I appreciate all the people to give the opportunity to me. Most of all, I thank my advisor, Prof. Truong Nguyen who gave a chance to join the Video Processing Lab at University of California San Diego and inspired various views on my research. I also really appreciated him to give an opportunity for me to find different ways in my life by providing academic and financial support during my time at UC San Diego. I would like to thank my committee members, Prof. Pamela Cosman, Prof. Willaim Hodgkiss, Prof. Gert Lanckriet and Prof. Serge Belongie for providing good lectures and comments, which help my research significantly. I offer my thanks to my fellow colleagues of the Video Processing Lab to help me with spending wonderful school life. Especially, I thank Meng-Ping Kao, Sanjeev Kumar, Min Li, Dung Vo, Karl Ni, Koohyar Minoo, Ryan Prendergast and Shay Har-Noy for helping my research as well as sharing good memories such as fishing, travel and workout. I am also grateful to Nickolaus Mueller, Meng-Ping Kao and Shay Har-Noy for proofreading my thesis. Finally, I wish to thank my mother, my late father, my older brother, my older sister and their families to always encourage me to finish Ph.D. courses and pray for my health. I sincerely appreciate my family's sacrifice. My wife and My lovely daughters, Doyoung An and Nayoung An make my life more beautiful and provide all my motivation. I love you forever.

Portions of the text of chapter 2 are adapted from material that has been submitted for publication as: C. An and T. Q. Nguyen, "Analysis of Lagrange Multiplier Selection in H.264 and Its adaptation using Classification-Maximization", under review with *the IEEE Transactions on Image Processing*, 2008, and C. An and T. Q. Nguyen, "Iterative Rate-Distortion optimization of H.264 with Constant Bit Rate Constraint" was published in *the IEEE Transactions on Image Processing*, vol. 17, pp. 1605-1615, Sep., 2008 and C. An and T. Q. Nguyen, "Iterative R-D optimization of H.264", in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, ICASSP-2008, Las Vegas, NV, USA, Mar., 2008. The dissertation author was the primary researcher for these publi-

cations, and the co-author directed and supervised the research which forms the basis for this chapter.

Portions of the text of chapter 3 are adapted from material that has been submitted for publication as: C. An and T. Q. Nguyen, "Analysis of Lagrange Multiplier Selection in H.264 and Its adaptation using Classification-Maximization", under review with *the IEEE Transactions on Image Processing*, 2008, and C. An and T. Q. Nguyen, "Analysis of Utility Functions for Video", in *Proceedings of the IEEE International Conference on Image Processing*, ICIP-2007, San Antonio, TX, USA, Sep., 2007, and C. An and T. Q. Nguyen, "Resource Allocation for TDMA Video Communication over AWGN using Cross-Layer Optimization Approach" was accepted to appear in *the IEEE Transactions on Multimedia*, 2008. The dissertation author was the primary researcher for these publications, and the co-author directed and supervised the research which forms the basis for this chapter.

Portions of the text of chapter 4 are adapted from material to be published as: C. An and T. Q. Nguyen, "Statistical Learning based Intra Prediction in H.264", in *Proceedings of the IEEE International Conference on Image Processing*, ICIP-2008, San Diego, CA, USA, Oct., 2008. The dissertation author was the primary researcher for this publication, and the co-author directed and supervised the research which forms the basis for this chapter.

Portions of the text of appendix A are adapted from material that has been submitted for publication as: C. An and T. Q. Nguyen, "Analysis of Lagrange Multiplier Selection in H.264 and Its adaptation using Classification-Maximization", under review with *the IEEE Transactions on Image Processing*, 2008, and C. An and T. Q. Nguyen, "Iterative Rate-Distortion optimization of H.264 with Constant Bit Rate Constraint" was published in *the IEEE Transactions on Image Processing*, vol. 17, pp. 1605-1615, Sep., 2008. The dissertation author was the primary researcher for these publications, and the co-author directed and supervised the research which forms the basis for this chapter.

VITA

1996	Bachelor of Science, Pusan National University, Republic of Korea
1996-1997	Graduate Teaching Assistant, Pusan National University, Republic of Korea
1997-1998	Graduate Research Assistant, Pusan National University, Republic of Korea
1998	Master of Science, Pusan National University, Republic of Korea
1998-2004	Senior Engineer, Digital Media R&D Center, Samsung Electronics
2004-2008	Graduate Research Assistant, University of California, San Diego
2008	Doctor of Philosophy, University of California, San Diego

PUBLICATIONS

Journal Papers

C. An and T. Q. Nguyen, "**Online and Batch Learning based Intra Prediction**", in preparation, 2008.

C. An and T. Q. Nguyen, "**Analysis of Lagrange Multiplier Selection in H.264 and Its adaptation using Classification-Maximization**", manuscript submitted to *the IEEE Transactions on Image Processing*, 2008.

C. An and T. Q. Nguyen, "**Iterative Rate-Distortion Optimization of H.264 with Constant Bit Rate Constraint**", *IEEE Transactions on Image Processing*, vol. 17, pp. 1605-1615, Sep., 2008.

C. An and T. Q. Nguyen, "**Resource Allocation for Error Resilient Video Coding over AWGN using Optimization Approach**", accepted, to appear in *the IEEE Transactions on Image Processing*, 2008.

C. An and T. Q. Nguyen, "**Resource Allocation for TDMA Video Communication over AWGN using Cross-Layer Optimization Approach**", accepted, to appear in *the IEEE Transactions on Multimedia*, 2008.

Conference Papers

C. An and T. Q. Nguyen, "**Statistical Learning based Intra Prediction in H.264**", in *Proceedings of the IEEE International Conference on Image Processing*, ICIP-2008, San Diego, CA, USA, Oct., 2008.

C. An and T. Q. Nguyen, "**Iterative R-D optimization of H.264**", in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, ICASSP-2008, Las Vegas, NV, USA, Mar., 2008.

C. An and T. Q. Nguyen, "**Cross-Layer Optimization for Video Communication over AWGN Channel**", in *Proceedings of the IEEE Global Communications Conference*, GLOBECOM-2007, Washington, DC, USA, Nov., 2007.

C. An and T. Q. Nguyen, "**Analysis of Utility Functions for Video**", in *Proceedings of the IEEE International Conference on Image Processing*, ICIP-2007, San Antonio, TX, USA, Sep., 2007.

C. An and T. Q. Nguyen, "**Low Complexity Scalable Video Coding**", in *Proceedings of the Asilomar Conference on Signals, Systems and Computers*, Monterey Bay, CA, USA, Oct., 2006.

ABSTRACT OF THE DISSERTATION

Optimization and Learning based Video Coding

by

Cheolhong An
Doctor of Philosophy in Electrical Engineering
University of California San Diego, 2008
Professor Truong Q. Nguyen, Chair

The complexity of video coding standards has increased significantly from H.262/MPEG-2 to H.264/AVC in order to increase coding efficiency. Complexity mainly was increased more by architecture than by algorithms: One 16x16 MB type in MPEG-2 was partitioned into various MB types such as 16x16, 8x16, 8x8, 4x4. Half pixel accuracy motion estimation was extended to support quarter pixel accuracy, and various simple directional filters were applied for intra prediction.

In this dissertation, we consider optimization and learning methods to solve video coding problems. In our approaches, complexity is mainly increased by algorithms to improve coding efficiency. Especially, we apply these methods for the Rate-Distortion (RD) optimization problem in H.264 and intra prediction as a new video coding scheme because they are highly related with numerical optimization and regression theories. For the RD optimization problem, we propose a general framework with consideration of temporal prediction dependency using the primal-dual decomposition and subgradient projection methods. As a result, optimality conditions among the Lagrange multipliers λ are derived for the optimal bit allocation. The proposed method is compared with the Rate Control (RC) algorithm in the reference software model (JM model) of H.264. In order to reduce the complexity of the proposed method, an adaptive Lagrange multiplier selection method is proposed in the RC algorithm using the Classification-Maximization (CM) algorithm. In addition, two variations of the CM algorithm, that is, Relaxed

CM (RCM) and Incremental CM (ICM) are proposed to improve the performance and avoid iterations. We compare λ of the proposed adaptive Lagrange multiplier selection methods with ones of the JM model and the greedy search.

Finally, we propose a new video coding scheme using learning methods. In particular, learning methods such as support vector regression and locally weighted learning are applied for intra prediction by means of batch and online learning. We present that online learning based intra prediction is better for video coding because of limited training time and nonstationary video sequences even though batch learning based intra prediction can achieve significant improvement in low-motion sequences. Experimental results show that online learning based video coding is promising for future video coding.

1 Introduction

H.264/AVC, which is the latest international video coding standard [1], was approved by ITU-T as Recommendation H.264 and by ISO/IEC as MPEG-4 part 10 Advanced Video Coding (AVC). H.264/AVC, which is denoted H.264 for simplicity hereafter, is designed for broad application areas such as broadcast over cable, DVD and multimedia streaming services over LAN. In order to support various applications, H.264 specifies 3 profiles which define a set of coding tools to generate interoperable bitstreams. In addition, it defines levels to constrain coding parameters such as picture sizes, coded bit rates, frame rates and memory requirements. All the decoders to support a specific profile must implement all the functions required for the profile. The base profile supports intra and inter coding without B slices and Context Adaptive Variable Length Coding (CAVLC) for entropy coding. The main profile includes Bi-directional inter coding (B slices), interlace, inter coding with weighted prediction using multiple references and Context Adaptive Binary Arithmetic Coding (CABAC). The extended profile extends the base profile with data partitioning for error resilience, inter coding for B slices and weighted prediction, and SP and SI slices to switch bit streams. Thus, the base profile considers video conferencing and wireless communication applications while the extended profile includes multiple video streaming applications, and the main profile can be used for broadcasting and video storage applications [2, 3].

H.264 has a similar structure to previous video coding standards such as MPEG-2 [4] and H.263 [5] as shown in Figure 1.1. H.264 processes Macro Blocks (MBs) to encode video sequences. For this work, a video sequence is divided into Groups Of Pictures (GOPs) and a GOP mainly includes three types of pictures: a

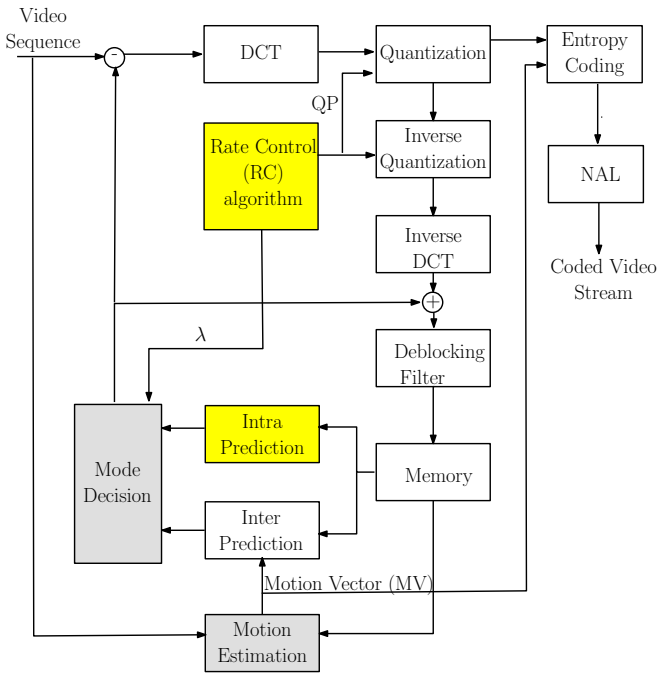


Figure 1.1: H.264/AVC video coding block diagram.

mandatory Intra (I) picture, Predictive (P) pictures and Bi-predictive (B) pictures. A picture can be partitioned into multiple slices which consist of MBs. In H.264, a MB is composed of Y, Cb and Cr components: a 16x16 pixels luminance (luma) component for Y brightness and two 8x8 pixels chrominance (chroma) components for Cb and Cr color information in the 4:2:0 sampling. Spatial and temporal redundancies are removed by intra and inter prediction, respectively. Intra prediction is conducted in the pixel domain, that is, adjacent neighboring reconstructed pixels are used to predict current pixels. Intra prediction can be applied in I, P and B pictures. Inter prediction performs Motion Estimation (ME) to find the best MB in the previous reconstructed frame for P pictures or the best MBs in the previous and future reconstructed frames for B pictures to minimize the Lagrangian cost which is the sum of distortion and coded bits multiplied by the Lagrange multiplier λ. In order to improve the performance of ME, a MB is partitioned into one 16x16, two 16x8, two 8x16 or four 8x8 blocks and a 8x8 block is further decomposed into two 8x4, two 4x8 or four 4x4 blocks. ME is separately applied to each block. The

best MB for intra or inter prediction is selected by a mode decision procedure to minimize the Lagrangian cost. Note that ME and the mode decision procedure are not specified by the H.264 standard [1].

After the mode decision procedure, a residual MB is obtained by subtracting the predicted MB from the current MB, and a 4x4 integer Discrete Cosine Transform (DCT) and quantization are applied to the residual MB to compact energy of the residual MB and to induce lossy compression as shown in Figure 1.1. Quantization levels and sign information are coded by CAVLC or CABAC. The entropy coded data are encapsulated into the Network Abstraction Layer (NAL) to map coded video data to the transport layers such as Real Time Transport (RTP)/Internet Protocol (IP) and MPEG-2 Transport Stream (TS). In order to remove mismatches between the encoder and the decoder, intra and inter prediction always use reconstructed (decoded) frames. The encoder generates reconstructed frames through inverse quantization, inverse DCT and the deblocking filter which are the same operations at the decoder except entropy decoding in Figure 1.1. The in-loop deblocking filter is incorporated to reduce blocking artifacts which are introduced by quantization.

If there are coded bit constraints from users or network capacity, the encoder needs to generate bit streams to satisfy the bit constraints. For this work, the encoder utilizes the Rate Control (RC) algorithm to determine frame bit constraints and to estimate Quantization Parameter (QP) for quantization step sizes. Finally, it derives the Lagrange multiplier λ for the mode decision procedure. The RC algorithm is not an integral part of the H.264 standard, but the standard requires the encoder to generate coded bit streams to be decoded by the Hypothetical Reference Decoder (HRD). The HRD specifies when coded bits arrive and are removed from the Coded Picture Buffer (CPB) without underflow and overflow [3]. The requirement of the HRD can be achieved by the RC algorithm with frame bit constraints.

In this dissertation, we will apply optimization and learning methods to solve video coding problems: the Rate-Distortion (RD) optimization problem, the Lagrange multiplier selection problem and intra prediction. In chapters 2 and 3,

we will propose a general framework to solve the RD optimization problem and adapt the Lagrange multiplier to video sequences for the RC algorithm. In chapter 4, a new video coding scheme will be proposed for intra prediction using online and batch learning methods.

In chapter 2, we will present the RD optimization problem with and without bit rate constraints. Especially, the relation between the RD optimization problem and the Lagrangian duality will be explained and then we will show that the RD optimization problem can be solved by the H.264 RC algorithm or the proposed method. First, we will show how the RC algorithm solves the RD optimization problem in section 2.2. Next, we will propose a general framework to solve the RD optimization problem using the primal-dual decomposition and subgradient projection methods [6–8] in section 2.3. In the framework, we will present how to reformulate the RD optimization problem and how to treat spatial and temporal prediction dependency using the reference software model [9]. As a result, we will derive optimality conditions among the pictures with temporal prediction dependency. Finally, experimental results will be discussed in section 2.4.

In chapter 3, we will propose an adaptive Lagrange multiplier selection method using the Classification-Maximization (CM) algorithm [10]. Even though the Lagrange multiplier can be obtained from the proposed method in section 2.3, complexity can be reduced by estimating the Lagrange multiplier λ for a given QP without solving the dual problem in the RD optimization problem. Generally, the RC algorithm uses a simple mapping function to derive λ for a given QP. Therefore, a constant λ is applied to encode various video sequences without considering different RD characteristics. In section 3.2, we will present how to reformulate the minimization of the Lagrangian cost problem as the maximization of the posterior probability problem and then the CM framework will be applied to adapt λ. In addition, we will propose relaxed CM and incremental CM methods to avoid CM iterations. Experimental results will be shown in section 3.3.

In chapters 2 and 3, we considered the RD optimization problem which is related with the RC algorithm and the mode decision procedure in Figure 1.1. The proposed methods in these chapters are compliant with the H.264 standard.

In chapter 4, we will propose a new video coding scheme using online and batch learning methods for intra prediction. It is the first approach to apply learning methods for prediction of video coding. Here, we intend to replace intra prediction in Figure 1.1 with the proposed method. However, the proposed method is not compatible with the H.264 standard since we propose new prediction methods. In section 4.2, we will briefly discuss features and limitations of intra prediction in H.264. In section 4.3, we will introduce the theoretical background of Support Vector Regression (SVR) and then apply SVR for batch learning based intra prediction. In order to consider nonstationarity of video sequences, we will apply an online learning method for intra prediction in section 4.5. Finally, the performance of batch and online intra prediction will be compared with H.264 in order to show some possibility to use learning methods for video coding in sections 4.4 and 4.6.

In chapter 5, the summary of contributions will be presented and their extensions will be discussed.

2 Rate-Distortion Optimization

2.1 Introduction

After Rate-Distortion (RD) optimization was introduced for video compression using the Lagrange multiplier [11, 12], there were many methods to reduce complexity in deciding MB modes and Motion Vectors (MVs) for the given Lagrange multiplier λ. Even though the RD optimization method is not mandatory for video compression standards such as H.264 [1], it is a main part of video coding to improve coding efficiency [12, 13]. In this section, we will briefly review the relation between the RD optimization and the previous work. The RD optimization with an inequality constraint in a picture is mathematically formulated as follows:

$$\begin{aligned} \min_{\mathbf{m}} \quad & \sum_{n=1}^{N} d_n(\mathbf{m}_n) \\ s.t. \quad & \sum_{n=1}^{N} x_n(\mathbf{m}_n) \leq X_F \end{aligned} \tag{2.1}$$

where $\mathbf{m}_n = (M_n, \mathbf{MV}_n, QP_n, \mathbf{Ref}_n)$ is a vector of a MB mode, MVs, QP and reference frames for inter prediction at the nth MB. N is the number of MBs in a picture and X_F is a bit constraint of a picture. $d_n(\mathbf{m}_n)$ and $x_n(\mathbf{m}_n)$ are distortion and coded bits of the nth MB, respectively. The optimization problem (2.1) can be solved by the Lagrangian duality in order to obtain the optimal solution if the problem is a convex optimization problem and satisfies Slater's condition [14]. Slater's condition is easily satisfied since there are $\mathbf{m}$ vectors which make the sum of coded bits less than X_F, but the problem (2.1) is mathematically not a convex optimization problem since the distortion function $d_n(\mathbf{m}_n)$ is not a convex

function [11] and the feasible set $\mathbf{m}_n$ is not a convex set [14]. However, a near optimal solution of the primal problem (2.1) can be obtained if the duality gap is small [6]. Therefore, the Lagrange duality is applied and the dual function of the primal problem (2.1) is

$$q(\lambda) = \min_{\mathbf{m}} \sum_{n=1}^{N} \Big(d_n(\mathbf{m}_n) + \lambda x_n(\mathbf{m}_n) \Big) - \lambda X_F \tag{2.2}$$

and its dual problem is

$$\max_{\lambda \geq 0} q(\lambda) \tag{2.3}$$

If we find the optimal solution of the dual problem, we can obtain the solution of the primal problem (2.1) after solving (2.2). In the JM model [9], independence among MBs is assumed in order to avoid the optimization problem for all the MBs. Consequently, the problem (2.2) becomes

$$q(\lambda) = \sum_{n=1}^{N} \min_{\mathbf{m}_n} \Big(d_n(\mathbf{m}_n) + \lambda x_n(\mathbf{m}_n) - \lambda X_n \Big) \tag{2.4}$$

where $\sum_{n=1}^{N} X_n = X_F$. Now, we can perform the RD optimization at each MB. In order to reduce the loss of coding efficiency from the independent assumption, references [11, 15, 16] solved the dependent optimization problem (2.2) using the dynamic programming without considering frame-level dependency or λ. Reference [17] applied the Viterbi Algorithm (VA) with the assumption that distortion and coded bits are only functions of QP.

In (2.4), λ and the bit constraint X_n are constants. As a result, the solutions are equivalent as follows:

$$\arg\min_{\mathbf{m}_n} \Big(d_n(\mathbf{m}_n) + \lambda x_n(\mathbf{m}_n) \Big) = \arg\min_{\mathbf{m}_n} \Big(d_n(\mathbf{m}_n) + \lambda x_n(\mathbf{m}_n) - \lambda X_n \Big) \tag{2.5}$$

Thus, the problem (2.4) becomes the minimization of the Lagrangian cost which is $\min_{\mathbf{m}_n} d_n(\mathbf{m}_n) + \lambda x_n(\mathbf{m}_n)$. However, it does not imply that the MB bit constraint X_n and the frame bit constraint X_F are useless because the bit constraints affect the optimal solution of λ in the dual problem (2.3). Therefore, we should carefully choose λ if there is a bit constraint.

The JM model applies the Rate Control (RC) algorithm to derive the optimal λ instead of solving directly the dual problem (2.3). The RC algorithm, which will be discussed in section 2.2, first decides the bit constraints X_n and X_F from a user bit per second (bps) constraint and then applies the quadratic model (2.6) to estimate QP to satisfy the bit constraints. Next, λ which corresponds to the solution of the dual problem (2.3) is derived from the mapping function (2.7) for the estimated QP. Finally, the primal solution $\mathbf{m}_n$ is obtained by solving the problem (2.5) for a given λ.

$$X_F = \gamma \frac{MAD}{Q_{step}} + \xi \frac{MAD^2}{Q_{step}^2}, \quad Q_{step} = \nu 2^{\frac{QP-12}{6}} \tag{2.6}$$

$$\lambda = \kappa 2^{\frac{(QP-12)}{3}} \tag{2.7}$$

where κ is a function of picture type (I, P, B), the number of referenced frames, and QP. γ and ξ are estimated using linear regression based on Mean Absolute Difference (MAD) and target bits X_F. ν is a function of QP. Estimation of X_F from the quantization step size Q_{step} in (2.6) was studied in [18,19] and the relation between λ and QP in (2.7) was derived from [12,20–22]. Thus, the JM model does not directly solve the dual problem (2.3). For a given λ, the JM model minimizes the Lagrangian function, that is, solves the problem (2.5). However, if there is no bit rate constraint, we can choose any QP to derive λ from (2.7). Consequently, the JM model has two coding modes: with or without a bit constraint.

Without a bit rate constraint, users specify any QP and a GOP structure, and then the JM model solves the problem (2.5) to determine MB modes and MVs for a given QP, reference frames and λ. As a result, users cannot expect how many bits are generated after encoding. However, if the generated bits are considered as the bit constraint X_f, the specified QP always satisfies the bit constraint X_f. λ from the specified QP is always the optimal dual solution to satisfy the Karush-Kuhn-Tucker (KKT) [14] conditions. With a bit rate constraint, users specify a desired bps and a GOP structure and then the JM model solves the problem (2.5) with the independent assumption. Although λ and QP are obtained from equations (2.5) and (2.6), the bit constraint X_F in (2.5) should be derived from user bit rate constraint because user bit rate constraint is average bits per second,

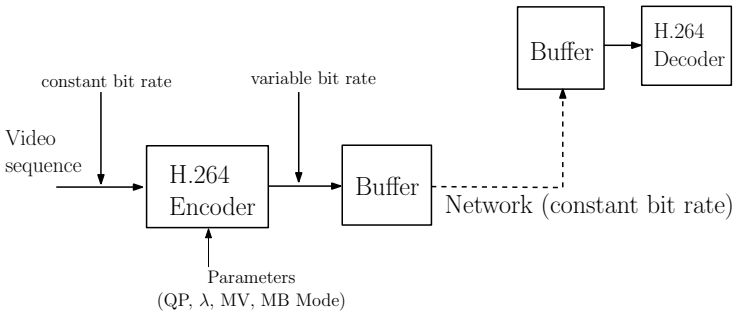


Figure 2.1: Example of a video streaming system.

but not a MB-level or a frame-level bit constraint. Therefore, the bit constraints of a MB or a frame, which are generalized as a basic unit (a group of MBs) [23,24], and a GOP should be derived from a given user bit rate constraint which will be explained in subsection 2.2.2. Without loss of generality, a basic unit is considered as a MB or a frame in this dissertation. If we estimate target bits for a basic unit, the remaining parameters can be obtained from (2.6) and (2.7).

Even though the RC algorithm induces loss of performance which will be shown in section 2.4, it is well known that the RC algorithm is necessary in the video streaming applications to satisfy network bit rates without a buffer overflow or underflow. In this section, we briefly mention the necessity of the RC algorithm with a simple example of the video streaming system in Figure 2.1. Original video sequences have constant bit rates according to the frame rate and frame size, but output bit rates of the video encoder become variable bit rates since intra and inter prediction errors of each frame highly depend on the correlation among the frames and within a frame. Thus, variable coded bits should be smoothed out to the constant or variable network through the buffer as shown in Figure 2.1. The virtual buffer fullness, which allows negative fullness, is illustrated in Figure 2.2 after encoding with and without RC. Decrease of the buffer fullness indicates that input rates of the buffer are lower than output rates. Otherwise, input rates are higher than output rates. The buffer fullness of encoding without RC, that is, encoding with QP fixed for all the frames highly fluctuates. The actual buffer fullness is around zero before the 70th frame since there is no negative buffer fullness and

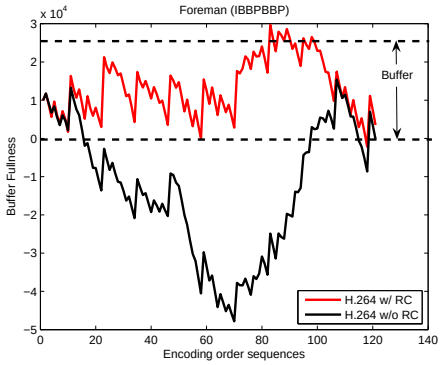


Figure 2.2: Virtual buffer fullness with and without RC.

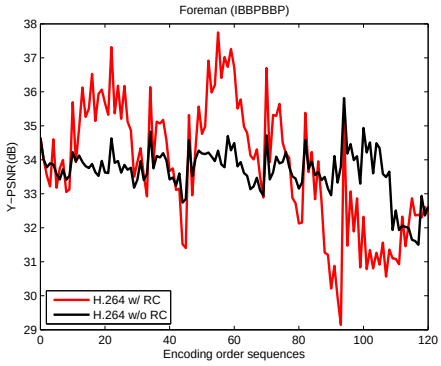


Figure 2.3: Y-PSNR (dB) of decoded sequences with and without RC.

then the buffer fullness increases. This phenomenon results from the increase of prediction errors because of high motion and scene changes after the 70th frame. Consequently, video streaming applications without RC require a larger buffer to compensate bit fluctuation. However, we do not know how large a buffer can compensate the bit fluctuation before encoding the whole sequence. Furthermore, different sequences, which may never be coded before, require different buffer sizes which are not known. Therefore, the encoder needs to control bit rates to prevent a buffer overflow and underflow for a given buffer size. Figure 2.2 shows that coded bits of the encoder with RC fluctuate within the buffer size. The encoder with RC mainly changes QP in order to control bit rates which induces larger variations of quantization distortion as shown in Figure 2.3. Even though quality of coded

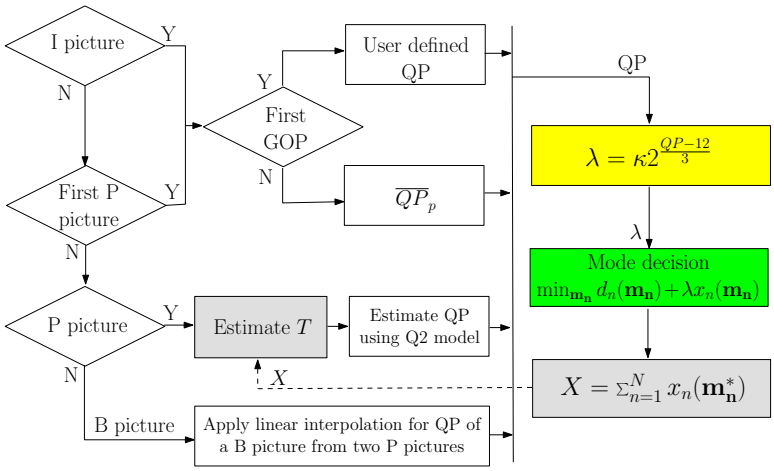


Figure 2.4: Rate control algorithm in the JM model.

sequences without RC is better because of lower variations of Peak Signal-to-Noise Ratio (PSNR), buffer management is required and accomplished through the RC algorithm for a given bit constraint with minimum distortion. This operation is mathematically formulated as the RD optimization problem (2.1).

2.2 H.264 Rate-Distortion Optimization

The JM model applies the RC algorithm to solve the RD optimization problem (2.1). The RC algorithm consists of four parts: bit allocation for basic units from a user bit constraint or an available network capacity, estimation of QP to satisfy the bit constraint of a basic unit (2.6) and selection of λ from the estimated QP (2.7). Finally, compatibility of the Hypothetical Reference Decoding (HRD) is checked.

2.2.1 Quantization Parameter Estimation

In this subsection, we estimate QP to satisfy the target bit constraint T for a picture within a GOP. Figure 2.4 illustrates the RC algorithm in the JM model [9]. A GOP consists of one I picture with or without several P and B pictures. As an

example, we assume that a GOP is composed of I, P and B pictures. In the first GOP of a sequence, QP of I and the first P pictures are set to a user defined QP, otherwise, they are $\overline{QP}_p$ which is an average QP of P pictures in the previous GOP:

$$\overline{QP}_p = \frac{1}{N_p}\sum_{n=1}^{N_p} QP_p^n - \min\left(\frac{N_p + N_b + 1}{15}, 2\right) \tag{2.8}$$

where N_p, N_b and QP_p^n are the number of P pictures and B pictures and QP of the nth P picture within a GOP, respectively. The second term of (2.8) is included with heuristic consideration of temporal prediction dependency which will be discussed in detail in subsection 2.2.4. The second term considers that better quality of the first I and P pictures reduces prediction errors of the other pictures if there are temporally related pictures which are P or B pictures within a GOP. For a given QP of I and P pictures, λ is derived from (2.7) and then the mode decision procedure is performed to decide MB modes and MVs as shown in Figure 2.4. The number of coded bits X is used to estimate target bits T which will be discussed in subsection 2.2.2. QP of the other P pictures within a GOP is estimated to satisfy target bits T using the quadratic model in (2.6). Here, we use an approximate bit constraint T instead of the true bit constraint X for the Fth frame. The main reason is that we have the chicken and egg dilemma. Coded bits X is known after the mode decision but QP is necessary to derive λ for the mode decision. QP can be estimated for the given bit constraint X. This chain can be broken by using the approximate bit constraint T instead of X. The Q2 model in (2.6) was derived by [18, 25] as follows:

$$X(D) = \ln\left(\frac{\sigma}{D}\right), 0 < D < \sigma \tag{2.9}$$

$$X(Q_{step}) = \left(\frac{\sigma}{Q_{step}} - 1\right) - \frac{1}{2}\left(\frac{\sigma}{Q_{step}} - 1\right)^2 + X_3, \quad \frac{\sigma}{D} > 1 \tag{2.10}$$

$$T(Q_{step}) \approx \frac{2\sigma}{Q_{step}} - \frac{\sigma^2}{2Q_{step}^2} = \gamma\frac{MAD}{Q_{step}} + \zeta\frac{MAD^2}{Q_{step}^2} \tag{2.11}$$

$$= \varrho_0\frac{MAD}{Q_{step}} + \varrho_1\frac{MAD}{Q_{step}^2} \tag{2.12}$$

where $\frac{1}{\sqrt{2}}\sigma$ is the standard deviation of DCT coefficients. The equation (2.10) is derived from the Talyor series of natural log, i.e. ln at 1 assuming that the quantization step size Q_{step} measures the distortion D. In (2.11), high order terms, which are denoted as X_3, are ignored and σ is assumed to be proportional to MAD [25]. Even though (2.11) is more reasonable, the JM model [9] follows the proposal in [26] as in (2.12). MAD of a current P picture is linearly predicted from MAD of the previous P picture which is denoted as MAD_{prv} since it is not known before encoding: $MAD = \rho_1 MAD_{prv} + \rho_0$ where model parameters ρ_1 and ρ_0 are updated after encoding the current P picture. After considering header bits H in (2.12), the quantization step size Q_{step} is

$$Q_{step} = \frac{2\varrho_1 MAD}{\sqrt{4\varrho_1 MAD(T-H) + \varrho_0^2 MAD^2} - \varrho_0 MAD} \tag{2.13}$$

QP is obtained from the mapping function between QP and Q_{step} in (2.6) where $\nu(QP\%6)$ is a function of QP mode 6 as defined in H.264 [1]. After encoding a P picture, the model parameters ϱ_0 and ϱ_1 are updated as follows [18, 26]:

$$\varrho_1 = \frac{N_w \sum_{i=1}^{N_w} R_i - \left(\sum_{i=1}^{N_w} Q_{step_i}^{-1}\right)\left(\sum_{i=1}^{N_w} Q_{step_i} R_i\right)}{n\sum_{i=1}^{N_w} Q_{step_i}^{-2} - \left(\sum_{i=1}^{N_w} Q_{step_i}^{-1}\right)^2} \tag{2.14}$$

$$\varrho_0 = \frac{\sum_{i=1}^{N_w}\left(Q_{step_i} R_i - Q_{step_i}^{-1}\varrho_1\right)}{N_w} \tag{2.15}$$

where N_w is the number of the previous pictures for regression and $R_i = \frac{X_i - H_i}{MAD_i}$. The windows size N_w is set to 20 at the maximum in the JM model [9].

QP of B pictures is directly derived from adjacent QP of I or P pictures without estimating target bits as shown in Figure 2.4. If the number of B pictures between I or P pictures is one, QP of the tth B picture is

$$QP_t = \begin{cases} QP_{t-1} + 2 & \text{if } QP_{t-1} = QP_{t+1} \\ \frac{QP_{t-1}+QP_{t+1}}{2} + 1 & \text{otherwise} \end{cases} \tag{2.16}$$

If the number of B pictures between I or P pictures L greater than one, QP of the

kth B picture among B pictures is

$$QP_{t+k} = QP_{t-1} + QP_s + \min\left(\left|\frac{QP_{t+L} - QP_{t-1}}{L-1}k\right|, 2k\right) \tag{2.17}$$

where $k = 0, ..., L-1$ and

$$QP_s = \begin{cases} \max(QP_{t+L} - QP_{t-1} + 2L, -3) & \text{if } QP_{t+L} - QP_{t-1} + 2L \leq 1 \\ 2 & \text{otherwise} \end{cases} \tag{2.18}$$

The second term in (2.17) is heuristically added to consider that QP of nonreferenced pictures is higher than one of referenced pictures. The third term in (2.17) is a linear interpolation method to estimate QP between QP of I or P pictures where QP difference of adjacent B pictures is limited by ± 2.

2.2.2 Basic Units Bit Allocation

The initial ith GOP bits are decided from a user bit rate constraint or network capacity bps as follows:

$$X^z_{GOP} = \frac{X^z_{net}}{f_{rate}} N_{GOP} - B^z_c \tag{2.19}$$

where N_{GOP} is the number of frames in a GOP and f_{rate} is the number of frames per second. X^z_{net} is the available network capacity at the last frame of the $(i-1)$th GOP where $z = (i-1) \cdot N_{GOP}$. Note that X^z_{net} is a minimum value of a user bit constraint or an available network capacity but we call it an available network capacity for simplicity. When the ith GOP bits are assigned, the buffer fullness B^z_c of the previous GOP is considered. Buffer fullness B^t_c at t is calculated from the previous buffer fullness B^{t-1}_c, current coded bits X^t and transmitted bits via network $\frac{X^t_{net}}{f_{rate}}$ as follows:

$$B^t_c = \min\left(\max\left(B^{t-1}_c + X^t - \frac{X^t_{net}}{f_{rate}}, 0\right), B_e\right) \tag{2.20}$$

where B_e is the maximum buffer size which is 2.56 times the user bps constraint in the JM model [9]. Thus, B_e buffers video streaming data for about 2.5 second (sec) at the desired bps. Figure 2.5 illustrates the fluid traffic model of (2.20).

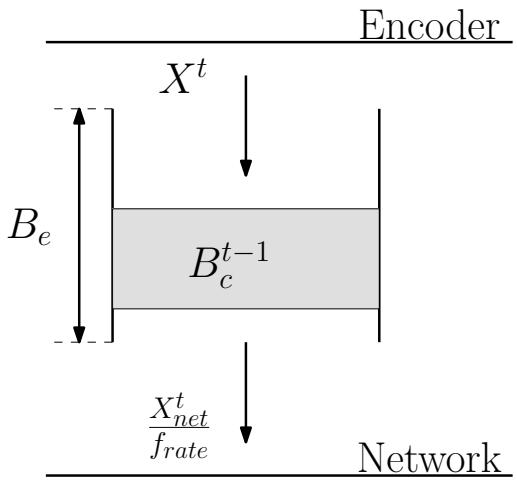


Figure 2.5: Buffer fullness B_c^t.

Note that average bits $\frac{X_{net}^t}{f_{rate}}$ are transmitted to the network from the buffer while coded bits of one frame X^t are buffered from the encoder. If we consider variations of the network capacity and coded bits, available GOP bits are updated frame by frame as follows:

$$X_{GOP}^t = X_{GOP}^{t-1} - X^t + \frac{X_{net}^t - X_{net}^{t-1}}{f_{rate}}(N_{GOP} - t) \tag{2.21}$$

where $X_{GOP}^{t-1} - X^t$ are the remaining GOP bits after encoding one frame at t and the third term of (2.21) represents a variation of the network capacity from $t-1$ to t which is assumed to be persistent in the remaining GOP frames $N_{GOP} - t$.

Next, we consider bit allocation to each frame for a given GOP bit constraint X_{GOP}. Here, we confine a basic unit as a frame since MB-level RC is far from the optimal bit allocation within a frame even though RC can more tightly satisfy bit constraints. In subsection 2.3.2, we will show that all the λ of MBs are equal for the optimal bit allocation of a frame with spatial independence. If a basic unit is smaller than the maximum number of MBs in a frame, the RC algorithm can assign different QP, that is, different λ to each basic unit. In contrast, if a frame is considered as a basic unit, the same λ are assigned to all the MBs within a frame which induces similar quality within a frame. For a frame-level bit allocation, we

define target bits T for P pictures as follows:

$$T_r^t = \frac{X_{GOP}^t}{N_p^t + \zeta N_b^t}, \quad \zeta = \frac{\overline{W}_b}{\overline{W}_p} \tag{2.22}$$

$$B_p^{t+1} = B_p^t - \frac{B_c^{z+2}}{N_P^{z+2}}, \quad T_p^{z+2} = B_c^{z+2} \tag{2.23}$$

$$T_n^r = \frac{X_{net}^t}{f_{rate}} + \gamma_P (B_p^t - B_c^t), \quad \gamma_p = 0.25 \text{ if } N_b^{z+1} > 0, \text{ otherwise } 0.5 \tag{2.24}$$

$$T^t = \beta_p T_r^t + (1 - \beta_p) T_n^t, \quad \beta_p = 0.9 \text{ if } N_b^{z+1} > 0, \text{ otherwise } 0.5 \tag{2.25}$$

where N_p^t and N_b^t are the numbers of unencoded P and B pictures within a GOP and $z + 2$ denotes an index within a GOP after encoding I and P pictures. Note that target bit allocation is only applied to P pictures after encoding an I picture and the first P picture for every GOP as shown in Figure 2.4. QP of the first I and P pictures in each GOP are directly estimated from the previous GOP or a given initial QP without target bit allocation which was explained in subsection 2.2.1. Target buffer level T^t in (2.25) is the weighted sum of T_r^t and T_n^t in the tth frame. For simplicity, we omit the frame index t. In (2.22), the remaining GOP bits X_{GOP} after encoding the first I and P pictures are divided by the remaining number of P and B pictures within a GOP to estimate target bits. Average weights for P pictures $\overline{W}_p$ can be different from average weights for B pictures $\overline{W}_b$ to allocate more bits for P pictures. Target bits T_n from the network in (2.24) are estimated from the weighted sum of the network capacity $\frac{X_{net}}{f_{rate}}$ and the difference between estimated buffer fullness and actual buffer fullness. In (2.23), the buffer fullness after encoding all P pictures within a GOP is intended to be zero where B_c^{z+2} is the buffer fullness after encoding the first I and P pictures. More detailed explanations are available in [23, 24, 27, 28].

2.2.3 Hypothetical Reference Decoder

One virtual decoder which is known as the Hypothetical Reference Decoder (HRD) in H.263 and H.264 [1, 29] and the Virtual Buffering Verifier (VBV) in MPEG-2 [30] forces the encoder to generate bit streams such that the hypothetical decoder buffer does not underflow nor overflow. The HRD, which is a normative

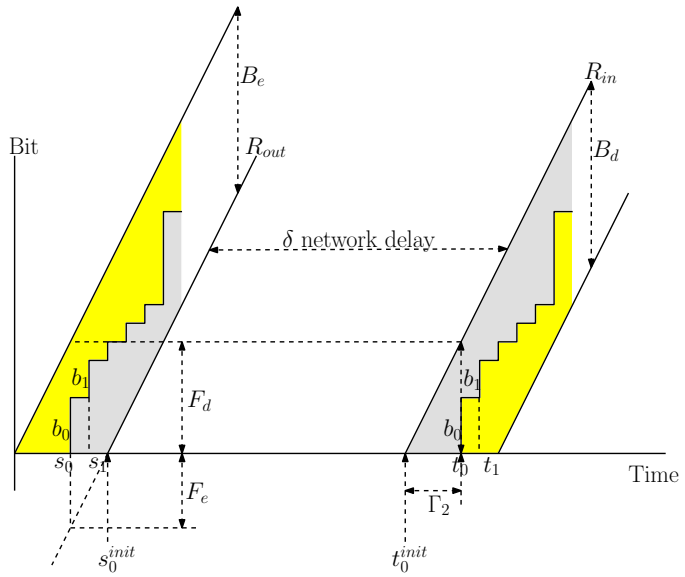


Figure 2.6: Buffer fullness of the encoder and the decoder with constant network delay.

part of the video coding standards [1, 29, 30], is conceptually connected to the output of the encoder and it consists of the decoder buffer, the decoder, and the display unit. The mathematical leaky bucket model was employed to characterize the hypothetical decoder and its input buffer called the Coded Picture Buffer (CPB) [1,31]. In the previous subsection, we estimate target bits T in order not to overflow nor to underflow the encoder buffer which is called the Encoder Picture Buffer (EPB) without considering the decoder side buffer, that is, the CPB. The RC algorithm in the encoder was studied to generate HRD compliant bitstreams such that the CPB in the decoder neither overflows nor underflows [24, 32–34]. As a result, target bits T for each picture are clipped with an upper and lower bound which are derived from the leaky bucket model.

The leaky bucket model contains three parameters (R, B, F_d) where R is an input rate, B is a decoder buffer size and F_d is initial buffer fullness. In Figure 2.6, we assume that the encoder buffer size B_e is equal to the decoder buffer size B_d, i.e. $B = B_e = B_d$. Furthermore, the output rate from the encoder buffer R_{out} is equal to the input rate to the decoder buffer R_{in}, i.e. $R = R_{in} = R_{out}$

where the rate (bits per second) R is constant since the slope of R is constant. We also assume that the network delay δ is constant. The HRD model assumes instantaneous encoding and decoding so that the plot of cumulative generated bits looks like a stair step. The ith picture within a sequence is instantaneously encoded and buffered with b_i bits at time s_i in Figure 2.6. The ith encoded picture is transmitted to the CPB and then b_i bits are instantaneously extracted at time t_i and decoded. In Figure 2.6, the buffered data are illustrated in the grey area and the available buffer size is marked as the yellow area. In Figure 2.6, when the first encoded bits b_0 are buffered to the EPB, the leaky bucket model assumes that the initial buffer fullness of the EPB is F_e. However, there are no bits in the buffer. The first bit of b_0 is transmitted into the network at time s_0^{init} and the final bit of b_0 is transmitted at time s_0^{fin} where

$$s_0^{init} = s_0 + \frac{F_e}{R}, \quad s_0^{fin} = s_0^{init} + \frac{b_0}{R} \tag{2.26}$$

The initial bit of the subsequent picture at the ith picture is transmitted at time $s_{i+1}^{init} = \max(s_i^{fin}, s_{i+1})$. In order to fully use the network capacity, the initial bit of the subsequent picture is continuously transmitted after sending the last bit of the previous picture:

$$s_{i+1}^{init} = s_i^{fin} = s_i^{init} + \frac{b_i}{R} = s_0^{init} + \sum_{k=0}^{i} \frac{b_k}{R} \tag{2.27}$$

At the decoder, the first bit of b_0 is stored into the CPB at time

$$t_0^{init} = s_0^{init} + \delta = s_0 + \frac{F_e}{R} + \delta \tag{2.28}$$

but decoding is delayed until the buffer fullness of the CPB reaches F_d as shown in Figure 2.6. The final bit of any picture i is stored into the CPB at time

$$t_i^{fin} = s_i^{fin} + \delta = s_i^{init} + \frac{bi}{R} + \delta = t_i^{init} + \frac{bi}{R} \tag{2.29}$$

In order to prevent the CPB from underflowing, the removal time of the ith picture from the CPB t_i is greater than or equal to the buffered time of the last bit t_i^{fin}, that is, $t_i \geq t_i^{fin}$. It means that when the decoder starts to extract the first bit of

the ith picture, the last bit of the ith picture exists in the CPB. Thus,

$$t_i = s_i + \frac{F_e}{R} + \frac{F_d}{R} + \delta = s_0 + \sum_{k=0}^{i-1}(s_{k+1} - s_k) + \frac{F_e}{R} + \frac{F_d}{R} + \delta \tag{2.30}$$

$$t_i^{fin} = s_i^{init} + \frac{b_i}{R} + \delta = s_0 + \frac{F_e}{R} + \sum_{k=0}^{i-1}\frac{b_k}{R} + \frac{b_i}{R} + \delta \tag{2.31}$$

$$t_i \geq t_i^{fin}, \quad b_i \leq F_d + \sum_{k=0}^{i-1}\Big((s_{k+1} - s_k)R - b_k\Big) \doteq U_b \tag{2.32}$$

where (2.30) is derived from $t_i = s_i + \frac{F_e}{R} + \frac{F_d}{R} + \delta$ and $s_i = s_0 + \sum_{k=0}^{i-1}(s_{k+1} - s_k)$ in Figure 2.6, and (2.31) is obtained from (2.29), (2.27) and (2.26). Finally, we can derive the upper bound U_b of coded bits for the ith picture to prevent the CPB from underflowing in (2.32). The upper bound U_b is just buffer fullness of the CPB with the assumption $(s_{k+1} - s_k) = (t_{k+1} - t_k)$. Thus, b_i bits must be smaller than the buffer fullness of the CPB just before b_i bits are extracted for decoding in order to avoid underflow. Moreover, if we increase the initial buffer fullness of the CPB F_d, the RC algorithm can allocate more bits for pictures such that it can reduce PSNR variations among the pictures which is illustrated in Figures 2.2 and 2.3. Even though an increase of initial buffer fullness F_d delays the first decoding time, an overall system delay is not changed if the buffer size B, that is, the sum of F_d and F_e, is fixed for the given network delay δ from (2.30). The JM model [9] sets F_d to $0.8B$ where $B = 2.56R$. Thus, the decoder waits for around 2 seconds before extracting data from the CPB.

In order to prevent the CPB from overflowing, the buffer fullness of the CPB is less than or equal to the decoder buffer size B_d after removing the coded bits of the ith picture:

$$B = B_d \geq F_d + \sum_{k=0}^{i-1}\Big(R(s_{k+1} - s_k) - b_k\Big) + R(s_{i+1} - s_i) - b_i \tag{2.33}$$

$$b_i \geq -F_e + \sum_{k=0}^{i-1}\Big(R(s_{k+1} - s_k) - b_k\Big) + R(s_{i+1} - s_i) \doteq L_b \tag{2.34}$$

where $B = F_d + F_e$. The JM model [9] sets F_e to 0 in order to allocate more bits to pictures for the given buffer size B. The upper bound (2.32) and the lower bound

(2.34) are recursively derived as follows [32, 33]:

$$U_b^i = U_b^{i-1} + \frac{R}{f_{rate}} - b_{i-1}, \quad U_b^0 = F_d \tag{2.35}$$

$$L_b^i = L_b^{i-1} + \frac{R}{f_{rate}} - b_{i-1}, \quad L_b^0 = \frac{R}{f_{rate}} \tag{2.36}$$

where $s_{i+1} - s_i = \frac{1}{f_{rate}}, \forall i$ since encoding is performed at the frame rate f_{rate}. Finally, the estimated target bits T^i for the ith picture, which was mentioned in subsection 2.2.2, is bounded by

$$T_{HRD}^i = \max\left(\min(T^i, \tilde{U}_b^i), L_b^i\right), \quad \tilde{U}_b^i = \omega_{HRD} U_b^i \tag{2.37}$$

where $\omega_{HRD} = 0.9$ is multiplied to compensate for uncertainty of the coded bits X^i in [9]. More detailed information is discussed in [24, 31–34].

2.2.4 Lagrange Multiplier Selection

In subsection 2.2.1, QP was estimated to satisfy a bit constraint. In this subsection, we will derive the Lagrange multiplier λ for the mode decision procedure for a given QP which is marked as the yellow area in Figure 2.4. The authors in [20] first showed the relation between λ and QP as follows:

$$X = a \log_2\left(\frac{b}{D}\right), \quad D = \frac{(Q_{step})^2}{12} \tag{2.38}$$

$$\lambda = -\frac{dD}{dX} = \frac{\log_e 2}{12a} Q_{step}^2 \approx 0.85 QP_{H.263}^2 \tag{2.39}$$

where a and b are model parameters, and X is the number of coded bits, and D is quantization distortion, and $Q_{step} = 2QP_{H.263}$ in H.263 [5]. The first term in (2.38) is derived from the Independent and Identically Distributed (IID) gaussian process with variance σ^2 and entropy coding: $D = \epsilon^2 \sigma^2 2^{-2P}$ where $\epsilon^2 = 1.2$ for the Laplace distribution of prediction errors and P is bits per pixel in [35]. Note that coded bits for a picture X are induced from multiplying the horizontal resolution H_r and the vertical resolution V_r with the bits per pixel P, i.e. $X = H_r \times V_r \times P$. The second term in (2.38) is uniform quantization distortion [35]. The first relation in (2.39) is derived in order to minimize the Lagrangian cost $L = D + \lambda X$ w.r.t. X, that

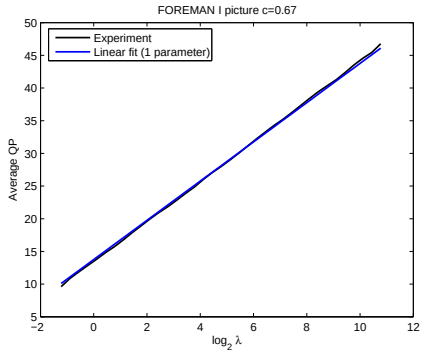
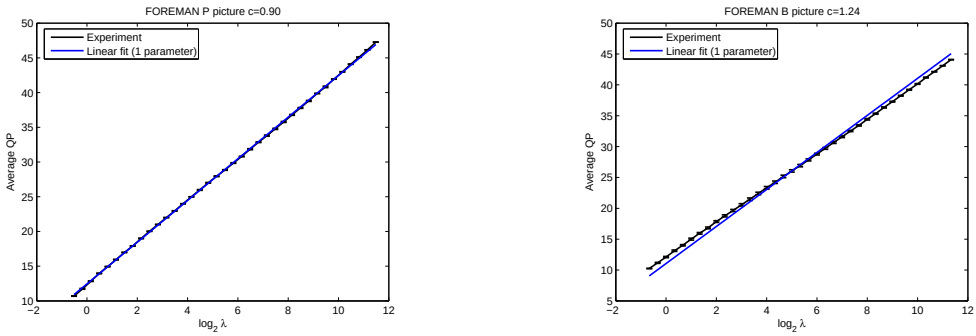


Figure 2.7: Relation between λ and QP of I pictures in the FOREMAN sequence.



(a) P picture (b) B picture

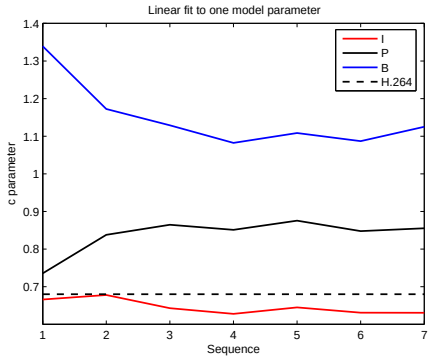
Figure 2.8: Relation between λ and QP of P and B pictures in the FOREMAN sequence.

is, $\frac{dL}{dX} = 0$. The Lagrange multiplier λ was determined by experimentally deciding $\frac{\log_e 2}{3a}$ as 0.85 in [20]. This value was further refined in [21] as 0.68. Consequently, the JM model [9] has the following relation:

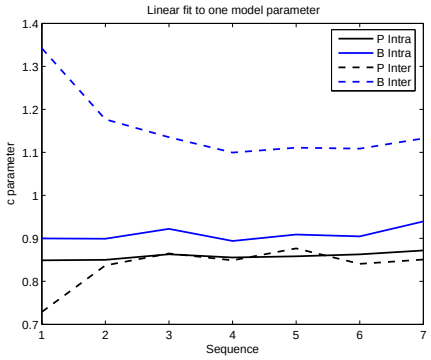
$$\lambda = c \cdot 2^{\frac{QP-12}{3}}, \quad c = \begin{cases} 0.68 & \text{if the number of B pictures } \ N_b > 0 \\ 0.85 & \text{otherwise} \end{cases} \tag{2.40}$$

where the relation between Q_{step} and QP is derived from (2.6) for H.264.

We performed several experiments to confirm the model parameter c in (2.40). A GOP has the IBBPBBPBBPBBI structure and QP varies $\pm$ 2 from the reference QP which corresponds to a given λ in (2.40). We use seven QCIF sequences such as SILENT, CARPHONE, FOREMAN, CREW, MOBILE, FOOT-



(a) Total model parameters c.



(b) Intra and inter c parameters.

Figure 2.9: c parameters of total, intra and inter prediction in I, P and B pictures according to the various sequences.

BALL and SOCCER which have from low-motion to high-motion characteristics. First, c in (2.40) is estimated from the linear relation between $\log_2 \lambda$ and QP. The model parameter c in the JM model does not depend on picture types such as I, P and B pictures. However, different c values are obtained from the experimental results according to different picture types as shown in Figures 2.7 and 2.8 where the model parameters of I, P and B pictures are 0.67, 0.84 and 1.15, respectively. We also consider variations of λ and QP relation at different positions within a GOP. For example, there are three P pictures within a GOP in these experiments and the relation between λ and QP is separately derived at each P picture. They are represented as error bars in Figure 2.8 (a), and the same experiment is applied to B pictures and the result is shown in Figure 2.8 (b). Overall variations are small as shown in Figure 2.8. Thus, positional variations of the model parameter c can be ignored to the same picture type within a GOP.

Figure 2.9 shows the model parameters c for different picture types from SILENT to SOCCER which are numbered from 1 to 7. The model parameters c of I pictures are approximately the same for the various sequences. The average c value of all the sequences is approximately 0.65 which is close to the result in (2.40). 0.84 and 1.15 are good c parameters for P and B pictures if the first two low-motion sequences are excluded in Figure 2.9 (a). Here, we can further decompose

Table 2.1: Model parameter c in (2.40)

JM	Proposed						
	I	P			B		
c_{ref}	c_I	Total c_P^t	Intra c_P^a	Inter c_P^r	Total c_B^t	Intra c_B^a	Inter c_B^r
0.68	0.65	0.84	0.86	0.84	1.15	0.91	1.16

the model parameters for intra and inter MBs in P and B pictures. Figure 2.9 (b) shows that the model parameter c of intra modes in P and B pictures varies less than for inter modes. 0.86 and 0.91 are average parameters of intra modes in P and B pictures for all the test sequences. 0.84 and 1.16 are experimental parameters for inter modes in P and B pictures, respectively. Table 2.1 summarizes overall results of the model parameter c.

We encode the video sequences using the different model parameters c for different picture types as listed in Table 2.1. Experimental results are listed in Tables 2.2 and 2.3. The average and maximum PSNR gains from $QP = 10$ to $QP = 40$ are calculated from a fitted PSNR to the experimental PSNR curve using the 7-degree polynomial [36]. The performance of I pictures in the JM model and experimental results are almost the same as shown in Table 2.2. However, we can see the performance improvement of P pictures with different λ for intra and inter MBs at high QP in Table 2.3. Note that we intentionally use the JM model parameter for I pictures (0.68 instead of 0.65) to remove the effects of temporal prediction dependency to P pictures. Consequently, the performance of the single model parameter for all the picture types in (2.40) can be improved by the different model parameters for different picture types.

In the previous paragraph, all the MBs were assumed to be temporally and spatially independent. Practically, we use inter and intra prediction to remove temporal and spatial redundancies in H.264 [1]. In section 2.3, we will prove that all the λ should be equal for the optimal bit allocation with the independent assumption and λ among the I, P and B pictures should have the following relation

Table 2.2: Average PSNR(dB) gain of I pictures ($c_I = 0.68$, $c_I^t = 0.65$)

Sequence	Max. gain PSNR	Avg. gain PSNR	QP			
			39	30	20	10
SILENT	0.02	0.00	0.02	0.00	0.00	0.00
CARPHONE	0.00	0.00	0.00	0.00	0.00	0.00
FOREMAN	0.01	0.00	-0.01	0.00	0.00	0.00
CREW	0.01	0.00	0.00	0.00	0.00	0.00
MOBILE	0.02	0.00	0.01	0.01	0.00	0.00
FOOTBALL	0.01	0.00	0.00	0.00	0.00	0.00
SOCCER	0.02	0.00	0.00	0.00	-0.01	0.01

Table 2.3: Average PSNR(dB) gain of P pictures with the different model parameters for intra and inter MBs ($c_I = 0.68$, $c_P^a = 0.86$, $c_P^r = 0.84$)

Sequence	Max. gain PSNR	Avg. gain PSNR	QP			
			40	30	20	10
SILENT	0.08	0.04	0.08	0.05	0.04	0.03
CARPHONE	0.07	0.03	0.07	0.05	0.02	0.01
FOREMAN	0.07	0.03	0.07	0.04	0.03	0.00
CREW	0.02	0.01	0.01	0.02	0.00	0.00
MOBILE	0.05	0.02	0.02	0.04	0.01	0.01
FOOTBALL	0.03	0.02	0.00	0.02	0.02	0.01
SOCCER	0.04	0.03	0.00	0.02	0.04	0.03

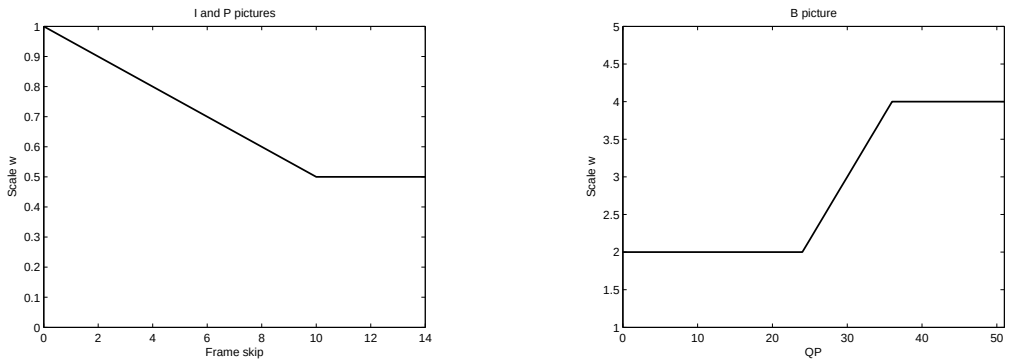
with temporal prediction dependency:

$$\lambda_I \leq \lambda_P \leq \lambda_B \tag{2.41}$$

where λ_I, λ_P and λ_B are λ of I, P and B pictures, respectively. However, we only consider temporal dependency among the MBs with spatial independence for simplicity. With spatial independence, a single λ is used to encode all the MBs for the optimal bit allocation within a picture. This condition is satisfied if the same QP is applied to all the MBs in (2.40). Temporal dependency can be considered through multiplying a scale factor w to (2.40) in order to satisfy the condition (2.41) as follows:

$$\lambda = w \cdot c \cdot 2^{\frac{QP-12}{3}} \tag{2.42}$$

The JM model applies different scale factors w for I, P and B pictures as shown in Figure 2.10. The scale factor w for I and P pictures decreases according to the



(a) w for I and P pictures.

(b) w for B pictures.

Figure 2.10: w scale factors for I, P and B pictures with temporal prediction dependency.

number of the frame skip (the number of frames between I or P and P pictures) which is shown in Figure 2.10 (a). The scale factor w for B pictures increases with respect to QP in Figure 2.10 (b). These scaling factors correspond to the results in section 2.3: Quality of reference pictures should be increased according to the number of pictures which use the reference pictures for inter prediction and quality of reference pictures is more important at high QP. These phenomena will be discussed in section 2.3.

2.3 Proposed Rate-Distortion Optimization

In the discussion regarding equations (2.6) and (2.7), the JM model mainly focuses on the real-time and low-complexity RC algorithm with the constant or variable bit rate constraint. As a result, the RC algorithm cannot tightly satisfy the bit constraint with loss of coding efficiency which will be shown in section 2.4. In the case of nonreal-time applications with the constant bit rate constraint, the loss can be reduced. In this section, we will apply the primal-dual decomposition and subgradient projection methods to solve the RD problem (2.1) with the constant GOP bit constraint. Although this method can be used for the optimal bit allocation of any basic unit with consideration of spatial and temporal prediction

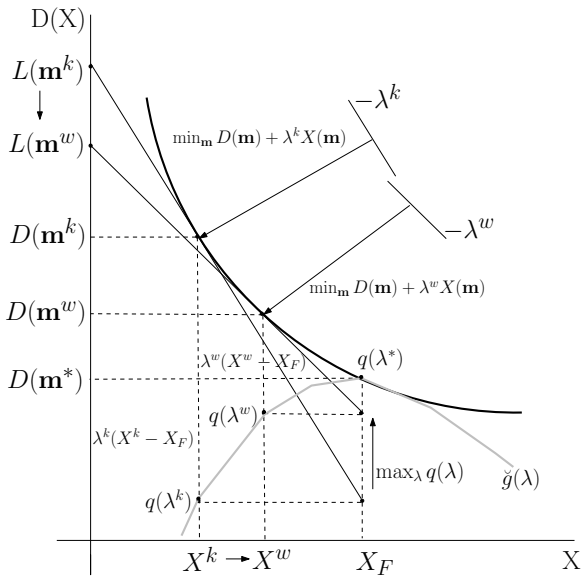


Figure 2.11: Geometric interpretation of an iterative RD optimization method.

dependency, we will only show the frame-level bit allocation within a GOP with considering temporal prediction dependency as an example. Thus, we ignore spatial prediction dependency so that all the MBs which have intra and spatial direct modes are independent.

2.3.1 Iterative Rate-Distortion Optimization

In this subsection, we solve the problem (2.1) for a given frame bit constraint X_F using an iterative RD optimization method. The procedure will be explained with geometric interpretation which is shown in Figure 2.11. Here, we assume that the RD function is smooth and continuous for simplicity. In Figure 2.11, let $\sum_{n=1}^{N} d_n(\mathbf{m}_n)$, $\sum_{n=1}^{N} x_n(\mathbf{m}_n)$ and $\sum_{n=1}^{N} l_n(\mathbf{m}_n)$ in (2.1) be $D(\mathbf{m})$, $X(\mathbf{m})$ and $L(\mathbf{m})$ which represent distortion, coded bits and the Lagrangian cost of a frame, respectively. The parameters k and w are iteration indices, and X^k is $X(\mathbf{m}^k)$ where $\mathbf{m}^k$ is the solution of (2.2) at the kth iteration. For a given λ^k, the JM model solves the problem (2.2) that is equivalent to finding $\mathbf{m}^k$ associated with the point $(X(\mathbf{m}^k),\ D(\mathbf{m}^k))$ at which λ^k is tangent to the RD function [11]. It

induces the minimum of $L(\mathbf{m})$ for a given λ^k. If the bit constraint X_F is equal to the number of coded bits $X(\mathbf{m}^k)$, $\mathbf{m}^k$ and λ^k are the optimal solutions of the primal problem (2.1) and its dual problem (2.3), respectively. Thus, the RD optimization without a bit constraint can always achieve the optimal solutions for a given λ with assumption that coded bits are constraint bits. If there is a bit constraint and $X_F \neq X(\mathbf{m}^k)$, $\mathbf{m}^k$ and λ^k are not the optimal solutions of the primal problem (2.1) and its dual problem (2.3). In this case, we can use iterative methods to find the optimal solutions. In Figure 2.11, the dual function value $q(\lambda^k)$ of (2.2) which is marked on the line of X^k is $D(\mathbf{m}^k) + \lambda^k(X^k - X_F)$. If we mathematically know the dual function, it is easy to find the solution of the dual problem (2.3) since its constraint is simple. The JM model can find the optimal primal solution from the optimal dual solution λ^*. However, it is difficult to find the dual function. Therefore, we try a different λ^w such that a value of the dual function increases. Figure 2.11 shows that if λ^w is slightly smaller than λ^k, $q(\lambda^w)$ is larger than $q(\lambda^k)$ which means that λ^w is closer than λ^k to the optimal solution of the dual problem (2.3). Furthermore, λ^w reduces the Lagrangian cost from $L(\mathbf{m}^k)$ to $L(\mathbf{m}^w)$.

In the next subsection, we will discuss how to decide the next iteration λ^w from λ^k. If we find λ^* which maximizes the dual function after several iterations, the optimal primal solution $\mathbf{m}^*$ can be found for a given λ^* from the mode decision procedure. Consequently, $q(\lambda^*)$ is $D(\mathbf{m}^*)$ since $X(\mathbf{m}^*)$ is X_F. It is a well known result from the KKT conditions [6, 14]. Practically, we need to find the operational RD function which consists of convex-hull points since the real RD function is neither a convex function nor a continuous function. Furthermore, there might be no feasible solution to satisfy the bit constraint X_F since $\mathbf{m}$ is a discrete feasible set. Therefore, we allow constraint violation within some range and find the best solution around the bit constraint. After several iterations, we can linearly approximate the dual function $q(\lambda)$ as $\breve{q}(\lambda)$ which is illustrated in Figure 2.11. If λ^* is obtained, QP can be derived from (2.7) or QP can be an optimization variable in the problem (2.2). If QP is considered as an optimization variable, various QP from a center QP are applied for quantization in the mode decision procedure where the center QP is decided from (2.7) [20, 21].

2.3.2 Primal-Dual Decomposition and Subgradient Projection Methods

We will introduce a general framework to solve the RD optimization problem using the primal-dual decomposition and subgradient projection methods [6–8]. The primal decomposition corresponds to deciding the optimal bit constraint of a basic unit which was discussed in subsection 2.2.2. Dual decomposition and the Lagrangian duality, as explained in the previous subsection, are equivalent to obtaining the optimal primal and dual solutions for a given optimal bit constraint as a result of the primal decomposition. For convenience, we simplify the notation of problem (2.1) as follows:

$$\min_{\mathbf{m}} \quad \sum_n d(\mathbf{m}_n), \quad s.t. \quad \sum_n x_n(\mathbf{m}_n) \leq X \tag{2.43}$$

$$\min_{\mathbf{y}} \min_{\mathbf{m}} \quad \sum_n d(\mathbf{m}_n), \quad s.t. \quad x_n(\mathbf{m}_n) \leq y_n, \quad \sum_n y_n \leq X, \quad \forall n, \tag{2.44}$$

$$\min_{\mathbf{m}} \quad \sum_n d(\mathbf{m}_n), \quad s.t. \quad x_n(\mathbf{m}_n) \leq y_n, \quad \forall n \tag{2.45}$$

$$\min_{\mathbf{y}} \quad q^*(\mathbf{y}), \qquad s.t. \quad \sum_n y_n \leq X \tag{2.46}$$

where $q^*(\mathbf{y}) = \min_{\mathbf{m}} \sum_n d(\mathbf{m}_n) + \lambda_n^* \{x_n(\mathbf{m_n}) - y_n\}$ which is the optimal value of (2.45). The original problem (2.43) can be reformulated into the problem (2.44) by introducing slack variables $\mathbf{y}$. Then, the problem (2.44) can be decomposed into the two optimization problems (2.45) and (2.46) according to the optimization variables $\mathbf{m}$ and $\mathbf{y}$. The decomposition from (2.43) to (2.46) is called the master primal decomposition, and the decomposition from (2.45) to (2.48) is known as the dual decomposition through the Lagrange duality. The problem (2.45) is solved

by the Lagrangian duality as follows:

$$q(\mathbf{y},\lambda) = \min_{\mathbf{m}} \sum_n d(\mathbf{m}_n) + \lambda_n\{x_n(\mathbf{m}_n) - y_n\} \tag{2.47}$$

$$= \sum_n \min_{\mathbf{m}_n} d(\mathbf{m}_n) + \lambda_n\{x_n(\mathbf{m}_n) - y_n\} \tag{2.48}$$

$$q^*(\mathbf{y}) = \max_{\lambda \succeq 0} q(\mathbf{y},\lambda) \tag{2.49}$$

$$= \max_{\lambda \succeq 0} \sum_n q_n(y_n,\lambda_n) = \sum_n \max_{\lambda_n \geq 0} q_n(y_n,\lambda_n) \tag{2.50}$$

$$= \sum_n q_n^*(y_n) \tag{2.51}$$

where $q_n(y_n,\lambda_n) = \min_{\mathbf{m}_n} d(\mathbf{m}_n) + \lambda_n\{x_n(\mathbf{m}_n) - y_n\}$. Equations (2.48) and (2.50) are derived from the independence assumption. The problem (2.48) is solved by the mode decision which is implemented in the JM model [9] and the dual problem (2.49) can be solved by the subgradient projection method [6] as follows:

$$\lambda_n^{k+1} = \left[\lambda_n^k + \eta^k \mathrm{g}_n^k\right]^+ = \max(\lambda_n^k + \eta^k \mathrm{g}_n^k, 0) \tag{2.52}$$

where η^k is a positive step size at the kth iteration and $[\cdot]^+$ denotes the projection onto the nonnegative orthant. The projection operation guarantees that the Lagrange multipliers λ_n satisfy their nonnegative conditions. The subgradient g_n^k of $q_n(\lambda_n^k, y_n)$ is $x_n^k - y_n$ which is derived in Appendix A.1.1. Therefore, the subgradient of $q_n(\lambda_n^k, y_n)$ is just the difference between coded bits and the constraint bits at the kth iteration. If we substitute g_n^k with $x_n^k - y_n$ in (2.52), it becomes

$$\lambda_n^{k+1} = \max(\lambda_n^k + \eta^k(x_n^k - y_n), 0) \tag{2.53}$$

where x_n^k are coded bits for a given λ_n^k. Equation (2.53) indicates that if coded bits x_n^k are smaller than the constraint bits y_n, the current Lagrangian multiplier λ_n^k decreases, otherwise, λ_n^k increases. From the RD optimization, smaller λ increases coded bits. Therefore, coded bits are getting close to the constraint bits after several iterations. The reason why we use the subgradient instead of gradient is that the RD function is not known exactly and even the operational RD function is a convex-hull of the RD function which is not a smooth function. Consequently,

the step size η^k is carefully selected since the subgradient direction is not always in increasing direction for any step size [6].

The master primal problem (2.46) is also solved by the subgradient projection method. However, the constraint is not as simple as in (2.52). The solution of the problem (2.46) is obtained from two procedures. First, the optimization variables y_n are updated by the subgradient as follows:

$$\tilde{y}_n^{k+1} = y_n^k - \eta^k g_n^k \tag{2.54}$$

and then $\tilde{\mathbf{y}}$ is projected onto the feasible constraint set as:

$$\min_{\mathbf{y}} \| \tilde{\mathbf{y}} - \mathbf{y} \|_2^2, \quad s.t. \quad \sum_n y_n \leq X, \quad y_n \geq 0 \tag{2.55}$$

which is formulated from the fact that the projected point $\mathbf{y}$ from $\tilde{\mathbf{y}}$ minimizes the distance between two points. This problem can be solved by using the very efficient algorithm discussed in Appendix A.2. The subgradient g_n of $q_n^*(y_n)$ is shown in Appendix A.1.2. As a result, the subgradient g_n^k of $q_n^*(y_n)$ at y_n^k is $-\lambda_n^k$ where λ_n^k is the optimal dual variable of the subproblem in (2.45), that is, the convergent solution of (2.53) for a given y_n^k after iterations. From Appendix A.1.2, the subgradient is the lower bound of sensitivity of the optimal dual function w.r.t. allocated bits y:

$$0 \geq \frac{q^*(\tilde{y}) - q^*(\hat{y})}{\tilde{y} - \hat{y}} \geq -\hat{\lambda}$$

and $\tilde{y}_n^{k+1} = y_n^k + \eta^k \lambda_n^k$ from (2.54). Consequently, more bits are allocated to MBs or pictures (basic units) which have larger λ since larger λ implies that distortion of a MB or a picture decreases further according to the unit bit increment of the constraint bit. Figure 2.12 shows the geometric interpretation of subgradient of $q^*(y)$. The optimal value $q^*(y)$ can be rewritten as follows for a given optimal dual variable $\hat{\lambda}^*$ (for simplicity, the index n is omitted):

$$q^*(\hat{y}, \hat{\lambda}^*) = \min_{\mathbf{m}} d(\mathbf{m}) + \hat{\lambda}^* \{x(\mathbf{m}) - \hat{y}\} \tag{2.56}$$

Reference [14] indicates that the optimal coded bits $x^* = x(\mathbf{m}^*)$ of the problem (2.56) satisfies the complementary slackness condition, that is, $\hat{\lambda}^*(x^* - \hat{y}) = 0$. For

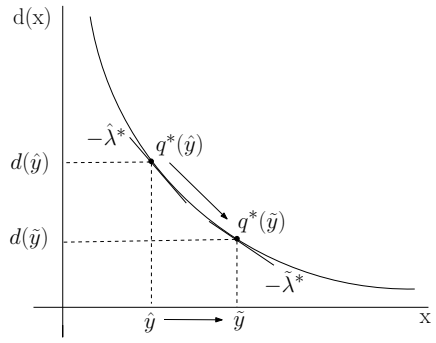


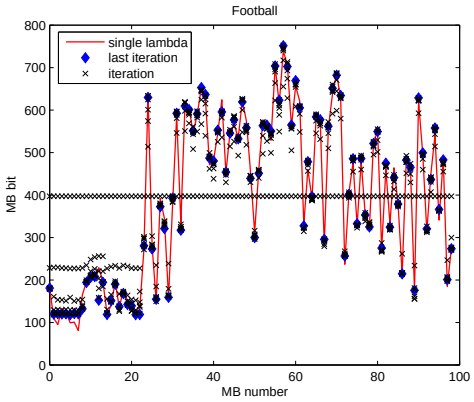
Figure 2.12: Geometric interpretation of subgradient of $q^*(y)$.

$\hat{\lambda}^* > 0$, $x^* = \hat{y}$. Consequently, $q^*(\hat{y}, \hat{\lambda}^*) = d(x^*) = d(\hat{y})$. If we allocate more bits from $\hat{y}$ to $\tilde{y}$, x^* changes from $\hat{y}$ to $\tilde{y}$. If we set $\tilde{y} = \hat{y} + \varepsilon$, the sensitivity of the optimal value $q^*(y)$ is

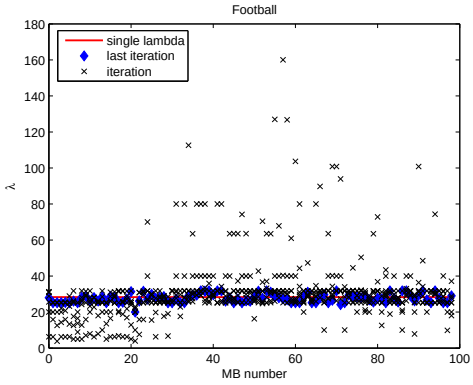
$$\lim_{\varepsilon \to 0} \frac{q^*(\tilde{y}) - q^*(\hat{y})}{\tilde{y} - \hat{y}} = \lim_{\varepsilon \to 0} \frac{d(\hat{y} + \varepsilon) - d(\hat{y})}{\varepsilon} = -\hat{\lambda}^* \tag{2.57}$$

Thus, λ indicates that MBs or pictures which have larger λ can reduce more distortion. If we reallocate constraint bits, the sum of distortion can be decreased. Furthermore, all the subgradients of MBs or pictures should be equal for the optimal bit allocation. Because sensitivity of the optimal values of all the MBs or pictures are equal, there is no way to reallocate constraint bits to decrease the sum of distortion. This can be clearly observed from (2.54) since $\tilde{y}_n^{k+1}$ increase equally if their subgradient g_n^k and step size η^k are equal, and then $\tilde{y}_n^{k+1}$ are projected onto the feasible set shown in (2.55). The projected y_n^{k+1} are the same as y_n^k which results from Appendix A.2.

Here, we show experimental results of the primal-dual decomposition and subgradient projection methods with spatially independent assumption for the MB-level bit allocation within an intra picture of the QCIF size. Without a bit rate constraint, λ is decided from (2.7) for a given QP and then the JM model of H.264 solves the problem (2.5) for all the MBs in a frame with independent assumption among the MBs. In this case, the JM model solves the problem of (2.43) using a single Lagrange multiplier for all the MBs. With independent assumption, it is optimal since sensitivity of all the MB (λ) is equal and λ is the optimal dual

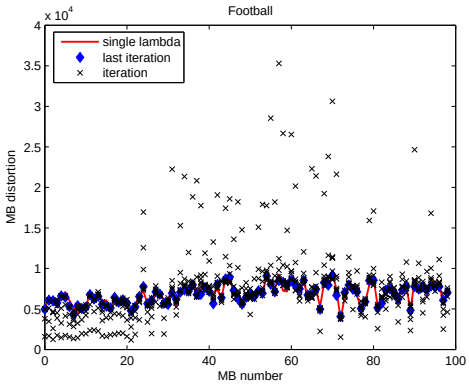


(a) MB bits

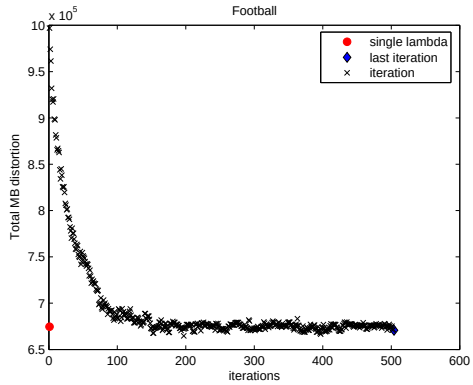


(b) λ

Figure 2.13: MB bits and λ of single λ vs. multiple λ_n.



(a) MB distortion



(b) Frame distortion

Figure 2.14: MB and frame distortion of single λ vs. multiple λ_n.

solution by assuming that coded bits are equal to constraint bits as explained in section 2.3.1.

Now, we solve the same problem using the primal-dual decomposition and subgradient methods with the bit constraint. The bit constraint is given from coded bits of the JM model after solving the problem (2.4) (encoding a frame). Thus, we solve the same problem with different methods. Consequently, each MB has its own Lagrange multiplier λ_n from the problems (2.45) and (2.46). Figure 2.13 (a) shows that total bits are equally divided into each MB at the initial iteration. As a result, every MB has very different λ_n and distortion which are shown in Figures 2.13 (b) and 2.14 (a). At the next iteration, more bits are allocated to MBs which have larger λ_n and on the other hand, fewer bits are allocated to MBs which have smaller λ_n shown in Figures 2.13 (a) and (b). At the last iteration which is marked as a diamond symbol in Figures 2.13 and 2.14, MB bits, λ_n and distortion are almost equal between a single λ and multiple λ_n since we solve the same problem using different methods. The variations of MB bits of the MB number from 0 to 7 and from 57 to 60 are almost equal from the initial to the last iteration (almost 300 bits are different) in Figure 2.13 (a). However, Figure 2.13 (b) shows that the decrease of distortion of MBs from 57 to 60 is much larger than the increase of distortion of MBs from 0 to 7 because MBs which have larger λ can reduce their distortion more efficiently. Consequently, the sum of MB distortion decreases after iterations which is shown in Figure 2.14 (b), and overall distortion of the primal-dual decomposition method is almost equal to distortion of a single λ.

2.3.3 Rate-Distortion Optimization with Temporal Prediction Dependency

In the previous subsection, we assumed that all the distortion and coded bit functions are independent. In this subsection, the independence assumption among the basic units is removed. Even though we only consider the RD optimization problem with temporal prediction dependency among the pictures within a

GOP, there is no restriction in applying the RD optimization with spatial prediction dependency. However, we still assume that all the MBs which have spatial prediction dependency are independent for simplicity, but temporal coding dependency is considered among the basic units (frames). With this assumption, a single λ for all the MBs within a frame is optimal which was explained in the previous subsection.

The equation (2.43) is reformulated for the frame-level RD optimization with a GOP bit constraint as follows:

$$\min_{\mathbf{s}} \sum_{f=1}^{F} D_f(\mathbf{s_f}, \mathbf{x_{ref}^f}) \quad s.t. \quad \sum_{f=1}^{F} X_f(\mathbf{s_f}, \mathbf{x_{ref}^f}) \leq X_{GOP} \tag{2.58}$$

where

$$D_f(\mathbf{s_f}, \mathbf{x_{ref}^f}) = \sum_{n=1}^{N} d_n(\mathbf{m}_n), \quad X_f(\mathbf{s_f}, \mathbf{x_{ref}^f}) = \sum_{n=1}^{N} x_n(\mathbf{m}_n)$$

and $\mathbf{s_f} = (\mathbf{M_f}, \mathbf{MV_f}, \mathbf{QP_f})$. $\mathbf{M_f}, \mathbf{MV_f}, \mathbf{QP_f}$ and $\mathbf{x_{ref}^f}$ are MB modes, MVs, QP and coded bits of reference frames of all the MBs in the fth frame, respectively. X_{GOP} is a GOP bit constraint and F is the number of frames within a GOP. The fth frame distortion D_f and bits X_f depend on all the MB modes, MVs and QP as well as coded bits of reference frames. Thus, every frame cannot be optimized independently in the problem (2.58) because of dependency on code bits of reference frames $\mathbf{x_{ref}^f}$.

Figure 2.15 illustrates mapping between a GOP structure and the primal-dual decomposition. As a specific example, we only consider the first GOP structure which starts with the Instantaneous Decoder Refresh (IDR) frame (the first B frames which are denoted as B_0 are included from the second GOP) and the close GOP, that is, the last P frame within a GOP is not used to predict B_1. However, the open GOP and any prediction dependency among the B and P frames within a GOP are not limited to the proposed method. In Figure 2.15, all the B frames within a GOP are predicted from the same reference frames I and P, and the P frame is predicted from the I frame. When we perform the master primal decomposition, dependency among the frames is considered. As in section 2.3.2, slack

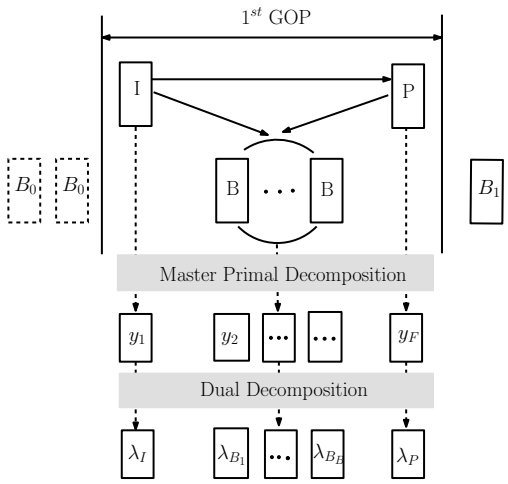


Figure 2.15: Mapping between GOP and the primal-dual decomposition.

variables y_f are introduced for frame bit constraints. Consequently, the problem (2.58) is decomposed into one master primal problem (2.60) and F subproblems (2.59) through the Lagrangian duality:

$$\begin{aligned} \min_{\mathbf{s}} \quad & D_f(\mathbf{s_f}, \mathbf{y^f_{ref}}) \\ s.t. \quad & X_f(\mathbf{s_f}, \mathbf{y^f_{ref}}) \leq y_f, \quad f \in \{1, ..., F\} \end{aligned} \tag{2.59}$$

$$\begin{aligned} \min_{\mathbf{y}} \quad & \sum_f Q^*_f(y_f, \mathbf{y^f_{ref}}), \\ s.t. \quad & \sum_f y_f \leq X_{GOP}, \quad f \in \{1, ..., F\} \end{aligned} \tag{2.60}$$

where $Q^*_f(y_f, \mathbf{y^f_{ref}})$ are the optimal values of subproblems (2.59) for a given $\mathbf{y}$ and the reference frame bits are $\mathbf{y^w_{ref}} = (y_1, y_F)$ where $w \in \{2, .., F-1\}$ for the B frames, $\mathbf{y^F_{ref}} = y_1$ for the P frame and $\mathbf{y^1_{ref}} = \varnothing$ since the I frame has no reference frames. Comparing (2.58) with (2.59), we observe the main benefit from the primal decomposition. In (2.58), the reference frame bits $\mathbf{x^f_{ref}}$ prevents independent optimization. However, $\mathbf{x^f_{ref}} = \mathbf{y^f_{ref}}$ in (2.59) because $X_f(\mathbf{s_f}^*) = y_f$ from the complementary slackness condition [14]. Independent optimization is clearer from the

following equations:

$$\min_{\mathbf{s}} \sum_{f} D_f(\mathbf{s_f}, \mathbf{y^f_{ref}}) - \lambda_f \Big(X_f(\mathbf{s_f}, \mathbf{y^f_{ref}}) - y_f \Big)$$
$$= \sum_{f} \min_{\mathbf{s_f}} D_f(\mathbf{s_f}, \mathbf{y^f_{ref}}) - \lambda_f \Big(X_f(\mathbf{s_f}, \mathbf{y^f_{ref}}) - y_f \Big)$$

where $\mathbf{y}$ and $\mathbf{y^f_{ref}}$ are constants. Therefore, we can reuse the same reference software model of H.264 [9] to minimize the Lagrangian cost in the problem (2.59) with consideration of dependency. Reference [17] also solved the frame-level dependent coding problem using the Viterbi algorithm, but it considered that distortion and coded bits are only functions of QP which makes the problem be tractable. If distortion and coded bits are functions of QP as well as MB modes, MVs and λ, the number of states in the Viterbi algorithm is too large to solve the problem. Here, we do not separately consider effects of all the parameters but rather we are interested in allocating constraint bits among the frames. Given a bit constraint, all the parameters are only optimization variables to satisfy the bit constraint. Especially, temporal dependency among the frames only depends on prediction quality. Thus, dependency among the frames is processed in the master primal problem as shown in (2.60) which is a much simpler optimization problem.

In order to solve the problem in (2.59), we use the Lagrangian duality and subgradient projection methods which were explained in section 2.3.2. Therefore, we only discuss the master primal problem (2.60) in this subsection. From Appendix A.1.3, the subgradients of $\sum_{f=1}^{F} Q^*_f(y_f, y^f_{ref})$ w.r.t. B, P and I pictures y_f at $\hat{y}_f$ are

$$-\hat{\lambda}_k, \quad -\left(\hat{\lambda}_F - \sum_{f=2}^{F-1} \min_{\mathbf{s_f}} L'_f(\mathbf{s_f}, \hat{y}_F)\right) \text{ and } \quad -\left(\hat{\lambda}_1 - \sum_{f=2}^{F} \min_{\mathbf{s_f}} L'_f(\mathbf{s_f}, \hat{y}_1)\right)$$

where $k \in \{2, ..., F-1\}$ and $L'_f(\mathbf{s_f}, \hat{y}^f_{ref}) = \left.\frac{\partial L_f(\mathbf{s_f}, y^f_{ref})}{\partial y^f_{ref}}\right|_{y^f_{ref_i} = \hat{y}^f_{ref}}$, $\hat{y}^f_{ref} \in \{\hat{y}_F, \hat{y}_1\}$ which represents that variations of the Lagrangian cost at the fth frame w.r.t. bit variations of its reference frames. Thus, $L'_f(\mathbf{s_f}, \hat{y}^f_{ref})$ is generally negative because increasing of reference frame bits induces decreasing of the Lagrangian cost at the fth frame. As a result, the subgradients of the referenced frames which are used

as reference frames for inter prediction are smaller than independent frames for $\hat{\lambda}_k = \hat{\lambda}_1 = \hat{\lambda}_F$. It means that more bits are allocated to referenced frames from the equation (2.54). This result matches with intuition: quality of referenced frames is more important than quality of nonreferenced frames because they are used for inter prediction. As explained in section 2.3.2, the subgradients of all the frames are equal for the optimal bit allocation. Therefore, relation of λ among the pictures is derived as follows:

$$\lambda_I - \sum_{f=2}^{F} \min_{\mathbf{s_f}} L'_f(\mathbf{s_f}, y_I) = \lambda_P - \sum_{f=2}^{F-1} \min_{\mathbf{s_f}} L'_f(\mathbf{s_f}, y_P) = \lambda_B \tag{2.61}$$

where $\lambda_I = \lambda_1$, $\lambda_P = \lambda_F$ and $\lambda_B = \lambda_k, \quad k \in \{2, .., F-1\}$. Consequently, $\lambda_I \leq \lambda_P \leq \lambda_B$. This result explains the reason why the JM model applies different scale factors w for λ in (2.42) with different picture types as well as the number of prediction dependency. Furthermore, if B frames are used for prediction, λ of referenced B frames are different from λ of nonreferenced B frames. However, the JM model [9] applies the same scale w for I and P pictures.

A remaining problem is how to estimate the quantity $\min_{\mathbf{s_f}} L'_f(\mathbf{s_f}, \hat{y}^f_{ref})$. We experimentally estimate it as shown in Figures 2.16 and 2.17. Figure 2.16 shows the Lagrangian cost variations of three B pictures and one P picture according to the variations of bits of an I reference frame for a given $\hat{\lambda}$. Even though the Lagrangian cost does not monotonically decrease, it mainly exponentially decreases and the sum of all the variations is closer to exponential decrement which is shown in Figure 2.17. Exponential decrement explains that allocated bits to reference frames (quality of reference frames) are more important at low bit rates, that is, at large λ. As a result, $\sum_f \min_{\mathbf{s_f}} L'_f(\mathbf{s_f}, \hat{y}^f_{ref})$ is modeled as $-\exp^{(\alpha_1 QP + \alpha_2)}$ where α_1 and α_2 are constants.

2.4 Experimental Results

In this section, we compare the performance of two coding modes in the JM model [9] with the proposed method (H.264 encoder + primal-dual optimizer) which is illustrated in Figure 2.18. We set QP=35 in the JM model without RC

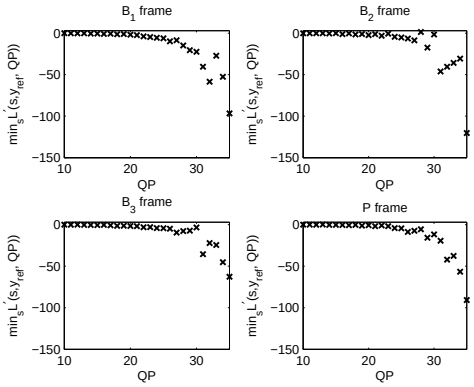


Figure 2.16: $\min_{\mathbf{s_f}} L'_f(\mathbf{s_f}, \hat{y}^f_{ref})$ of three B pictures and one P picture at $\hat{\lambda}(QP)$.

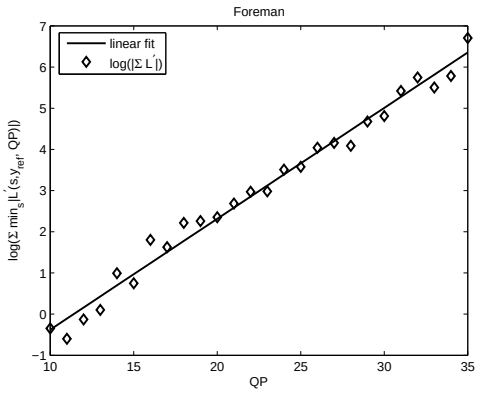


Figure 2.17: $\log \left| \sum_f \min_{\mathbf{s_f}} L'_f(\mathbf{s_f}, \hat{y}^f_{ref}) \right|$ and its linear fit at $\hat{\lambda}(QP)$.

and then the coded bits after encoding are used for constraint bits. The initial QP is 35 in the JM model with RC and the proposed encoder as shown in Figure 2.18. In the JM model with RC, the initial QP is used to encode I and the first P pictures in the first GOP so that the RC algorithm is only applied for B pictures which was discussed in subsection 2.2.1. In contrast, the proposed encoder ignores the initial QP.

2.4.1 GOP Bit Allocation with Independent Assumption

In this experiment, we assume that all the MBs are spatially and temporally independent. Therefore, a global λ for all the MBs within a GOP induces optimal

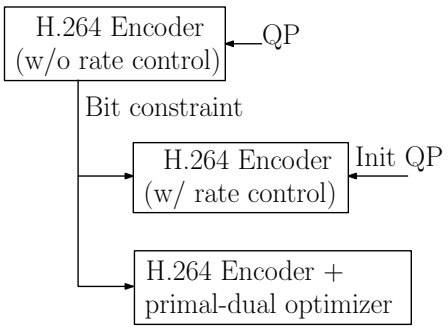
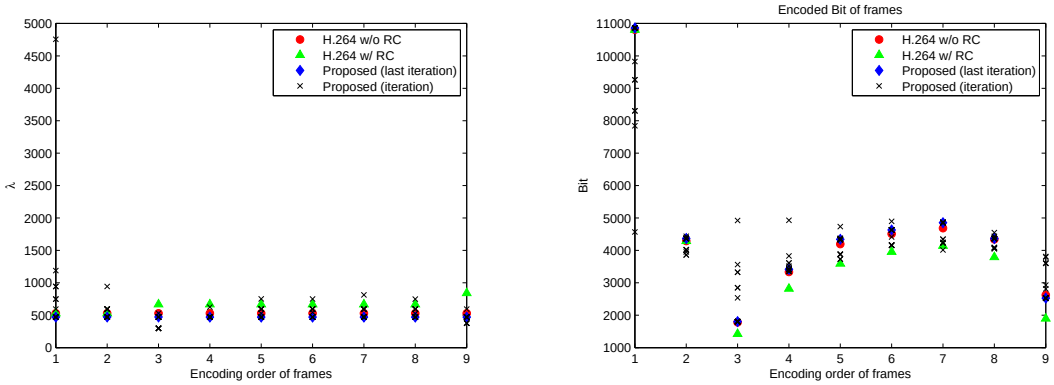


Figure 2.18: Experimental environment.

bit allocation since all the Lagrange multipliers λ are equal. In order to compare the performance, we set κ of (2.7) in the JM model to be equal for all the picture types with independent assumption. A GOP consists of one I and P pictures and seven B pictures in Figure 2.15. Figure 2.19 (a) shows that λ of all the frames are equal in the JM model without RC, and the proposed method also has equal λ after iterations, but the JM model with RC has different λ, especially at the last B frame. Figures 2.19 (b) and 2.20 illustrate coded bits of each frame and its Y-PSNR (dB), respectively. The JM model without RC and the proposed method show almost the same performance since they solve the same problem. However, the JM model with RC estimates encoded bits and QP to satisfy bit constraints and its dual variable λ is derived from (2.7) as explained in section 2.2. In order to compare exactly the performance, the proposed method does not perform QP optimization to minimize the Lagrangian cost. QP is derived from λ using (2.7). Thus, the proposed method uses the optimization variables (MVs, MB modes) which are the same in the JM model without RC. However, the proposed method derives the similar results of the JM model without RC with showing that its PSNR is higher than the JM model with RC in Figure 2.20.

2.4.2 GOP Bit Allocation with Temporal Dependency

In this experiment, we consider temporal prediction dependency among the frames within a GOP. Therefore, w of (2.42) in the JM model is applied



(a) λ of frames. (b) Coded bits of frames.

Figure 2.19: λ and coded bits of frames with independent assumption.

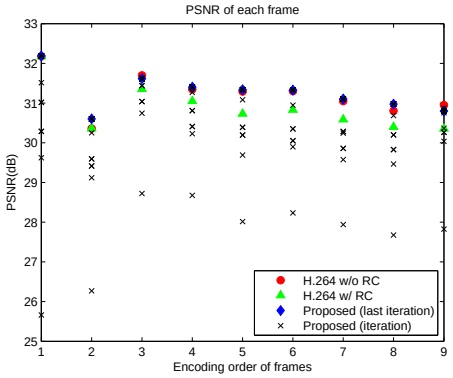
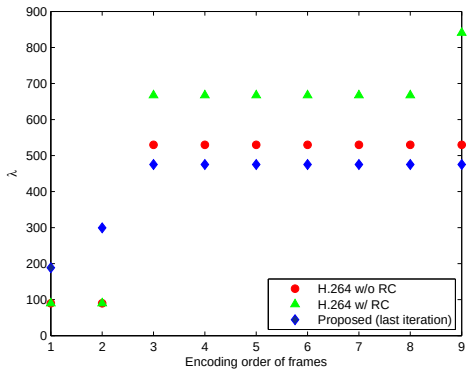
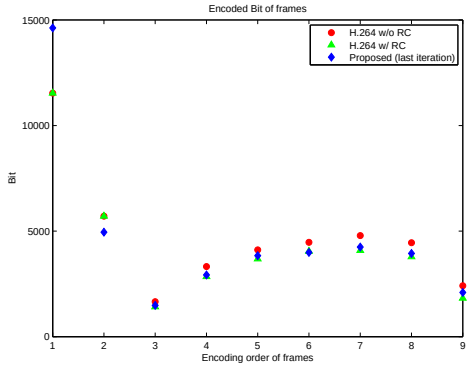


Figure 2.20: Y-PSNR(dB) of frames with independent assumption.

for different picture types [9]. The proposed method performs QP optimization within ± 1 at center QP which is derived from λ which gives more achievable bit regions. Here, we only show the results at the last iteration. Figure 2.21 (a) shows the different Lagrange multipliers λ among the I, P and B pictures. All the λ of B pictures are equal except the JM model with RC. Thus, bit allocation of the JM model with RC for B frames is not optimal since there is no prediction dependency among the B pictures as discussed in subsection 2.3.3. The proposed method applies different λ for I and P pictures as a result of (2.61), but the JM model without RC uses the same λ for I and P pictures. Figures 2.21 (b)

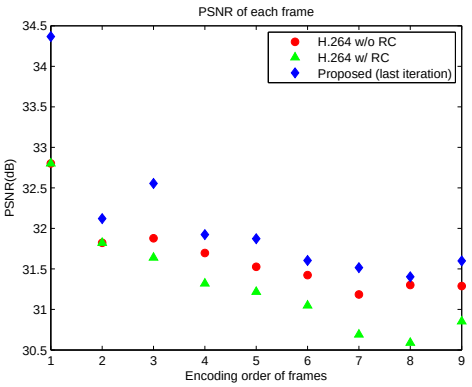


(a) λ of frames.

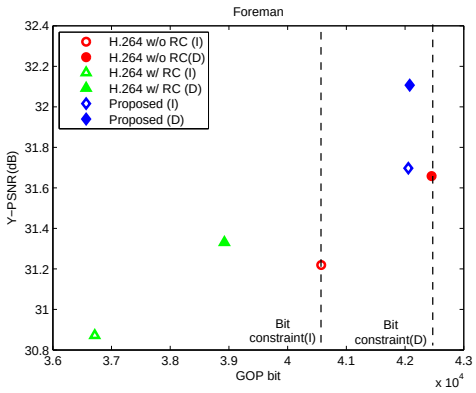


(b) Coded bits of frames.

Figure 2.21: λ and coded bits of frames with temporal dependency.



(a) PSNR of frames with dependency.



(b) Average PSNR vs. total coded bits.

Figure 2.22: PSNR (dB) of frames with dependent and independent cases.

and 2.22 (a) show that the proposed method uses similar total bits and achieves 0.8 dB higher PSNR than the JM model with RC. Figure 2.22 (b) illustrates overall encoded bits and average Y-PSNR (dB). Note that GOP bits of the JM model without RC are constraints to the JM model with RC and the proposed method. The constraint bits with temporal dependency (coded bits of the JM model without RC) is higher than ones with independent assumption because λ of I and P pictures become smaller. The RD optimization increases coded bits at smaller λ. In subsection 2.4.1, the proposed method allows constraint violations

within 5% of allocated bits for I picture and 2% for P and B pictures. In this subsection, 2%, 1% and 1% constraint violations are allowed for I, P and B pictures, respectively. Thus, the experiment with temporal dependency meets more tightly the bit constraint. Note that small bit constraint violations are allowed to consider convex-hull points around bit constraints. The JM model with RC does not satisfy the constraint with lower PSNR as shown in Figure 2.22 (b). These experiments are applied to various sequences which have different motion activity and different resolutions. From Tables 2.4 and 2.5, the proposed method achieves similar coded bits and PSNR of the JM model without RC, that is, the proposed method achieves the optimal results with the independence assumption. The JM model with RC shows larger loss at high motion sequences such as FOOTBALL and BUS and high resolution sequences of PARIS and TEMPETE. Thses phenomena are similar to the results with temporal dependency which are shown in Tables 2.6 and 2.7. The proposed method achieves higher PSNR within the bit violation constraint. Another different GOP structure which consists of one I and fifteen P pictures is tested and the results are presented in Tables 2.8 and 2.9. The JM model with RC achieves better results in the I and P GOP structure than in the I, B and P GOP structure since reference frames are close to a current frame. However, the propose method consistently shows good results.

2.5 Conclusion

In this section, we proposed a general framework to solve the RD optimization problem by using the primal-dual decomposition and subgradient projection methods. As a result of primal decomposition, we reused the JM model of H.264 to solve the RD optimization problem with prediction dependency. Furthermore, the optimality conditions for a given GOP bit constraint are derived under consideration of temporal dependency. From the experimental results, the proposed method is promising in compensating for loss of coding efficiency for nonreal-time applications with a constant bit rate constraint. However, the complexity of the encoder increases proportionally to the number of iterations that highly depends

Table 2.4: Coded bits and their difference ratio (%) with independent assumption

Sequence	Resolution	H.264 w/o RC	H.264 w/ RC	Proposed
FOOTBALL	QCIF	125374	-11.45	-1.08
BUS	QCIF	81569	-11.67	-1.90
FOREMAN	QCIF	40572	-9.51	+7.37
MOBILE	QCIF	51493	-5.05	+0.69
NEWS	QCIF	20598	-3.56	+1.11
CARPHONE	QCIF	20028	-6.38	+3.76
PARIS	CIF	120936	-7.37	+1.54
TEMPETE	CIF	125409	-9.57	+0.84

Table 2.5: PSNR (dB) and their difference with independent assumption

Sequence	Resolution	H.264 w/o RC	H.264 w/ RC	Proposed
FOOTBALL	QCIF	28.26	-0.55	-0.07
BUS	QCIF	28.29	-0.40	+0.04
FOREMAN	QCIF	31.22	-0.35	+0.57
MOBILE	QCIF	27.43	-0.11	+0.01
NEWS	QCIF	31.37	-0.06	+0.07
CARPHONE	QCIF	31.80	-0.15	+0.22
PARIS	CIF	30.08	-0.17	+0.06
TEMPETE	CIF	28.73	-0.16	0.00

on initial values. Therefore, the JM model with RC can cooperate with the proposed method for initial values of iterations which reduces the number of iterations. Furthermore, a dual optimal solution λ is estimated for a given QP to reduce complexity. In the next section, we will present a method to adapt λ to video contents for a given QP.

2.6 Acknowledgement

Portions of the text of this chapter are adapted from material that has been submitted for publication as: C. An and T. Q. Nguyen, "Analysis of Lagrange Multiplier Selection in H.264 and Its adaptation using Classification-Maximization", under review with *the IEEE Transactions on Image Processing*, 2008, and C. An and T. Q. Nguyen, "Iterative Rate-Distortion optimization of H.264 with Constant Bit Rate Constraint" was published in *the IEEE Transactions on Image Process-*

ing, vol. 17, pp. 1605-1615, Sep., 2008, and C. An and T. Q. Nguyen, "Iterative R-D optimization of H.264", in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, ICASSP-2008, Las Vegas, NV, USA, Mar., 2008. The dissertation author was the primary researcher for these publications, and the co-author directed and supervised the research which forms the basis for this chapter.

Table 2.6: Coded bits and their difference ratio (%) with temporal dependency

Sequence	**Resolution**	**H.264 w/o RC**	**H.264 w/ RC**	**Proposed**
FOOTBALL	QCIF	129448	-10.49	+1.08
BUS	QCIF	82603	-13.99	-3.54
FOREMAN	QCIF	42453	-8.32	+0.60
MOBILE	QCIF	56008	-4.06	-0.80
NEWS	QCIF	22583	-2.44	+0.58
CARPHONE	QCIF	22004	-9.39	-0.49
PARIS	CIF	128667	-5.69	+0.04
TEMPETE	CIF	139088	-6.89	-1.17

Table 2.7: PSNR (dB) and their difference with temporal dependency

Sequence	**Resolution**	**H.264 w/o RC**	**H.264 w/ RC**	**Proposed**
FOOTBALL	QCIF	28.54	-0.53	+0.12
BUS	QCIF	28.78	-0.37	+0.27
FOREMAN	QCIF	31.66	-0.33	+0.56
MOBILE	QCIF	28.29	-0.12	+0.14
NEWS	QCIF	32.36	-0.06	+0.24
CARPHONE	QCIF	32.69	-0.16	+0.13
PARIS	CIF	30.76	-0.16	+0.41
TEMPETE	CIF	29.56	-0.14	+0.10

Table 2.8: Coded bits and their difference ratio (%) with temporal dependency

Sequence	**Resolution**	**H.264 w/o RC**	**H.264 w/ RC**	**Proposed**
Football	QCIF	199320	+0.14	+0.34
Bus	QCIF	104637	+1.87	+0.76
Foreman	QCIF	46374	-0.74	+0.23
Mobile	QCIF	87382	+2.66	-0.20
News	QCIF	25580	+6.11	+0.23
Carphone	QCIF	26308	+9.19	-0.71
Paris	CIF	164918	+4.27	+0.64
Tempete	CIF	230810	+1.71	+0.18

Table 2.9: PSNR (dB) and their difference with temporal dependency

Sequence	**Resolution**	**H.264 w/o RC**	**H.264 w/ RC**	**Proposed**
Football	QCIF	28.87	-0.03	+0.08
Bus	QCIF	28.74	+0.05	+0.09
Foreman	QCIF	31.68	-0.16	+0.09
Mobile	QCIF	27.39	+0.24	+0.10
News	QCIF	31.89	+0.19	+0.28
Carphone	QCIF	32.40	+0.37	+0.00
Paris	CIF	30.51	+0.15	+0.01
Tempete	CIF	29.27	+0.14	+0.11

3 Adaptive Lagrange Multiplier Selection

3.1 Introduction

In subsection 2.2.4, we discussed the Lagrange multiplier λ selection problem for a given QP in the JM model [9]. The relation between λ and QP in the JM model was presented with temporal prediction dependency in (2.42). However, we only adapt λ without considering temporal dependency in this subsection. In (2.40), λ is fixed for a given QP and GOP structure even though various sequences show different c values as shown in Figure 2.9. Furthermore, each MB and picture can have various RD characteristics in Figure 3.1 which is not reflected in (2.40). In order to overcome the limitations of equation (2.40), reference [37] proposed the adaptive λ estimation method in the ρ-domain which defines the percentage of zeros in quantized residuals: $\lambda = \delta_1 \ln\left(\frac{\sigma^2}{D(\rho)} + \delta_2\right)\frac{D(\rho)}{R(\rho)}$ where σ^2 is the variance of prediction errors, and δ_1 and δ_2 are model parameters which are decided by experiments before encoding. The authors of [37] did not discuss how to choose the model parameters which might be different in various sequences. The previous work [37] was extended to derive the relation between λ and QP in [38, 39] instead of the relation between λ and ρ. However, the formulation of the relation was too complex to be presented in [38, 39]. The authors [40] showed a very interesting interpretation of the RD optimization problem as a maximization problem of the posterior probability using the Generalized Gaussian distribution. However, they derived an incorrect interpretation for the mode decision since we do not

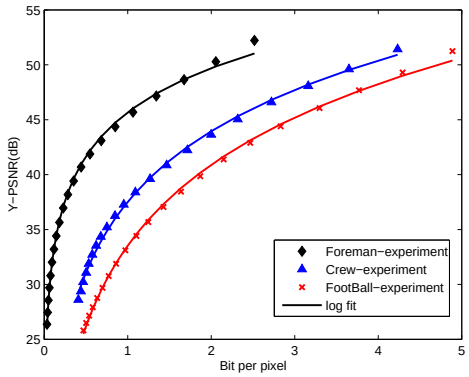


Figure 3.1: Y-PSNR (dB) vs. coded bits of the FOREMAN, CREW and FOOTBALL sequences.

code quantization errors but rather quantized prediction errors. All the previous work [20,21,37–41] extended (2.38) and (2.39) to incorporate model parameters to capture variations of RD characteristics which can change adaptively λ. In this section, we will propose a data-driven approach such as Classification-Maximization (CM) [10] to adapt λ to video sequences. Note that the CM algorithm does not guarantee the global optimal solution but we will see that λ of CM are very close to λ of Greedy Search (GS) in section 3.3.

3.2 Proposed Adaptive Lagrange Multiplier Selection

We will apply the CM algorithm to adapt the Lagrange multiplier λ for a given QP to minimize the Lagrangian cost in the mode decision procedure. Next, variations of the CM algorithm such as Relaxed CM (RCM) and Incremental CM (ICM) will be proposed to improve the CM algorithm and to avoid iterations of the CM algorithm.

3.2.1 Classification-Maximization (CM)

We reformulate the minimization of the Lagrangian cost (3.1) in the RD optimization as the maximization of the posterior distribution (3.5) as follows:

$$\mathbf{m}^* = \arg\min_{\mathbf{m}} \sum_{n=1}^{N} d_n(\mathbf{m}_n) + \lambda x_n(\mathbf{m}_n) \tag{3.1}$$

$$= \arg\min_{\mathbf{m}} \sum_{n=1}^{N} \beta_{ML} d_n(\mathbf{m}_n) + \alpha_{ML} x_n(\mathbf{m}_n) \tag{3.2}$$

$$= \arg\max_{\mathbf{m}} \prod_{n=1}^{N} \left\{ \left(\frac{\beta_{ML}}{\pi} \right)^{\frac{1}{2}} e^{-\beta_{ML} d_n(\mathbf{m}_n)} \right\} \left\{ \alpha_{ML} e^{-\alpha_{ML} x_n(\mathbf{m}_n)} \right\} \tag{3.3}$$

$$= \arg\max_{\mathbf{m}} \prod_{n=1}^{N} P(\mathbf{y}_n, x_n | \mathbf{m}_n, \beta_{ML}, \alpha_{ML}, \mathbf{y}_{ref}, QP) \tag{3.4}$$

$$= \arg\max_{\mathbf{m}} \prod_{n=1}^{N} P(\mathbf{m}_n | \mathbf{y}_n, x_n, \beta_{ML}, \alpha_{ML}, \mathbf{y}_{ref}, QP) \tag{3.5}$$

$$\approx \arg\max_{\mathbf{m}} \int \prod_{n} P(\mathbf{m}_n | \mathbf{y}_n, x_n, \beta, \alpha, \mathbf{y}_{ref}, QP) P(\beta, \alpha | \mathbf{y}_n, x_n, \mathbf{y}_{ref}, QP) d\alpha d\beta \tag{3.6}$$

where $\lambda = \frac{\alpha_{ML}}{\beta_{ML}}$ and $\alpha_{ML}, \beta_{ML} > 0$ in (3.2). Normalization factors for the Gaussian and Exponential distributions are multiplied in (3.3). Note that the authors of [40] derived the same formulation (3.5) with considering the first and second parts of (3.1) as the likelihood function and a prior probability, respectively. Here, we formulate two terms of (3.1) as the joint distribution (3.4). Thus, the posterior distribution (3.5) is derived from the likelihood function (3.4) with a uniform prior probability of $\mathbf{m}_n$. This assumption is reasonable since there is no preference to any MB mode during the mode decision procedure. In (3.6), uncertainty of the model parameters α and β is considered. However, the maximum values of the posterior probability of α and β are only considered in the maximization step. The more complete procedure is described in Appendix A.3. Consequently, the CM algorithm which comprises the classification step and the maximization step is applied iteratively as follows:

C-step: For given parameters β_{ML}, α_{ML}, QP and references $\mathbf{y}_{ref}$, MB modes are classified to minimize the Lagrangian cost, that is, maximize the posterior probability.

M-step: The likelihood function $P(\mathbf{y}, \mathbf{x}|\alpha, \beta, \mathbf{M}, QP, \mathbf{y}_{ref})$ is maximized for given classified MB modes:

$$\alpha_{ML} = \frac{N}{\sum_{n=1}^{N} x_n(\mathbf{m}_n^*)}, \quad \frac{1}{\beta_{ML}} = \frac{2\sum_{n=1}^{N} d_n(\mathbf{m}_n^*)}{N}$$
$$\lambda = \frac{\alpha_{ML}}{\beta_{ML}} = \frac{2\sum_{n=1}^{N} d_n(\mathbf{m}_n^*)}{\sum_{n=1}^{N} x_n(\mathbf{m}_n^*)} \tag{3.7}$$

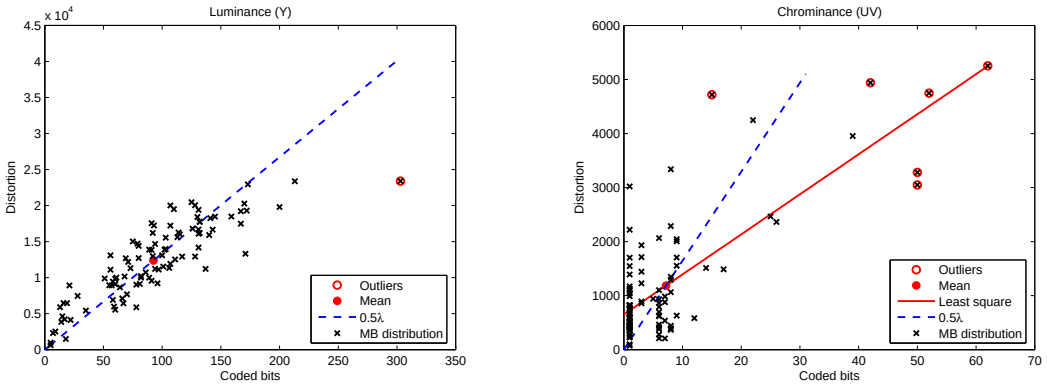
$$\lambda_L = \lambda, \quad \lambda_C = \frac{2\{N\sum_{n=1}^{N} d_n^C(\mathbf{m}_n^*)x_n^C(\mathbf{m}_n^*) - \sum_{n=1}^{N} x_n^C(\mathbf{m}_n^*)\sum_{n=1}^{N} d_n^C(\mathbf{m}_n^*)\}}{N\sum_{n=1}^{N}\left(x_n^C(\mathbf{m}_n^*)\right)^2 - \left(\sum_{n=1}^{N} x_n^C(\mathbf{m}_n^*)\right)^2} \tag{3.8}$$

Note that the CM algorithm of this dissertation is slightly different from the general CM algorithm because we use a single λ for all the MB modes (classes) so that we do not separately maximize the likelihood function for each class. Finally, λ is updated as in (3.7) at each iteration. For a given updated λ, the C-step can be performed until convergence. A more detailed formulation is described in Appendix A.3.

As a result of the M-step, λ is just 2 times average distortion over average bits in (3.7). Figures 3.2 (a) and 3.2 (b) show that the distribution of distortion and code bits of luma and chroma are quite different. Therefore, we apply different λ for luma and chroma as follows:

$$\begin{aligned}\mathbf{m}^* &= \arg\min_{\mathbf{m}} \sum_{n=1}^{N} d_n(\mathbf{m}_n) + \lambda x_n(\mathbf{m}_n) \\ &= \arg\min_{\mathbf{m}} \sum_{n=1}^{N} d_n(\mathbf{m}_n) + \lambda_L x_n^L(\mathbf{m}_n) + \lambda_C x_n^C(\mathbf{m}_n) - w_0\end{aligned}$$

where the MB distortion $d_n(\mathbf{m}_n)$ is the sum of the luma distortion $d_n^L(\mathbf{m}_n)$ and the chroma distortion $d_n^C(\mathbf{m}_n)$, and the MB coded bits $x_n(\mathbf{m}_n)$ is the sum of the luma bits with the MB header bits $x_n^L(\mathbf{m}_n)$ and the chroma bits $x_n^C(\mathbf{m}_n)$. The constant offset w_0 is subtracted for chroma since chroma is highly biased as shown in Figure 3.2 (b). The luma λ_L and the chroma λ_C in (3.8) are separately derived from the M-step of the CM algorithm. Especially, if we derive λ for chroma without considering the offset w_0, λ_c is very large to represent the relation between distortion and coded



(a) Luma distribution.

(b) Chroma distribution.

Figure 3.2: Distribution of luminance and chrominance distortion and coded bits in an I picture.

bits which is denoted as the dashed line in Figure 3.2 (b). Note that λ of luma is 2 times the slope of the dashed line in Figure 3.2 (a) and λ of chroma is 2 times the slope of the solid line in Figure 3.2 (b) as a result of (3.8). We can derive the more robust relation between distortion and coded bits after removing outliers which are circled in Figures 3.2 (a) and 3.2 (b). Here, outliers are classified if distortion or coded bits of MBs are greater than or smaller than 3 times the standard deviation from average distortion or average coded bits.

3.2.2 Relaxed Classification-Maximization (RCM)

In this subsection, we propose the RCM agorithm to improve the performance at high QP (QP $\geq$ 30). In the CM algorithm, distortion and coded bits of classified MB modes $\mathbf{m}^*$ are only used to decide λ in (3.8). In the intra MB mode decision, we select one luma intra mode among 144 (16 blocks $\times$ 9 predictions) 4x4 luma intra prediction modes and four 16x16 luma intra prediction modes, and we decide one chroma intra mode among four 8x8 chroma intra prediction modes to minimize the Lagrangian cost for one MB in (3.1). When an updated λ is applied for the C-step after the M-step, the updated λ can cause a different mode to be chosen among the possible modes. Especially, all the distortion and coded

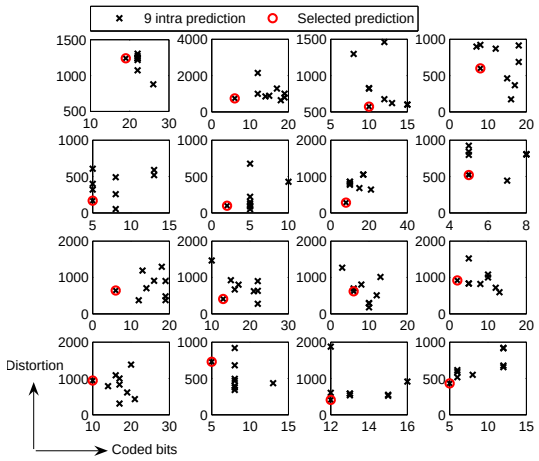


Figure 3.3: Distribution of distortion and coded bits of nine 4x4 intra prediction in a MB.

bits of 4x4 luma intra prediction are very closely distributed as shown in Figure 3.3. Thus, an updated λ usually causes a different prediction mode to be the best mode without changing MB types (16x16 and 4x4 MB types). We can confirm this result from Figure 3.4. A different MB type is rarely chosen with different λ from Figure 3.4 (d) and prediction modes mostly vary with different λ as shown in Figures 3.4 (a), (b) and (c). Therefore, we can consider all the distortion and coded bits of prediction modes in a MB type to derive λ. However, we do not consider distortion and coded bits of different MB types. For example, if a 4x4 MB type is selected after the mode decision procedure, distortion and coded bits of the 16x16 MB type are not included but all the distortion and coded bits of the 4x4 MB type are considered to derive λ at the M-step. Consequently, we relax the C-step to include possible variations.

3.2.3 Incremental Classification-Maximization (ICM)

We propose the ICM algorithm to avoid iterations of the CM and RCM algorithms. The ICM algorithm performs the C-step and the M-step at each MB instead of a picture level which is done in the CM and RCM algorithms. Note that we denoted the ICM algorithm as an incremental version of the CM algorithm but

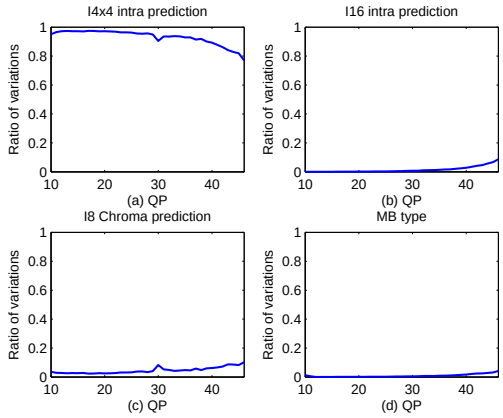


Figure 3.4: Ratio of variations of 4x4 luma intra prediction, 16x16 luma intra prediction, 8x8 chroma prediction and MB types w.r.t. variations of λ at each QP.

it also includes an incremental version of the RCM algorithm, because the RCM algorithm is just a relaxed version of the CM algorithm at the C-step. After the best MB mode is decided (C-step), λ is updated as follows (M-step):

$$\begin{aligned}\lambda^k &= \frac{2\sum_{n=1}^{k} d_n(\mathbf{m}_n^*)}{\sum_{n=1}^{k} x_n(\mathbf{m}_n^*)} = \frac{2\sum_{n=1}^{k-1} d_n(\mathbf{m}_n^*)}{\sum_{n=1}^{k} x_n(\mathbf{m}_n^*)} + \frac{2d_k(\mathbf{m}_k^*)}{\sum_{n=1}^{k} x_n(\mathbf{m}_n^*)} \\ &= \left(1 - \frac{x_k(\mathbf{m}_k^*)}{\sum_{n=1}^{k} x_n(\mathbf{m}_n^*)}\right)\lambda^{k-1} + \frac{2d_k(\mathbf{m}_k^*)}{\sum_{n=1}^{k} x_n(\mathbf{m}_n^*)} \\ &= \lambda^{k-1} + \frac{1}{\sum_{n=1}^{k} x_n(\mathbf{m}_n^*)}\left(2d_k(\mathbf{m}_k^*) - \lambda^{k-1}x_k(\mathbf{m}_k^*)\right)\end{aligned}$$

where λ^0 can be derived from λ of the JM model or average λ of the previous frame. Thus, we can simply update λ from the previous λ, current coded bits and distortion at each MB. Here, we avoid applying different λ for luma and chroma from experimental results. Chroma λ is adaptively derived from luma λ.

3.3 Experimental Results

First, we greedily search λ to achieve the highest PSNR at the same coded bits for every picture. In order to find the best λ in RD sense, the 3-degree polynomial is used to fit a PSNR curve of a frame as in [36]. In Figure 3.5, we

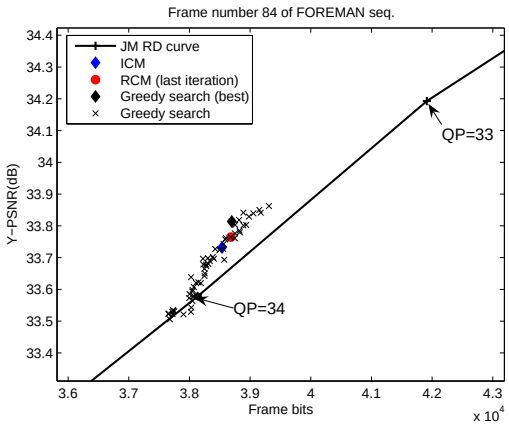


Figure 3.5: PSNR vs. coded bits of JM, RCM, ICM and GS in an I picture.

compare the PSNR and coded bits of JM [9], RCM, ICM and GS. The RD curve of the JM model is obtained from $QP = 32$ to $QP = 35$ for an I picture. GS is performed by changing λ from λ of a center QP - 2 to λ of a center QP + 1 where the center QP is 34 and a step size is 1% of λ from (2.40) for the given center QP which is denoted as λ of JM. Here, the ICM algorithm is an incremental version of the RCM algorithm. Figure 3.5 shows that the performance of ICM is very close to RCM (only the last iteration is marked) and GS. We also compare the best λ of GS and the average λ of ICM with λ of JM in the FOREMAN sequence as shown in Figure 3.6. Note that all the frames are coded as I pictures and average λ of all the MBs in a picture is illustrated in ICM since λ of each MB is different. λ of JM is used for the initial λ, that is, λ_0 at each picture in ICM. Figure 3.6 shows that λ of JM is constant for all the pictures as a result of (2.40) but λ of GS, which is smaller than λ of JM, highly fluctuates. λ of ICM are close to the best λ of GS in all the pictures and they follow the general variations of λ of GS. The average λ of JM, ICM and GS in the whole sequence are 137.08, 103.44 and 97.00, respectively. The Y-PSNR (dB) gain of ICM and GS with respect to JM at the same coded bits are depicted in Figure 3.7. Generally, ICM gives better results than JM but there are a few worse frames since the M-step of ICM does not consider spatial dependency, that is, ICM assumes that all the MBs are independent. It is confirmed that the

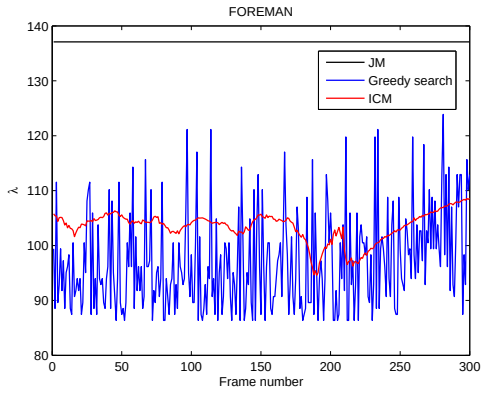


Figure 3.6: λ of JM, ICM and GS at QP=34.

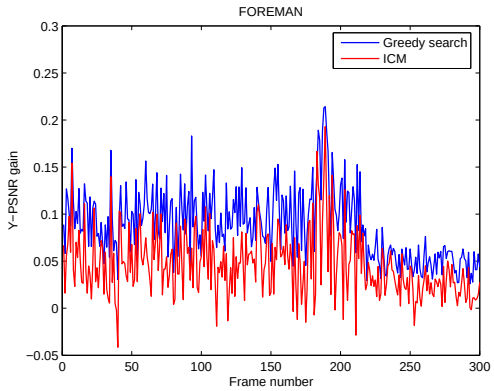


Figure 3.7: Y-PSNR gain of ICM and GS w.r.t. JM at QP=34.

likelihood function of a frame is the multiplication of the likelihood functions of MBs in Appendix A.3. Therefore, spatial dependency may induce unpredictable results. However, ICM generally achieves better results and does similar gain of GS at the high-gain frames which are from the 180th frame to the 200th frame in Figure 3.7.

Next, we apply the ICM and RCM algorithms to various sequences with different resolutions and frame rates. The 7-degree polynomial is used to fit PSNR in order to calculate the PSNR difference at the same code bits [36]. The experimental results are presented in Tables 3.1 and 3.2 where we compare Y-PSNR of three methods from $QP = 20$ to $QP = 46$. In Table 3.1, the performance of

Table 3.1: Maximum and average Y-PSNR (dB) gain of QCIF size I pictures from $QP = 20$ to $QP = 46$

Sequence (QCIF)	CM		RCM		ICM	
	Max.	Avg.	Max.	Avg.	Max.	Avg.
SILENT	0.07	0.04	0.07	0.05	0.06	0.04
CARPHONE	0.08	0.06	0.08	0.05	0.08	0.05
NEWS	0.10	0.05	0.10	0.04	0.09	0.04
FOREMAN	0.09	0.04	0.10	0.03	0.10	0.04
CREW	0.08	0.05	0.09	0.05	0.08	0.05
MOBILE	0.05	0.03	0.05	0.02	0.05	0.03
FOOTBALL	0.03	0.02	0.04	0.00	0.04	0.03
SOCCER	0.15	0.05	0.14	0.03	0.11	0.03

Table 3.2: Maximum and average Y-PSNR (dB) gain of CIF size I pictures from $QP = 20$ to $QP = 46$

Sequence (CIF)	RCM		ICM	
	Max.	Avg.	Max.	Avg.
SILENT	0.10	0.04	0.05	0.03
NEWS	0.12	0.05	0.12	0.06
FOREMAN	0.15	0.04	0.15	0.05
STEFAN	0.06	0.04	0.05	0.03
CREW	0.07	0.05	0.06	0.05
SOCCER	0.13	0.03	0.10	0.02

RCM, CM and ICM is similar at the QCIF sequences. However, CM is usually slightly better than RCM from $QP = 20$ to $QP = 30$, and ICM is better than RCM and CM in the MOBILE and FOOTBALL sequences even though RCM and CM incorporate iterations. The main reason is that ICM naturally uses different λ for MBs but RCM and CM apply the same λ for all the MBs within a frame. Sometimes, different λ within a frame are helpful to improve PSNR because of spatial dependency. In the CIF size sequences, we do not compare PSNR of CM since the performance of CM is similar to RCM. In simple background sequences such as SILENT, NEWS and FOREMAN in Tables 3.1 and 3.2, RCM and ICM obtain higher PSNR gain at the CIF size than at the QCIF size. λ of JM, which is derived from uniform quantization and high-rate entropy constrained coding, is not accurate in simple background sequences and high QP since the distribution of

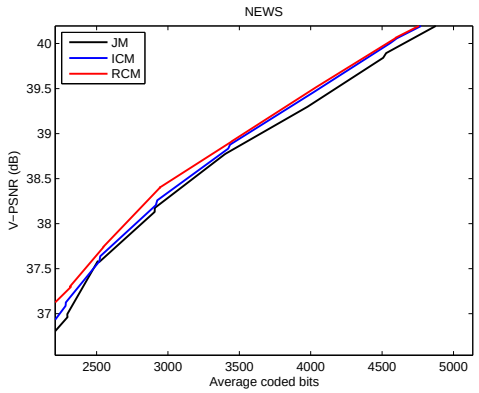


Figure 3.8: V-PSNR of JM, ICM and RCM

DCT coefficients is not uniform at high QP and the other bits such as bits for MB prediction modes instead of residual bits are dominant. Therefore, the proposed methods can achieve higher gain than JM. From the experimental results, ICM is generally better than JM from $QP = 25$ to $QP = 45$ but the JM model is better at the other QP.

In CM and RCM, different λ are applied for luma and chroma. Here, we only show Figure 3.8 to present some gains from different λ for luma and chroma because the RD curve of chroma is highly nonlinear so that the PSNR difference is not accurate from curve fitting. The main reason of nonlinearity in the chroma RD curve is that QP for chroma, that is, QPc in H.264 [1] is nonlinearly related with QP for luma. From Figure 3.8, the RCM algorithm achieves higher PSNR than the JM model at the same code bits for chroma. Even though we apply the same λ for luma and chroma in the ICM algorithm, adaptivity of λ to video contents can improve PSNR of chroma as well as luma in Figure 3.8.

3.4 Conclusion

In this section, we solved the adaptive Lagrange multiplier selection problem using the CM, RCM and ICM algorithms. From the experimental results, the simple adaptive method ICM achieves similar performance to RCM and CM, and RCM is generally better than CM at high QP. Therefore, we need to apply the different methods according to the operating QP range. This issue can be solved by Expectation-Maximization (EM) [42] since the EM algorithm automatically controls relaxation of the CM algorithm. However, we did not apply EM and its online EM [43] with consideration of improvement versus complexity. Here, we only applied the proposed methods for I pictures but the proposed methods can be applied for λ of P and B pictures. However, we need to consider the Lagrange multiplier selection problem with temporal dependency in order to improve the performance of P and B pictures because temporal dependency is dominant for λ of P and B pictures. As mentioned in section 2.3 for temporal dependency, we need to decide relative difference among the I, P and B pictures. Therefore, it is still necessary to estimate more accurate λ of all the pictures using the CM methods and then relative difference is considered among the pictures for temporal dependency. We consider it for future work.

3.5 Acknowledgement

Portions of the text of this chapter are adapted from material that has been submitted for publication as: C. An and T. Q. Nguyen, "Analysis of Lagrange Multiplier Selection in H.264 and Its adaptation using Classification-Maximization", under review with *the IEEE Transactions on Image Processing*, 2008, and C. An and T. Q. Nguyen, "Analysis of Utility Functions for Video", in *Proceedings of the IEEE International Conference on Image Processing*, ICIP-2007, San Antonio, TX, USA, Sep., 2007, and C. An and T. Q. Nguyen, "Resource Allocation for TDMA Video Communication over AWGN using Cross-Layer Optimization Approach" was accepted to appear in *the IEEE Transactions on Multimedia*, 2008.

The dissertation author was the primary researcher for these publications, and the co-author directed and supervised the research which forms the basis for this chapter.

4 Intra Prediction

4.1 Introduction

H.264 [1] applies inter and intra prediction to remove temporal and spatial redundancies among the pictures in Figure 1.1. In this chapter, we only consider intra prediction to reduce spatial redundancies within a picture. Intra prediction uses reconstructed neighboring MB pixels to predict current MB pixels. In H.264 [1], various directional filters are applied to predict current MB pixels during the mode decision procedure. The best filter which minimizes the Lagrangian cost in (2.5) is then selected and a specified index of the best filter is transmitted to the decoder. Thus, the previous research [44–51] mainly proposed methods to improve the performance of intra prediction or to decide the best intra prediction filter with low complexity and minimum loss of performance. [44,45] proposed fast mode decision in the DCT domain and the methods in [46,47] presented how intra prediction in the pixel domain corresponds to one in the DCT domain. [46] proposed the additional prediction modes with complexity of the mode decision procedure increased. The authors of [48] matched a template for 2x2 block with similarity measure in the pixel domain and applied the best matched block for intra prediction. The approaches in [49,50] imitated motion estimation and compensation to search for the best matching block among the neighboring blocks with subpixel accuracy. Reference [51] proposed the hybrid method to use both intra prediction of H.264 and block matching. The previous work [48–51] improved the performance in the PSNR sense but the improvements were under 0.5 dB.

In the proposed methods in sections 4.3 and 4.5, we will simultaneously con-

sider both improvement of intra prediction and simplification of the mode decision procedure at the same time. This can be achieved by applying an accurate prediction method instead of using multiple simple prediction methods. For this work, we consider statistical learning methods such as Support Vector Machines for Regression (SVR) [52–54] and Locally Weighted Projection Regression (LWPR) [55] to improve the performance of current H.264 intra prediction by means of batch and online learning. Support vector machines for classification and regression have been developed under profound theoretical background and have been successfully applied to many classification and time series prediction problems [54]. Even though SVR can be extended for online learning [56–58], we only apply SVR for batch learning and use LWPR as an online learning method. The main reasons are that online SVR [56] updates globally coefficients of SVR which is endangered by negative inference, that is, forgetting of useful knowledge while learning from new data [59] and online SVR in [57, 58] cannot guarantee a bound on the number of operations required per iteration. Therefore, they are difficult for real-time applications. In order to simplify the mode decision procedure, we only use a single MB type and one or two prediction methods: learning based prediction or spatial average prediction. To our knowledge, this work is the first to apply learning methods for video compression.

4.2 H.264 Intra Prediction

H.264 [1] uses 9 directional intra prediction modes for 4x4 blocks and 4 intra prediction modes for 16x16 MB. The main idea of H.264 intra prediction is extrapolation of the pixels which are on the row and column directly adjacent to a current block. Adjacent pixels, denoted as grey areas in Figure 4.1, are used for prediction of the current block. All the intra prediction modes of each MB type are performed in the pixel domain through directional extrapolation. For example, 4x4 intra prediction extrapolates adjacent neighboring pixels in 9 directions which are indexed from 0 to 8 as shown in Figure 4.1 and Table 4.1. The best prediction mode is decided by the mode decision procedure which minimizes the Lagrangian

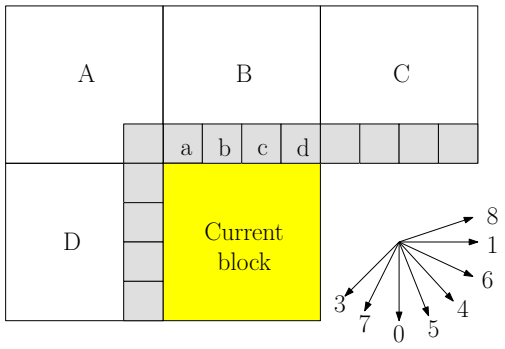


Figure 4.1: H.264 intra prediction

Table 4.1: 4x4 intra prediction modes

4x4 intra prediction mode	Direction
0	vertical
1	horizontal
2	DC
3	down left
4	down right
5	vertical right
6	horizontal down
7	vertical left
8	horizontal up

cost [11,12]. The prediction errors are transformed by a DCT, and DCT coefficients are then quantized. Finally, quantization indices are coded by entropy coding such as CAVLC and CABAC [1, 2]. Reconstructed pixels are obtained from adding predicted pixels to errors which are reconstructed by decoding, inverse quantization and inverse DCT transformation in Figure 1.1. Note that these reconstructed pixels are used for intra prediction instead of original pixels in order to prevent the drift problem between the encoder and the decoder. The deblocking filter is not applied to the reconstructed pixels for intra prediction. This step is also applied to batch and online learning based intra prediction such that reconstructed DCT coefficients or reconstructed pixels are input features without the deblocking filter in subsections 4.3.3 and 4.5.5.

Even though the directional intra prediction of H.264 effectively reduces spatial redundancies on various sequences, H.264 pixel domain prediction only predicts a portion of DCT coefficients [47] if we transform H.264 intra prediction in the pixel domain to one in the DCT domain. For example, the vertical prediction of the 4x4 block only predicts 4 DCT coefficients as follows:

$$DCT \begin{bmatrix} a & b & c & d \\ a & b & c & d \\ a & b & c & d \\ a & b & c & d \end{bmatrix} = DCT \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x & x & x & x \\ x & x & x & x \\ x & x & x & x \\ a & b & c & d \end{bmatrix}$$

$$= \begin{bmatrix} 1 & -1.264 & 1 & -0.632 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} DCT \begin{bmatrix} x & x & x & x \\ x & x & x & x \\ x & x & x & x \\ a & b & c & d \end{bmatrix}$$

where a, b, c and d are adjacent pixels of the top block in Figure 4.1 and x denotes any value. More complicated directional prediction modes such as modes 7 and 8 predict all the DCT coefficients but they have regularity in the DCT domain [47]. In order to reduce these limitations, new directional prediction can be added which increases complexity of the mode decision procedure. In the next section, we will improve the prediction performance as well as simplify the mode decision procedure using learning based intra prediction.

4.3 Intra Prediction through Batch Learning

4.3.1 Support Vector Machines for Regression

We consider SVR as a batch learning method. SVMs are statistical learning tools based on Vapnik-Chervonenkis (VC) theory and Structural Risk Minimization (SRM) principles [54]. SRM is an inductive principle for model selection which is used for learning from finite training data sets. It describes a general model of capacity control and provides a trade-off between hypothesis space complexity (VC

dimension of approximating functions) and quality of fitting the training data. In the statistical learning theory and SVMs, regularization networks also can approximately implement SRM principles when an optimal regularization parameter is chosen [60]. Therefore, SVR solves a Regularized Risk (summation of empirical risk and regularizer) Minimization (RRM) problem to estimate a linear function $f(\mathbf{x}) = \langle \mathbf{w}, \Phi(\mathbf{x}) \rangle + b$ for the ϵ-incentive loss function as follows [54]:

$$\begin{aligned} \min_{\mathbf{w},b,\xi,\xi^*} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i^*) \\ s.t.(\langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle) - y_i \leq \epsilon + \xi_i \\ y_i - (\langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle) \leq \epsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0, i = 1, ..., N \end{aligned} \tag{4.1}$$

where C is an inverse regularization parameter and Φ is a nonlinear mapping function from input data $\mathbf{x}$ into a high-dimensional feature space. N denotes the number of training samples and $\xi_i^{(*)}$ are slack variables to allow violations of conditions, commonly referred to as a soft margin. y_i are outputs corresponding to input data $\mathbf{x}_i$ and ϵ is a parameter which denotes zero loss if an absolute prediction error $|y_i - f(\mathbf{x}_i)|$ is smaller than ϵ as shown in Figure 4.2. The optimization problem (4.1) is a quadratic convex optimization problem and its solutions are global optimal solutions which is a main feature of SVMs. The primal optimization problem (4.1) can be solved by a primal optimization view [61] or a dual optimization view through the Lagrangian duality [54]. Two optimization views derive the same regression function $f(\mathbf{x})$ as a solution of (4.1):

$$\begin{aligned} f(\mathbf{x}) = \langle \mathbf{w}, \Phi(\mathbf{x}) \rangle + b &= \sum_{i=1}^{N}(\alpha_i^* - \alpha_i)\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}) \rangle + b \\ &= \sum_{i \in SV}^{\#SV}(\alpha_i^* - \alpha_i)\langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}) \rangle + b \qquad (4.2) \\ &= \sum_{i \in SV}^{\#SV}(\alpha_i^* - \alpha_i)k(\mathbf{x}_i, \mathbf{x}) + b \qquad (4.3) \end{aligned}$$

where $\mathbf{w} = \sum_{i=1}^{N}(\alpha_i^* - \alpha_i)\Phi(\mathbf{x}_i)$ and $\alpha_i^{(*)}$ are dual optimal solutions of a dual optimization problem. Note that input data $\mathbf{x}_i$ which are associated with nonzero

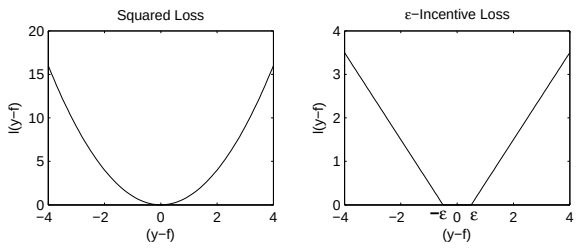


Figure 4.2: The squared loss function and the ϵ-incentive loss function.

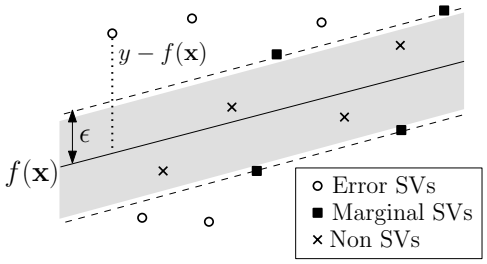


Figure 4.3: Classification of training input data $\mathbf{x}_i$.

$\alpha_i^{(*)}$ are called Support Vectors (SVs). After solving (4.1), training data $\mathbf{x}_i$ are classified into three types according to the absolute prediction errors $|y_i - f(\mathbf{x}_i)|$: marginal SVs, error SVs and non SVs as illustrated in Figure 4.3. Thus, if absolute prediction errors are equal to ϵ, the input data $\mathbf{x}_i$ are called marginal SVs and if they are larger than ϵ, the input data are error SVs and otherwise, the input data are not SVs whose $\alpha^{(*)}$ are zeros. Consequently, only SVs among the training data contribute regressor output as (4.2) which gives sparse solutions to SVMs with the ϵ-incentive loss function. However, the sparsity is only achieved by specific loss functions which have zero-gradient loss functions.

Reference [61] denoted that the dual optimal solutions $\alpha_i^{(*)}$ are related with gradients of a loss function, that is, $\alpha_i^{(*)}$ are zeros if gradients of a loss function at $\mathbf{x}_i$ are zeros. From Figure 4.2, gradients of the squared loss function are not zeros except at zero, but the ϵ-incentive loss function has zero loss and zero gradients between $-\epsilon$ and ϵ. Therefore, any input data which have absolute loss within ϵ are not SVs. Least Squares (LS) SVM [62] which uses the squared loss function

does not have sparse solutions. In the proposed method, we only consider the ϵ-incentive loss function for sparse solutions. In order to reduce complexity of inner products in the high dimensional feature (Hilbert) space in (4.2), the kernel trick [54] was introduced to compute inner products in the feature space through a kernel function on input data $\mathbf{x}_i$ in (4.3) as follows: $k(\mathbf{x}_i, \mathbf{x}) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}) \rangle$. In particular, the Radial Basis Function (RBF) is considered as a kernel function in the proposed method:

$$k(\mathbf{x}_i, \mathbf{x}) = e^{-\gamma \|\mathbf{x}_i - \mathbf{x}\|_2^2} \tag{4.4}$$

4.3.2 Model Selection of Support Vector Regression

Before solving the optimization problem (4.1), we have to decide the kernel function $k(\cdot)$, the kernel parameter γ, the loss function parameter ϵ and the inverse regularization parameter C. This decision process is known as model selection. In the proposed method, the well-known RBF is used and the other parameters are obtained from Cross Validation (CV). In particular, 5 fold CV is considered, that is, training data $\mathbf{x}_i$ are divided into 5 sets and then one of the 5 sets is used for test and the others are used for training to determine SVs and their weights $\alpha^{(*)}$ for given parameters. This operation is performed 5 times using a different test set each time. Finally, the best parameters which give the minimum average Mean Square Error (MSE) through 5 fold CV are applied to the problem (4.1). Generally, grid-search on the parameters γ, ϵ and C is used for CV [63]. However, references [64–66] obtained better performance from the Genetic Algorithm (GA) for model selection. The GA is a powerful stochastic search and optimization technique based on the processes of evolution theory. The GA uses three operators to generate test patterns: reproduction, crossover and mutation. In this subsection, we use the GA Matlab tool box [67]. Because optimal parameters do not exist uniquely [68], model parameters are selected to generate fewer SVs and smaller C in the case of the same MSE to find sparser solutions and smaller regularized risk in (4.1). Figure 4.4 shows that the GA generates better test patterns which are closer to the global solutions according to generations because of the smaller (average) minimum MSE. However, it keeps generating new test patterns from

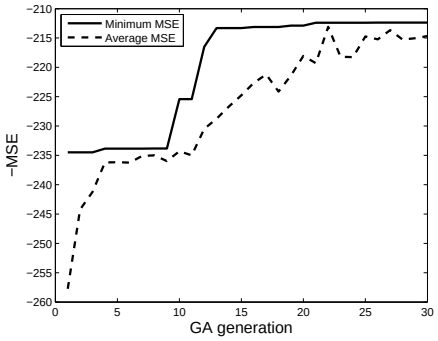


Figure 4.4: Example of model selection in the GA.

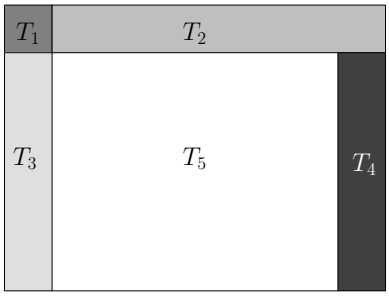


Figure 4.5: Classes of input features in a frame.

three operators to escape local minima.

4.3.3 Batch Learning based Intra Prediction

In this subsection, we apply SVR for batch learning based intra prediction which is not compliant with H.264 [1]. Here, we propose a new video coding method by means of SVR. For this work, we decide output and input features of SVR. The output of SVR $f(\mathbf{x})$ is a DCT coefficient and input features $\mathbf{x}$ in (4.3) are DCT coefficients of neighboring MBs. Input features are classified into 5 types from T_1 to T_5 whose classifications are based on available neighboring 8x8 MBs as in Figure 4.5. For example, a left MB is only available in the T_2 class. The main structural difference from H.264 [1] is that the 8x8 MB type and the a 8x8 DCT are only applied instead of the 16x16 and 4x4 MB types and a 4x4 DCT. Furthermore,

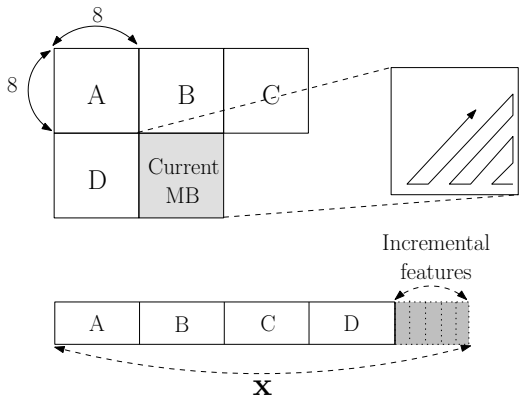


Figure 4.6: Patch of intra prediction and incremental intra prediction.

intra prediction is performed in the DCT domain by a single prediction method of SVR as opposed to the 9 or 4 directional prediction methods of the pixel domain in H.264. Thus, there is no need to allocate bits to indicate MB types and the best intra prediction method in the proposed method. Note that classification types are fixed according to the position of a frame, which is already known at the decoder.

Figure 4.6 illustrates that DCT coefficients of a current MB are predicted from incremental intra prediction with the inverse zig-zag scan order in the T_5 class. The highest DCT coefficient is only predicted from DCT coefficients of neighboring MBs which are denoted as A, B, C and D in Figure 4.6. DCT coefficients of each MB become a vector by means of the zig-zag scan and are then concatenated into a vector of input features $\mathbf{x}$ as in Figure 4.6. If there are no available neighboring MBs, the highest DCT coefficient of the T_1 class is coded without intra prediction. The next highest DCT coefficient, based on the inverse zig-zag scan order, is predicted from DCT coefficients of neighboring MBs and the reconstructed highest DCT coefficient of a current MB which is obtained from adding the predicted DCT coefficient to the inverse quantized DCT coefficient. Finally, the DC coefficient of a current MB is predicted from DCT coefficients of neighboring MBs and all the reconstructed AC coefficients of a current MB. Incremental intra prediction overcomes smaller input features, especially, the T_1 MB which has no neighboring MBs. Current intra prediction of H.264 [1] subtracts 128

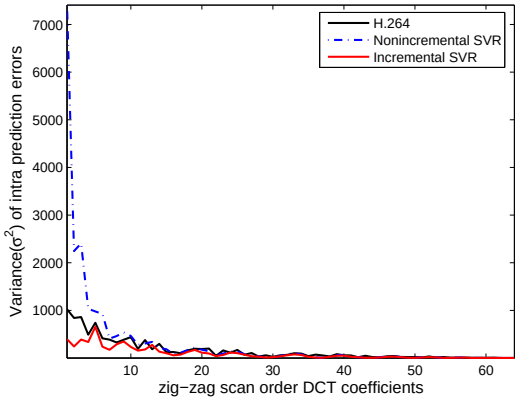


Figure 4.7: Variance of intra prediction errors of the T_5 class in the FOREMAN sequence.

in the pixel domain which corresponds to subtracting a constant value from the DC coefficient in the DCT domain. Incremental intra prediction with the inverse zig-zag scan order overcomes these limitations and utilizes the fact that low frequency DCT coefficients are more important than high frequency DCT coefficients. Thus, the DC coefficient and low frequency AC coefficients have higher dimension of input features than high frequency AC coefficients.

Figure 4.7 shows the variance of intra prediction errors of the DCT coefficients of the T_5 class in the FOREMAN sequence according to three different prediction methods: H.264 intra prediction, nonincremental batch intra prediction and incremental batch intra prediction. In this experiment, only a 8x8 DCT is applied for intra prediction errors to all three methods. We use the base layer of Joint Scalable Video Model (JSVM) [69] for intra prediction of H.264 which is compatible with H.264 [1]. Nonincremental batch intra prediction does not utilize current MB information for intra prediction, that is, it only uses neighboring MBs from A to D in Figure 4.6. Incremental batch intra prediction achieves smaller variance of prediction errors in the DCT domain than H.264 and nonincremental batch intra prediction in Figure 4.7. Thus, DCT coefficients of a current MB carry very important features to the SVR learning system.

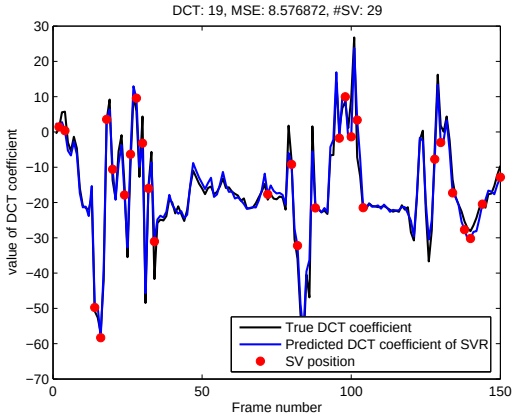


Figure 4.8: Intra prediction of the 19th DCT coefficient of the T_1 class in the FOREMAN sequence.

4.4 Experimental Results of Batch Intra Prediction

In this experiment, we assume that the encoder and the decoder already have SVs, their weights $\alpha^{(*)}$, b and the kernel parameter γ which are needed for prediction in (4.3). The FOREMAN sequence is used for training and test. Because the number of the T_1 class among all the frames is small, we train SVR of the T_1 class using every other frame. We use every 20th frame to train SVR of the T_5 class and every 5th frame for the T_2, T_3 and T_4 classes. The RBF kernel in (4.4) is considered as a kernel function and the kernel parameter γ, the loss function parameter ϵ and the regularization parameter C are obtained from the 5 fold CV through GA as explained in subsection 4.3.2. Then, the optimization problem (4.1) is solved by the LIBSVM tools [70] to decide SVs and their weights $\alpha^{(*)}$ for given model parameters.

As an example, Figure 4.8 illustrates intra prediction of the 19th DCT coefficient for the T_1 class in the FOREMAN sequence. There is only one T_1 class in a frame. 29 SVs of the 19th DCT coefficient whose positions are marked as circles in Figure 4.8 are used to predict all the T_1 classes in 150 frames. 29 SVs among the 75 training input data $\mathbf{x}$ are enough to predict the 19th DCT coefficient

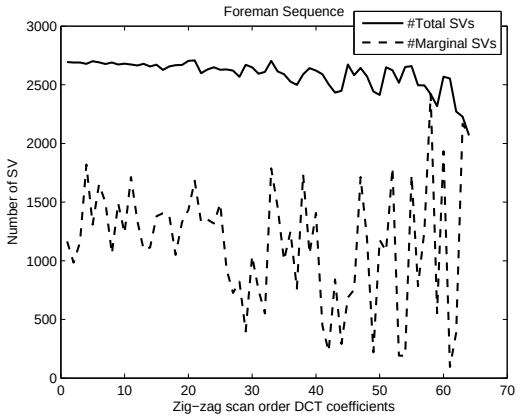


Figure 4.9: The number of SVs at the T_5 class in the FOREMAN sequence.

with ϵ-incentive loss from Figure 4.8. The number of total SVs, that is, the sum of the number of marginal SVs and the number of error SVs, and the number of marginal SVs of 64 DCT coefficients are shown in Figure 4.9. Almost half of SVs in the low frequency (under 23rd) DCT coefficients are on the margin which denotes that absolute prediction errors are exactly equal to ϵ in Figure 4.3. In contrast, prediction of the high frequency DCT coefficients except some DCT coefficients is difficult because the number of error SVs increases which means that absolute prediction errors are larger than ϵ during the training stage.

In Figure 4.10, the performance of H.264 intra prediction is compared with the performance of batch learning based intra prediction with and without classification. Here, we only compare the PSNR of intra prediction without considering coded bits because side information to indicate the best prediction mode and MB types is not coded in batch learning based intra prediction. Thus, batch intra prediction only uses the 8x8 MB type and one prediction mode. Note that the PSNR of intra prediction is obtained from prediction errors E_r at each frame using $10 \log_{10} \frac{255^2}{E_r^2}$. The regular peaks in Figure 4.10 result from the fact that every 20th frame is used for training since it is easy to predict training data in test. However, the PSNR improvement of the other frames is as high as 4dB. The average Y-PSNR of batch intra prediction with classification is over 3.6dB higher than H.264

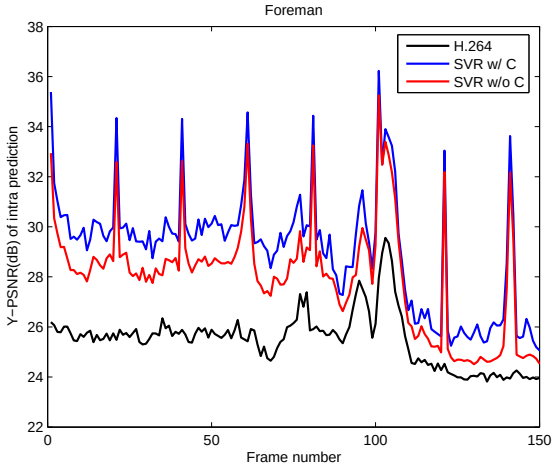


Figure 4.10: Y-PSNR (dB) of batch learning based intra prediction in the FOREMAN sequence.

intra prediction.

Even though batch intra prediction with classification significantly improves the performance of intra prediction over H.264, the complexity is very high since each class performs its own batch intra prediction. In order to reduce the complexity, only batch intra prediction which has T_5 input features is applied to all the frames. If there are no available neighboring MBs, DCT coefficients of corresponding MBs are considered as zeros. Every 20th frames are only used to train batch intra prediction without classification. In Figure 4.10, the performance of batch intra prediction without classification is lower than batch intra prediction with classification but it is still significantly higher than that of H.264 intra prediction. The main reason for the loss of performance of batch intra prediction without classification is that the dimension of input features at the T_1 and T_2 classes is too low to predict well the whole sequence. In order to compensate for the low dimension, more frequently sampled training data are needed in batch intra prediction with classification. In Table 4.2, we compare the performance of batch intra prediction for the various sequences. The performance is highly dependent on video characteristics. The performance improvement significantly decreases from the low-motion video sequences such as NEWS and FOREMAN to

Table 4.2: Average Y-PSNR (dB) of intra prediction.

Sequence	H.264	Batch intra prediction w/o class
NEWS	21.79	30.62
FOREMAN	25.49	27.98
HARBOUR	21.50	25.56
CITY	25.79	26.49
MOBILE	18.12	18.60
CREW	28.13	28.48
FOOTBALL	24.11	24.35

the high-motion video sequences, for example, CREW and FOOTBALL. Thus, the performance improvement depends on nonstationarity of the video sequences. The high-motion sequences, which are more nonstationary, require more sample data for training to capture nonstationarity but here we use the same sampling period for all the sequences (every 20th frame). If we increase sampling rates for training, the training time and complexity of the batch intra prediction method increase. Therefore, we will consider online learning based intra prediction to compensate for nonstationarity of video sequences in section 4.5.

4.5 Intra Prediction through Online Learning

4.5.1 Online and Batch Learning

In spite of the solid theoretical foundations of statistical learning methods such as SVR [52] and Gaussian Process Regression (GPR) [71], they are usually used for batch learning which means that all the parameters are decided in the training time and are not updated during the test. Thus, batch learning requires a lot of training data and assumes that statistics of training data are similar to those of the test data. Because of nonstationary characteristics of video data, it is difficult to apply batch learning for real-time video compression. Even though training with regularly sampled data enables SVR to achieve the high prediction performance on the low-motion sequences as shown in section 4.4, there is a sort of sampling problem to catch nonstationarity. More training data are necessary

in order to encode high-motion video sequences. Therefore, we consider online learning which keeps updating learning parameters and SVs during the test. First, the authors of [56] extended SVR for online learning in the reproducing kernel Hilbert space for classification and regression using a stochastic gradient descent method:

$$\begin{aligned} f_{k+1}(\mathbf{x}) &= (1 - \frac{\eta_k}{C})(f_k(\mathbf{x}) - b) - \eta_k \ell'(y_k, f_k(\mathbf{x}_k))k(\mathbf{x}_k, \mathbf{x}) \\ &= \sum_{n=1}^{k} a_n k(\mathbf{x}_k, \mathbf{x}) + b \end{aligned} \tag{4.5}$$

where

$$\begin{cases} a_k = -\eta_k \ell'(y_k, f_k(\mathbf{x}_k)), & n = k \\ a_n = (1 - \frac{\eta_k}{C})a_n, & n < k \end{cases} \tag{4.6}$$

and $\ell'(y_k, f_k(\mathbf{x}_k))$ is a gradient of a loss function at $\mathbf{x}_k$. Here, we can see $\mathbf{x}_k$ is not a SV if a gradient of a loss function is zero. Otherwise, $\mathbf{x}_k$ becomes a SV. $(1 - \frac{\eta_k}{C})$ in (4.6) reflects aging effects to all the previous weight values a_n. Note that unlike batch learning updated a_n are not guaranteed to satisfy the KKT conditions [14]. In order to satisfy the KKT optimality conditions, references [57, 58] proposed accurate online SVR methods which update iteratively all the coefficients of SVs to satisfy the KKT conditions. However, they cannot guarantee a bound on the number of operations required per iteration. Consequently, they are not applicable for real-time applications. These kinds of online regressor [56–58] affect many or all the parameters by incrementally learning from a single new query point which is endangered by negative inference, that is, forgetting of useful knowledge while learning from new data [59]. In order to overcome these limitations, we will apply LWPR for incremental online learning in high dimensional spaces with selection of input features [55].

4.5.2 Locally Weighted Projection Regression

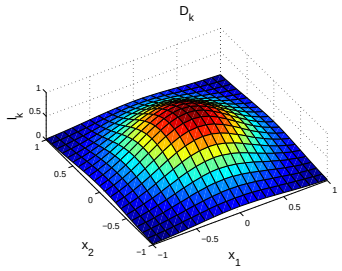
LWPR combines both Locally Weighted Learning (LWL) for localized learning [72] and Partial Least Squares (PLS) regression [73] for input dimensionality

reduction. A measure of locality (similarity) at each data point is computed from the Gaussian kernel:

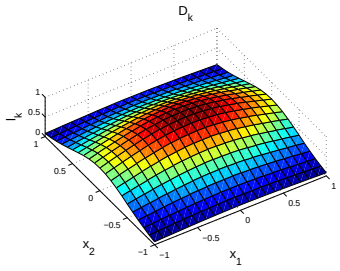
$$l_k(\mathbf{x}) = k(\mathbf{x}_k, \mathbf{x}) = e^{-0.5(\mathbf{x}-\mathbf{x}_k)^T \mathbf{D}_k (\mathbf{x}-\mathbf{x}_k)} \tag{4.7}$$

where $\mathbf{x}_k$ is a center of a Receptive Field (RF) which denotes validity of a local model (for simplicity, we call $\mathbf{x}_k$ a RF) and $\mathbf{D}_k$ is a positive semidefinite matrix that determines the size and shape of the local model. If we compare $l_k(\mathbf{x})$ with the RBF kernel function of SVR in (4.4), RFs are similar to SVs but RFs are not selected as a result of the explicit optimization like SVR. They are heuristically chosen based on locality, that is, if a test or training point $\mathbf{x}$ is not similar to the other RFs with threshold l_{gen}, it becomes a RF which is more adequate for online learning. If $\mathbf{D}_k$ is proportional to an identity matrix $2\gamma\mathbf{I}$ (isotropic matrix), the locality measure is equal to the RBF kernel function of SVR in (4.4). An isotropic matrix $\mathbf{D}_k$ gives an equal weight on input features so that it does not have input feature selection. Automatic Relevance Determination (ARD) [74] can be applied to find relevance of input features by means of the Bayesian inference in SVR but it only derives global relevance of input features for all the SVs. In [59], relevance of input features for each RF is incrementally determined by the stochastic gradient descent to minimize a penalized leave-one-out CV which is more applicable for online learning than ARD in SVM. Thus, each RF has its own shape $\mathbf{D}_k$ and is updated by seeing a new query point. For example, Figure 4.11 illustrates the relation between the locality $l_k(\mathbf{x})$ and the distance metric matrix $\mathbf{D}_k$ in two dimensional input features $\mathbf{x}$ where the center of a RF $\mathbf{x}_k$ is $(0, 0)$. Figure 4.11 (a) shows the same locality measure in all the directions in the isotropic matrix (a) in Figure 4.11 (d). The diagonal matrix (b) in Figure 4.11 (d) gives different locality to axis as shown in Figure 4.11 (b). The general matrix (c) in Figure 4.11 (d) achieves different locality in any direction in Figure 4.11 (c). Larger d in the distance matrix $\mathbf{D}_k$ which corresponds to smaller variance of the Gaussian kernel makes a RF $\mathbf{x}_k$ more localized.

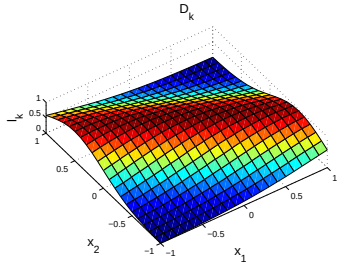
In LWL [72], the output of a learning system $\hat{y}$ is the normalized weighted mean of training data y_k for a given test point $\mathbf{x}$. It minimizes the weighed least



(a) Isotropic $\mathbf{D}_k$



(b) Diagonal $\mathbf{D}_k$



(c) General $\mathbf{D}_k$

$$\begin{bmatrix} d & 0 \\ 0 & d \end{bmatrix} \quad \begin{bmatrix} d_{11} & 0 \\ 0 & d_{22} \end{bmatrix} \quad \begin{bmatrix} d_{11}d_{12} \\ d_{21}d_{22} \end{bmatrix}$$

(a) (b) (c)

(d) $\mathbf{D}_k$

Figure 4.11: Locality $\ell_k(\mathbf{x})$ w.r.t. the distance matrix $\mathbf{D}_k$.

squared loss $L(\mathbf{x}) = \sum_{k=1}^{\#RF} l_k(\mathbf{x})(\hat{y} - y_k)^2$:

$$\hat{y}(\mathbf{x}) =^{LWL} \frac{\sum_{k=1}^{\#RF} l_k(\mathbf{x}) y_k}{\sum_{k=1}^{\#RF} l_k(\mathbf{x})} =^{LWPR} \frac{\sum_{k=1}^{\#RF} l_k(\mathbf{x}) \hat{y}_k(\mathbf{x})}{\sum_{k=1}^{\#RF} l_k(\mathbf{x})} \tag{4.8}$$

However, y_k is only an available value at corresponding training data $\mathbf{x}_k$. It is better to estimate $y(\mathbf{x})$ at $\mathbf{x}_k$ instead of y_k if $\mathbf{x} \neq \mathbf{x}_k$. The estimated $y(\mathbf{x})$ at $\mathbf{x}_k$, that is, $\hat{y}_k(\mathbf{x})$ is linearly estimated at each RF with locality information [72]. If $\mathbf{x}$ is quite different from $\mathbf{x}_k$ and the relation between input and output changes slowly, $\hat{y}_k(\mathbf{x})$ is also different from the corresponding true output of $\mathbf{x}$. Therefore, the locality $l_k(\mathbf{x})$ prevents $\hat{y}_k(\mathbf{x})$ from affecting the final estimated output $\hat{y}(\mathbf{x})$ in (4.8).

LWL was further extended for high dimensional input space. The authors of [55] applied the PLS regression for the output of RFs $\hat{y}_k(\mathbf{x})$ among the projection methods such as Principle Component Analysis (PCA) and Factor Analysis (FA). The projection methods can prevent overfitting at each RF as the ϵ-incentive loss function of SVR does in addition to excluding irrelevant input features. The output

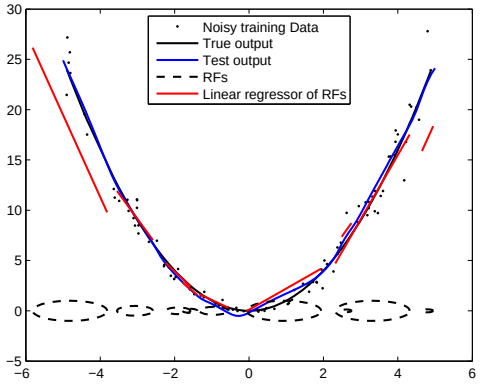


Figure 4.12: Example of LWPR.

$\hat{y}(\mathbf{x})$ is quite similar to the output of SVR in (4.3) but here $l_k(\mathbf{x})$, $\hat{y}_k(\mathbf{x})$ and the number of RFs are updated for online learning after seeing a test point $\mathbf{x}$.

4.5.3 Partial Least Squares Regression

In order to reduce dimensionality of input features, the PCA regression which constructs linear projections of data to preserve characteristics of input data is considered. However, PCA regression does not guarantee that principal components which preserve input data describe well output data. In regression problems, we need to find principal components which represent input data as well as output data. In this sense, the PLS regression is an efficient method to maximize covariance projected data $\mathbf{Xu}$ and weighted output $\mathbf{Wy}$ at each RF as follows [73]:

$$\max_{\mathbf{u}} cov(\mathbf{Xu}, \mathbf{Wy}), \quad s.t.\ \mathbf{u^T u} = 1 \tag{4.9}$$

where $\mathbf{X}$ is a matrix whose row represents input features $\mathbf{x}_i^T$, and $\mathbf{W}$ is a diagonal matrix of the local weight $l_k(\mathbf{x}_i)$ in (4.7), and a vector $\mathbf{y}$ consists of outputs y_i which are associated with inputs $\mathbf{x}_i$. In particular, each local model of RFs is achieved by regression to the weighted output $\mathbf{Wy}$ [55]. The optimal solution $\mathbf{u}$ in (4.9) is $\frac{\mathbf{X^T W y}}{\mathbf{y^T W^T X X^T W y}}$ which is derived in Appendix A.4.

Next, we find a scalar β to minimize the weighted least squares loss:

$$\min_{\beta} (\mathbf{y} - \mathbf{X}\mathbf{u}\beta)^{\mathbf{T}}\mathbf{W}(\mathbf{y} - \mathbf{X}\mathbf{u}\beta) \tag{4.10}$$

The solution of the above equation β is $\frac{(\mathbf{X}\mathbf{u})^{\mathbf{T}}\mathbf{W}\mathbf{y}}{(\mathbf{X}\mathbf{u})^{\mathbf{T}}\mathbf{W}\mathbf{X}\mathbf{u}}$. Finally, these operations are performed iteratively to decide the adequate number of projections at each RF. If the number of projections is smaller than the dimension of input features, it prevents overfitting to training data and it removes effects of irrelevant input features on the output by means of PLS. At each iteration, true output is deflated as follows: $\mathbf{y}_r = \mathbf{y}_{r-1} - \mathbf{z}_r\beta_r$ where r is a iteration index and $\mathbf{z}_r = \mathbf{X}_{r-1}\mathbf{u}_r$. $\mathbf{X}$ is also deflated to generate orthogonal (conjugate) projections with respect to $\mathbf{W}$ at each iteration:

$$\mathbf{X}_r = \mathbf{X}_{r-1} - \mathbf{z}_r\mathbf{p}_r^T \tag{4.11}$$

where $\mathbf{p}_r = \frac{\mathbf{X}_{r-1}^T\mathbf{W}\mathbf{z}_r}{\mathbf{z}_r^T\mathbf{W}\mathbf{z}_r}$ derived in Appendix A.5. The number of projections at each RF increases until decrease of MSE is smaller than the specified threshold. Interested readers can also refer to [55, 59, 73] for more information.

As an example, we show how locally linear functions can represent a globally nonlinear function. Figure 4.12 illustrates linear regressors which learn from noisy training data using LWPR to predict a quadratic nonlinear function. 100 training data are generated from $x^2 + \epsilon$ where $-5 \leq x \leq 5$ and ϵ is random noise. After training, 9 RFs are selected and the size of the distance metric is denoted as a dashed line at each RF. Each RF has its own linear regressor which is only fitted to the local region $l_k(\mathbf{x})$ in (4.7) by means of PLS and all the RFs cover overall areas. Note that there is overlap among the RFs because the Gaussian function of $l_k(\mathbf{x})$ decreases smoothly locality as shown in Figure 4.11. Even if LWPR learns from noisy data, test outputs are very close to true outputs with small RFs in Figure 4.12.

Here, we summarize PLS for incremental update from a test $\mathbf{x}$. In (4.12), the mean of the kth RF is subtracted from $\mathbf{x}$ and then $\mathbf{x}$ is projected onto $\mathbf{u}_r$ and deflated by $\mathbf{p}_r$ in (4.13). The estimated output $\hat{y}_k$ is obtained from (4.14). Thus, the accuracy of the predicted output $\hat{y}_k$ at the kth RF depends on the number of

projections.

For prediction at the kth RF:

$$\hat{y}_k = \beta_0, \quad \mathbf{x} = \mathbf{x} - \bar{\mathbf{x}}_\mathbf{k} \tag{4.12}$$

for $r = 1 : \#\text{projections}$

$$z_r = \mathbf{u}_r^T \mathbf{x}, \quad \mathbf{x} = \mathbf{x} - z_r \mathbf{p}_r \tag{4.13}$$

$$\hat{y}_k = \hat{y}_k + \beta_r z_r \tag{4.14}$$

end

After seeing a test $\mathbf{x}$, the parameters β_r, $\mathbf{u}_r$ and $\mathbf{p}_r$ of the kth RF are updated as follows:

$$Ł_k = \tau L_k, \quad l_k^0 = l_k(\mathbf{x}) \tag{4.15}$$

$$\bar{\mathbf{x}}_\mathbf{k} = \frac{L_k \bar{\mathbf{x}}_\mathbf{k} + l_k^0 \mathbf{x}_k}{L_k + l_k^0}, \quad \beta_0 = \frac{L_k \beta_0 + l_k^0 \tilde{y}}{L_k + l_k^0} \tag{4.16}$$

$$L_k = L_k + l_k^0, \quad \mathbf{x} = \mathbf{x} - \bar{\mathbf{x}}_\mathbf{k}, \quad \tilde{y} = \tilde{y} - \beta_0 \tag{4.17}$$

for $r = 1 : \#\text{projections}$

$$\mathbf{s}_{xy}^r = \tau \mathbf{s}_{xy}^r + l_k^0 \tilde{y} \mathbf{x} \tag{4.18}$$

$$\mathbf{u}_r = \frac{\mathbf{s}_{xy}^r}{\| \mathbf{s}_{xy}^r \|_2}, \quad z_r = \mathbf{u}_r^T \mathbf{x} \tag{4.19}$$

$$s_{zz}^r = \tau s_{zz}^r + l_k^0 z_r^2, \quad s_{yz}^r = \tau s_{yz}^r + l_k^0 z_r \tilde{y}, \quad \mathbf{s}_{xz}^r = \tau \mathbf{s}_{xz}^r + l_k^0 z_r \mathbf{x} \tag{4.20}$$

$$\beta_r = \frac{s_{yz}^r}{s_{zz}^r}, \quad \mathbf{p}_r = \frac{\mathbf{s}_{xz}^r}{s_{zz}^r} \tag{4.21}$$

$$\mathbf{x} = \mathbf{x} - z_r \mathbf{p}_r, \quad \tilde{y} = \tilde{y} - \beta_r z_r \tag{4.22}$$

end

where τ represents aging for previous information and L_k is the sum of locality at the kth RF. The average input $\bar{\mathbf{x}}_\mathbf{k}$ and the average output β_0 are updated from the weighted average in (4.16). The projection direction $\mathbf{u}_r$ is incrementally updated in (4.18) and (4.19) as a result of (4.9). β_r and $\mathbf{p}_r$ are incrementally updated in (4.20) and (4.21) from (4.10) and (4.11). The sufficient information $L_k, \bar{\mathbf{x}}_\mathbf{k}, \mathbf{s}_{xy}, \mathbf{s}_{zz}, \mathbf{s}_{yz}$ and $\mathbf{s}_{xz}$ are saved for the next update. Note that reconstructed

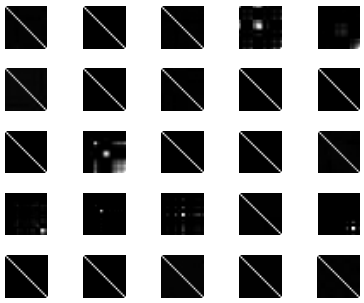


Figure 4.13: General distance matrices $\mathbf{D}_k$ of receptive fields.

pixels $\tilde{y}$ are used for update instead of original pixels y to remove the drift problem at the decoder.

4.5.4 Learning Distance Matrix

The distance matrix $\mathbf{D}_k$ is important to decide the shape and range of a RF as shown in Figure 4.11. Online intra prediction which will be discussed in subsection 4.5.5 has high dimensional input features according to adjacent neighboring pixels. In order to reduce the complexity of learning the distance matrix, we only consider a diagonal distance matrix since many distance matrices have strong diagonal elements which is illustrated in Figure 4.13. The distance matrix $\mathbf{D}_k$ is updated to minimize the regularized leave-one-out CV as follows [55, 59]:

$$J_k = \frac{1}{\sum_{i=1}^{M} l_k(\mathbf{x}_i)} \sum_{i=1}^{M} \frac{l_k(\mathbf{x}_i)(y_i - \hat{y}_k(\mathbf{z}_i))^2}{(1 - l_k(\mathbf{x}_i)\mathbf{z}_i^T \mathbf{P}_z \mathbf{z}_i)^2} + \frac{1}{NC} \sum_{i,j=1}^{N} D_{k_{ij}}^2$$

where $\mathbf{P}_z = (\mathbf{Z}^T\mathbf{Z})^{-1}$ and $\mathbf{z}_i$ are projected input features discussed in subsection 4.5.3, and $l_k(\mathbf{x}_i)$ in (4.7) is a locality measure between $\mathbf{x}_i$ and $\mathbf{x}_k$. M, N and k are the number of training data, input dimensions and an index of a RF, respectively. The sum of elements of the distance matrix is penalized with a regularization parameter $\frac{1}{C}$ in order to prevent RFs from shrinking. The minimization is achieved

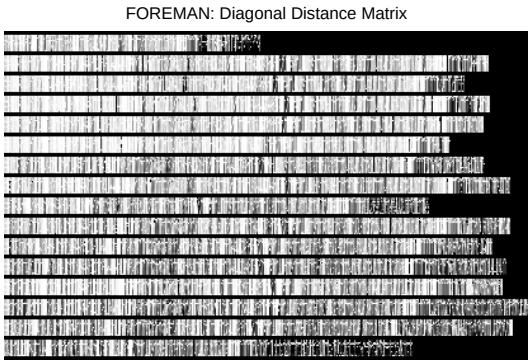


Figure 4.14: Diagonal distance matrices $\mathbf{D}_k$ of receptive fields.

by the gradient descent with learning rate η_k^r:

$$\mathbf{M}_k^{r+1} = \mathbf{M}_k^r - \eta_k^r \cdot \frac{\partial J_k}{\partial \mathbf{M}_k}, \quad \mathbf{D}_k = \mathbf{M}_k^T \mathbf{M}_k \tag{4.23}$$

where $\mathbf{M}_k$ is an upper triangular matrix resulting from the Cholesky decomposition of $\mathbf{D}_k$. In order to increase learning speed, the Stochastic Meta Descent (SMD) algorithm [75] is applied to adapt a gradient-step size as follows (RF index k is omitted for simplicity):

$$\eta^r = \eta^{r-1} \cdot \max\left(\frac{1}{2}, 1 - \mu \frac{\partial J}{\partial \mathbf{M}} v^r\right) \tag{4.24}$$

where

$$v^r = \tau v^{r-1} - \eta^{r-1}\left(\frac{\partial J}{\partial \mathbf{M}} + \tau \mathbf{H}^{r-1} v^{r-1}\right) \tag{4.25}$$

and μ and τ are a meta-step size and a decay factor, respectively. $\mathbf{H}$ is the Hessian of $J(\mathbf{M})$ with respect to $\mathbf{M}$. Thus, the SMD algorithm adjusts step sizes in the log space and optimizes over an exponentially decaying trace of gradients. More detailed formulations are described in Appendix A.6.

An example of the distance matrices $\mathbf{D}_k$ is shown in Figure 4.14 where each column represents a normalized diagonal distance matrix (diagonal elements only) of a RF. The horizontal black lines in Figure 4.14 separate RFs of 16 LWPRs which

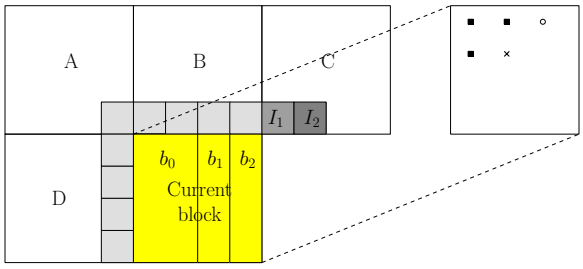


Figure 4.15: Input features of online intra prediction.

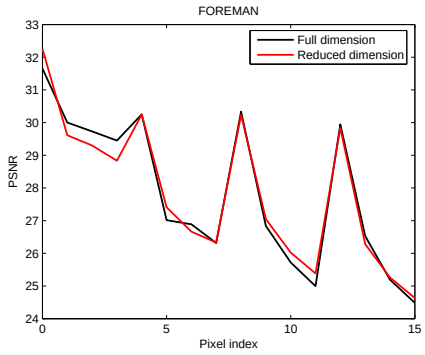


Figure 4.16: Prediction performance vs. Dimension of input features.

are associated with 16 pixels of the 4x4 MBs in online intra prediction which will be explained in subsection 4.5.5. Figure 4.14 indicates that each LWRP regressor has the different number of RFs which corresponds to the number of the columns, and the shapes of the diagonal distance matrices are different, that is, each column has various grey colors from white to black which corresponds from a large diagonal element to a small diagonal element.

4.5.5 Online Learning based Intra Prediction

We apply LWPR [55] for online learning based intra prediction in the pixel domain in order to reduce the dimension of input features. Input features of regressors are adjacent neighboring pixels of 4x4 MBs which are denoted as grey pixels in Figure 4.15 instead of DCT coefficients of batch intra prediction in subsection

4.3.3. These input features are similar to the neighboring pixels of 4x4 blocks in H.264 as shown in Figure 4.1. In order to reduce input features, pixels of a current MB have different dimensions. Input features of the b_0 pixels in a current MB are the light grey pixels in Figure 4.1. Input features of the b_1 pixels are increased by concatenating the I_1 pixel to the light grey pixels. Input features of the b_2 pixels are extended by concatenating the I_2 pixel to input features of the b_1 pixels. Because of intra prediction in the pixel domain, incremental intra prediction with the inverse zig-zag scan order cannot be applied with the single-pass DCT transform. If neighboring pixels are not available, corresponding values are considered as 128. In Figure 4.16, we compare the performance of intra prediction with two different input features: input features denoted as the full dimension are the neighboring pixels of 4x4 blocks in H.264 as shown in Figure 4.1, and input features denoted as the reduced dimension are the pixels in Figure 4.15. The reduced dimension online learning achieves similar performance compared to the full dimension as shown in Figure 4.16. Note that pixel indices are assigned by row-wise numbering in a 4x4 block.

In online learning based intra prediction, we use two prediction modes: LWPR using adjacent neighboring pixels as in Figure 4.15 and a spatial average method, that is, a local average of predicted values. As an example of Figure 4.15, the cross pixel of a current MB can be predicted using LWPR or an average of predicted values which are marked as the square pixels that correspond to the left, left top and top predicted values of a current pixel. Note that the square pixels are not reconstructed pixels since a current MB is being processed and therefore DCT, quantization and inverse quantization cannot be performed before obtaining all the prediction errors of a current MB. However, if there are available reconstructed pixels for the left, left top and top pixels, reconstructed pixels are used instead of predicted pixels. For example, the first pixel (left top) of a current MB is spatially predicted from an average of reconstructed pixels. In this case, we need a flag to indicate which method is used to choose better prediction.

The first row of a frame which is indicated as T_1 and T_2 in Figure 4.5 is differently processed. In the T_1 class, we build separate LWPR for 2x2 blocks,

that is, the first 4x4 MB is subdivided into 2x2 blocks and 2x2 block pixels are predicted from adjacent neighboring pixels of 2x2 blocks. Next, a 2x2 DCT is applied to prediction residuals of 2x2 blocks and then 2x2 blocks are reconstructed after quantization, inverse quantization and inverse DCT. The reconstructed pixels can be used for input features of LWPR. Even though a 4x4 DCT better condenses energy than four 2x2 DCT, we apply a 2x2 DCT instead of a 4x4 DCT for the first MB to reconstruct 2x2 blocks. Prediction errors of the first MB are usually larger than the other MBs since the online learning method cannot learn well without meaningful input features (all the input features are 128). Therefore, we build separate regressors for the first MB which are assigned for 2x2 blocks. In the T_2 class, we separately build LWPR with input features extended since left 4 pixels are only available for input features. Let the left 4 pixels be p_k where $k = 0, 1, 2, 3$. Pixel difference $g_k = p_k - p_{k+1}$ and $g_k - g_{k+1}$ are concatenated to the left 4 pixels for 9 dimensional input features.

A new RF is added if any current RFs are not activated, that is, if the locality l_k of (4.7) is smaller than a threshold l_{gen}. A generation number is assigned to the new RF to remove the oldest RF when the number of RFs reaches the maximum limitation. The new added RF is activated after $\mathbf{D}_k$ is updated 5 times in order to decide its size or shape. If a new RF is added, a predicted value of the regressor is not accurate since no RFs match with a test sample $\mathbf{x}$. In this case, an average of left, left top and top reconstructed or predicted pixels is mainly selected for intra prediction.

A RF is removed in order to manage the number of RFs. Pruning of a RF is occurred if the locality (4.7) of two RFs are too close at the query point or the number of RFs reaches the maximum. If RFs overlap too much with the other RFs, a RF with a larger determinant of $\mathbf{D}_k$ in (4.7) is pruned as in [59]. This corresponds to removing a RF with a smaller shape among the overlapped RFs to reduce effects to the regressor output. In the other case, the least frequently activated RF is removed among the oldest RFs as indicated by the generation number. Pruning of old or overlapped RFs and adding of new RFs occur during encoding such that regressors maintain the prediction performance of RFs for nonstationary

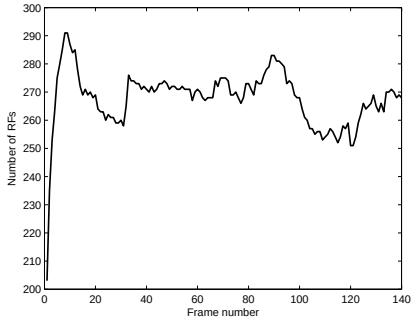


Figure 4.17: Number of receptive fields.

video sequences. Figure 4.17 shows that the number of RFs saturates after several frames. However, it does not mean that all the RFs are fixed because the locality l_k is updated with the distance matrix learning as discussed in subsection 4.5.4, and pruning and adding RFs happens during encoding.

Here, we summarize the overall procedure of online learning based intra prediction.

Training procedure:

1. Decide RFs from training data based on the locality $l_k(\mathbf{x}) < l_{gen}, \forall k$.

2. If training data $\mathbf{x}$ activates RFs ($l_k(\mathbf{x}) > l_{act}$), the activated RFs whose center are $\mathbf{x}_k$ update $\mathbf{D}_k$ as in (4.23) and the number of projections as in subsection 4.5.3.

3. Repeat the procedures 1 and 2 until the number of iterations reaches the maximum.

Testing procedure:

1. Prepare input features $\mathbf{x}$ from adjacent neighboring pixels as shown in Figure 4.15. For the first row MBs, the extended input features are used as explained in subsection 4.5.5. If there are no available neighboring pixels, corresponding input features are considered as 128.

2. Learning based intra prediction is obtained from the equations (4.8), (4.12) and (4.14) for a given $\mathbf{x}$ and RFs $\mathbf{x}_k$, and spatial average prediction is calculated from left, left top and top predicted or reconstructed pixels.

3. Decide a prediction method to minimize prediction errors.

4. Perform a 4x4 DCT and quantization to prediction errors and code quantization indexes. The first MB in a frame is partitioned into 4 blocks, and a 2x2 DCT instead of a 4x4 DCT is applied to reconstruct 4 pixels in a block as discussed in subsection 4.5.5.
5. Reconstruct pixels in a MB by adding predicted pixels to reconstructed prediction errors after inverse quantization and inverse 4x4 DCT.
6. Update RFs from input features $\mathbf{x}$ and reconstructed pixels $\tilde{y}$.
6.1. Do the following procedure from $k = 1$ to $\#RF$.
6.1(a). Calculate the locality (activity) $l_k(\mathbf{x})$ in (4.7).
6.1(b). If the locality is smaller than threshold l_{act}, go to 6.1
6.1(c). Update projections and regressions of PLS as in subsection 4.5.3.
6.1(d). Update the distance matrix of a RF $\mathbf{D}_k$ in (4.23).
7. If the maximum activity is smaller than the threshold l_{gen}, create a new RF from input features $\mathbf{x}$.
8. If two RFs overlap too much or the number of RFs reaches the maximum number of RFs, prune a RF as discussed in subsection 4.5.5.

4.6 Experimental Results of Online Intra Prediction

In contrast to batch learning based intra prediction, we only use the first frame for training and then test 150 frames of the FOREMAN sequence. After training, the shapes of RFs $\mathbf{D}_k$ in (4.7) which decide relevance of input features are illustrated in Figure 4.14. Prediction Y-PSNR (dB) of online intra prediction at each frame is shown in Figure 4.18. There is only one peak which is different from batch intra prediction in Figure 4.10 since only the first frame is used for training. Table 4.3 shows average Y-PSNR (dB) values of the various sequences. The prediction performance of online learning based intra prediction decreases from the low-motion sequences (NEWS and FOREMAN) to the high-motion sequences (CREW and FOOTBALL). However, the variations of online intra prediction are smaller than those of batch intra prediction as seen from comparing Tables 4.2 and

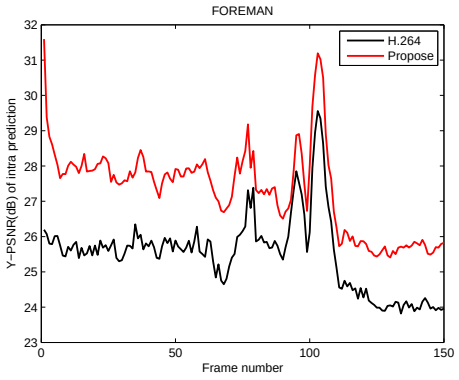


Figure 4.18: Y-PSNR (dB) of online learning based intra prediction.

Table 4.3: Average Y-PSNR (dB) of intra prediction.

Sequence	H.264	Online intra prediction
NEWS	21.79	26.87
FOREMAN	25.49	27.32
HARBOUR	21.50	23.92
CITY	25.79	26.98
MOBILE	18.12	19.60
CREW	28.13	28.99
FOOTBALL	24.11	25.24

4.3.

4.7 Conclusion

In this chapter, we proposed new video coding methods for intra prediction using learning methods such as SVR and LWPR to improve the performance of H.264 intra prediction by means of batch and online learning. From experimental results, the performance of intra prediction can be significantly improved by the proposed methods. Even though batch learning can achieve better results than online learning for slow-motion sequences, batch learning requires a lot more time for training than online learning. Furthermore, the more active video sequences, the more training data are required to compensate for the nonstationarity. Conse-

quently, online learning is much more promising for real-time video compression. Future research topics include how to increase learning rate and reduce tuning parameters and complexity of regressors while maintaining the performance for real-time applications.

4.8 Acknowledgement

Portions of the text of this chapter are adapted from material to be published as: C. An and T. Q. Nguyen, "Statistical Learning based Intra Prediction in H.264", in *Proceedings of the IEEE International Conference on Image Processing*, ICIP-2008, San Diego, CA, USA, Oct., 2008. The dissertation author was the primary researcher for this publication, and the co-author directed and supervised the research which forms the basis for this chapter.

5 Summary and Extensions of Contributions

In this dissertation, video coding problems were solved by the optimization and learning methods since the RD optimization problem and intra prediction in video coding are highly related with numerical optimization and regression theory. We presented how to apply optimization and learning methods for video coding problems and showed some possibility to improve the performance of video coding.

5.1 Optimization based Video Coding

We proposed the general framework to solve the RD optimization problem using the primal-dual decomposition and subgradient projection methods. As a result, we derived optimality conditions among the Lagrange multipliers λ with spatial or temporal prediction dependency. Furthermore, we reused the reference software model to solve the dependent optimization problem. Even though this method requires iterations with high complexity, it can be a reference model to improve and to compare the performance of the current RC algorithm. The proposed method can be applied to other video coding problems such as inter-layer or quality-layer bit allocation problems in scalable video coding. We also derived the optimality conditions of the Lagrange multipliers λ for the optimal bit allocation. These conditions can be satisfied by two steps: The Lagrange multipliers of pictures are derived without considering temporal prediction dependency and then temporal dependency is considered to assign different weights to the Lagrange mul-

tipliers among pictures. For the first step, we applied the CM algorithm to adapt λ for a given QP. In order to remove iterations of the CM algorithm, the incremental CM algorithm was proposed with similar performance. Even though the performance improvement was very limited from the experimental results, the proposed method is a general solution to derive λ for any video coding. The performance improvement is larger for highly compressed video sequences since residual bits are smaller and side bits are larger. Thus, the proposed method will be more useful in future video coding which aims to reduce residuals with increased side bits (complexity). Even though temporal dependency was not adaptively derived in this dissertation, the proposed method will be useful to be combined with temporal prediction dependency.

5.2 Learning based Video Coding

In this work, we first applied the online and batch learning methods for video coding. Even though batch learning based intra prediction improved significantly the performance of intra prediction at the low-motion sequences, it was difficult to apply batch intra prediction for the high-motion sequences. In order to adapt nonstationarity of video sequences, we surveyed online learning methods such as online SVR and accurate online SVR. Because of high correlation of local areas in video sequences, we applied locally weighted learning for online intra prediction with local updates. Furthermore, relevance of input features is decided by stochastic gradient descent learning. As a result, online learning based intra prediction achieved larger improvement than the current H.264 standard in the various sequences. In this dissertation, the proposed method was only applied for intra prediction but it can be extended for inter prediction.

A Appendix

A.1 Derivation of subgradient

A subgradient of f at x is any scalar g that satisfies the inequality $f(y) \geq f(x)+g(y-x), \forall y$. If f is differential and a convex function, the unique subgradient g is $f'(x)$ which is the gradient decent method.

A.1.1 Subgradient of $q_n(\lambda_n, y_n)$

It is reformulated from [6] for the thesis's notation.

$$\begin{aligned} q_n(\tilde{\lambda}_n, y_n) &= \min_{x_n}\{d(x_n) + \tilde{\lambda}_n(x_n - y_n)\} \\ &\leq d(\hat{x}_n) + \tilde{\lambda}_n(\hat{x}_n - y_n) \\ &= d(\hat{x}_n) + \hat{\lambda}_n(\hat{x}_n - y_n) + (\hat{x}_n - y_n)(\tilde{\lambda}_n - \hat{\lambda}_n) \\ &= q_n(\hat{\lambda}_n, y_n) + (\hat{x}_n - y_n)(\tilde{\lambda}_n - \hat{\lambda}_n), \quad \forall \tilde{\lambda}_n \end{aligned}$$

where $\hat{x}_n = \arg\min_{x_n}\{d(x_n) + \hat{\lambda}_n(x_n - y_n)\}$. Thus, the subgradient of $q_n(\lambda_n, y_n)$ at $\hat{\lambda}_n$ is $\hat{x}_n - y_n$.

A.1.2 Subgradient of $q_n^*(y_n)$

It is reformulated from [6] for this paper's notation.

$$\begin{aligned} q_n^*(\tilde{y}_n) &= \max_{\lambda_n \geq 0} q_n(\lambda_n, \tilde{y}_n) \\ &= \max_{\lambda_n \geq 0} \min_{x_n} \{d(x_n) + \lambda_n(x_n - \tilde{y}_n)\} \\ &\geq \min_{x_n} \{d(x_n) + \hat{\lambda}_n(x_n - \tilde{y}_n)\} \\ &= \min_{x_n} \{d(x_n) + \hat{\lambda}_n(x_n - \hat{y}_n) - \hat{\lambda}_n(\tilde{y}_n - \hat{y}_n)\} \\ &= q_n^*(\hat{y}_n) - \hat{\lambda}_n(\tilde{y}_n - \hat{y}_n), \forall \tilde{y}_n \end{aligned}$$

where $\hat{\lambda}_n = \arg\max_{\lambda_n} \min_{x_n} \{d(x_n) + \lambda_n(x_n - \hat{y}_n)\}$. The subgradient of optimal dual function $q_n^*(y)$ at $\hat{y}_n$ is $-\hat{\lambda}_n$.

A.1.3 Subgradient of $\sum_f Q_f^*(y_f, y_{ref}^f)$

$$\sum_f Q_f^*(\tilde{y}_f, \tilde{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) = \sum_f \max_{\lambda_f \geq 0} \min_{\mathbf{s_f}} \left\{ D_f(\mathbf{s_f}, \tilde{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) + \lambda_f \left(X_f(\mathbf{s_f}, \tilde{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) - \tilde{y}_f \right) \right\}$$

$$\geq \sum_f \min_{\mathbf{s_f}} \left\{ D_f(\mathbf{s_f}, \tilde{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) + \hat{\lambda}_f \left(X_f(\mathbf{s_k}, \tilde{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) - \tilde{y} \right) \right\}$$

$$= \sum_f \min_{\mathbf{s_f}} \left\{ D_f(\mathbf{s_f}, \hat{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) + \hat{\lambda}_f \left(X_f(\mathbf{s_f}, \hat{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) - \hat{y}_f \right) - \hat{\lambda}_f(\tilde{y}_f - \hat{y}_f) \right.$$

$$\left. + D_f(\mathbf{s_f}, \tilde{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) - D_f(\mathbf{s_f}, \hat{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) + \quad \hat{\lambda}_f \left(X_f(\mathbf{s_f}, \tilde{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) - X_f(\mathbf{s_f}, \hat{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) \right) \right\}$$

$$= \sum_f \left\{ Q_f^*(\hat{y}_f, \hat{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) - \hat{\lambda}_f(\tilde{y}_f - \hat{y}_f) + \min_{\mathbf{s_f}} \left\{ D_f(\mathbf{s_f}, \tilde{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) - D_f(\mathbf{s_f}, \hat{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) \right.\right.$$

$$\left.\left. + \hat{\lambda}_f \left(X_f(\mathbf{s_f}, \tilde{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) - X_f(\mathbf{s_f}, \hat{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) \right) \right\} \right\}$$

$$= \sum_f \left\{ Q_f^*(\hat{y}_f, \hat{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) - \hat{\lambda}_f(\tilde{y}_f - \hat{y}_f) + \min_{\mathbf{s_f}} \left\{ L_f(\mathbf{s_f}, \tilde{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) - L_f(\mathbf{s_f}, \hat{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) \right\} \right\}$$

$$\geq \sum_f \left\{ Q_f^*(\hat{y}_f, \hat{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) - \hat{\lambda}_f(\tilde{y}_f - \hat{y}_f) + \min_{\mathbf{s_f}} \sum_i^R L_f'(\mathbf{s_f}, \hat{y}_{ref_i}^f)(\tilde{y}_{ref_i}^f - \hat{y}_{ref_i}^f) \right\}$$

$$\geq \sum_f \left\{ Q_f^*(\hat{y}_f, \hat{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) - \hat{\lambda}_f(\tilde{y}_f - \hat{y}_f) + \sum_i^R \min_{\mathbf{s_f}} L_f'(\mathbf{s_f}, \hat{y}_{ref_i}^f)(\tilde{y}_{ref_i}^f - \hat{y}_{ref_i}^f) \right\}$$

where $L_f(\mathbf{s_f}, \mathbf{y}_{\mathbf{ref}}^{\mathbf{f}}) = D_f(\mathbf{s_f}, \mathbf{y}_{\mathbf{ref}}^{\mathbf{f}}) + \hat{\lambda}_f X_f(\mathbf{s_f} \mathbf{y}_{\mathbf{ref}}^{\mathbf{f}})$, $\mathbf{y}_{\mathbf{ref}}^{\mathbf{f}} \in \{\hat{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}} \tilde{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}\}$ and $L_f'(\mathbf{s_f}, \hat{y}_{ref_i}^f) = \left. \frac{\partial L_f(\mathbf{s_f}, y_{ref_i}^f)}{\partial y_{ref_i}^f} \right|_{y_{ref_i}^f = \hat{y}_{ref_i}^f}$ and $\hat{\lambda}_f = \arg\max_{\lambda_f \geq 0} \min_{\mathbf{s_f}} \left\{ D(\mathbf{s_f}, \hat{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) + \lambda_f \left(X_f(\mathbf{s_f}, \hat{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) - \hat{y}_f \right) \right\}$ and R is the number of reference frames. In this paper, bits of the reference frames of B picture $\mathbf{y}_{\mathbf{ref}}^{\mathbf{f}}$ are y_1 and y_F and the reference frame bit of P picture is

y_1 from Figure 2.15. Consequently, the following equation is derived

$$\begin{aligned}
&\sum_{f=1}^{F} Q_f^*(\tilde{y}_f, \tilde{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) \\
&\geq \sum_{f=1}^{F} Q_f^*(\hat{y}_f, \hat{\mathbf{y}}_{\mathbf{ref}}^{\mathbf{f}}) - \left(\hat{\lambda}_F - \sum_{f=2}^{F-1} \min_{\mathbf{s_f}} L'_f(\mathbf{s_f}, \hat{y}_F)\right)(\tilde{y}_F - \hat{y}_F) \\
&- \sum_{f=2}^{F-1} \hat{\lambda}_f(\tilde{y}_f - \hat{y}_f) - \left(\hat{\lambda}_1 - \sum_{f=2}^{F} \min_{\mathbf{s_f}} L'_f(\mathbf{s_f}, \hat{y}_1)\right)(\tilde{y}_1 - \hat{y}_1)
\end{aligned}$$

A.2 Projection onto the feasible constraint set

The equation (2.55) is reformulated as follows:

$$\min_{\mathbf{y}} \frac{1}{2} \parallel \tilde{\mathbf{y}} - \mathbf{y} \parallel_2^2, \quad s.t. \;\; \mathbf{y}^T\mathbf{1} \leq X, \quad \mathbf{y} \succeq 0$$

where $\mathbf{1}$ is a vector whose elements are 1.

$$\begin{aligned}
L &= \frac{1}{2}(\tilde{\mathbf{y}} - \mathbf{y})^T(\tilde{\mathbf{y}} - \mathbf{y}) + \mu(\mathbf{y}^T\mathbf{1} - X) \\
\nabla_{\mathbf{y}} L &= \tilde{\mathbf{y}} - \mathbf{y} + \mu\mathbf{1} = 0, \quad \mathbf{y} = \tilde{\mathbf{y}} - \mu\mathbf{1} \qquad \text{(A.1)} \\
\frac{\partial L}{\partial \mu} &= \mathbf{y}^T\mathbf{1} - X = 0, \quad \mathbf{y}^T\mathbf{1} = X \\
X &= (\tilde{\mathbf{y}} - \mu\mathbf{1})^T\mathbf{1} = \sum_{n=1}^{N} \tilde{y}_n - \mu N \\
\mu &= \frac{\sum_{n=1}^{N} \tilde{y}_n - X}{N} \qquad \text{(A.2)}
\end{aligned}$$

The algorithm 1 in [76] was derived from the above results: from the complementary slackness condition [14], $\mu = 0$ if $\sum_n \tilde{y}_n < X$ where $\tilde{\mathbf{y}} = \mathbf{y}$ from (A.1) or $\mu = 0$ from (A.2) if $\sum_n \tilde{y}_n = X$. Thus, $\mu = 0$ if $\sum_n \tilde{y}_n \leq X$ and then $\mathbf{y} = \tilde{\mathbf{y}}$ from (A.1). If $\mu > 0$, $\sum_n y_n = X$ and $\mathbf{y} = \tilde{\mathbf{y}} - \mu\mathbf{1} \succeq 0$.

The algorithm 1 is summarized as follows: $S = \{1, ..., N\}$

$$\text{if} \quad \sum_{n=1}^{N} \tilde{y}_n \leq X,$$

$$\mathbf{y} = \tilde{\mathbf{y}}$$

else

$$\mu = \frac{\sum_{n=1}^{N} \tilde{y}_n - X}{N} : L1$$

if $\tilde{y}_n \geq \mu$ where $n \in S$, then $y_n = \tilde{y}_n - \mu$

else $N = N - 1$, $y_n = \tilde{y}_n$, $\{S\} = \{S\} - \{n\}$, go to $L1$

A.3 Classification-Maximization

Classification-step:

$$\mathbf{M}^* = \arg\min_{\mathbf{M}} \sum_{n=1}^{N} d_n(\mathbf{m}_n) + \lambda x_n(\mathbf{m}_n) \tag{A.3}$$

$$= \arg\min_{\mathbf{M}} \sum_{n=1}^{N} \beta_{ML} d_n(\mathbf{m}_n) + \alpha_{ML} x_n(\mathbf{m}_n), \text{ where } \lambda = \frac{\alpha_{ML}}{\beta_{ML}} \text{ and } \alpha_{ML}, \beta_{ML} > 0$$

$$= \arg\max_{\mathbf{M}} \sum_{n=1}^{N} -\beta_{ML} d_n(\mathbf{m}_n) - \alpha_{ML} x_n(\mathbf{m}_n) \tag{A.4}$$

$$= \arg\max_{\mathbf{M}} \prod_{n=1}^{N} e^{-\beta_{ML} d_n(\mathbf{m}_n) - \alpha_{ML} x_n(\mathbf{m}_n)}$$

$$= \arg\max_{\mathbf{M}} \prod_{n=1}^{N} \left\{ \left(\frac{\beta_{ML}}{\pi} \right)^{\frac{1}{2}} e^{-\beta_{ML} d_n(\mathbf{m}_n)} \right\} \left\{ \alpha_{ML} e^{-\alpha_{ML} x_n(\mathbf{m}_n)} \right\}$$

$$= \arg\max_{\mathbf{M}} \prod_{n=1}^{N} \left\{ \left(\frac{\beta_{ML}}{\pi} \right)^{\frac{1}{2}} e^{-\beta_{ML} \{\mathbf{y}_n - \hat{\mathbf{y}}_n(\mathbf{m}_n)\}^T \{\mathbf{y}_n - \hat{\mathbf{y}}_n(\mathbf{m}_n)\}} \right\} \left\{ \alpha_{ML} e^{-\alpha_{ML} x_n(\mathbf{m}_n)} \right\} \tag{A.5}$$

$$= \arg\max_{\mathbf{M}} \prod_{n=1}^{N} P(\mathbf{y}_n | \mathbf{m}_n, \beta_{ML}, \mathbf{y}_{ref}, QP) P(x_n | \mathbf{m}_n, \alpha_{ML}, \mathbf{y}_{ref}, QP)$$

$$= \arg\max_{\mathbf{M}} \prod_{n=1}^{N} P(\mathbf{y}_n, x_n | \mathbf{m}_n, \beta_{ML}, \alpha_{ML}, \mathbf{y}_{ref}, QP)$$

$$= \arg\max_{\mathbf{M}} \prod_{n=1}^{N} \frac{P(\mathbf{y}_n, x_n | \mathbf{m}_n, \beta_{ML}, \alpha_{ML}, \mathbf{y}_{ref}, QP) P(\mathbf{m}_n | \beta_{ML}, \alpha_{ML}, \mathbf{y}_{ref}, QP)}{P(\mathbf{y}_n, x_n | \beta_{ML}, \alpha_{ML}, \mathbf{y}_{ref}, QP)} \tag{A.6}$$

$$= \arg\max_{\mathbf{M}} \prod_{n=1}^{N} P(\mathbf{m}_n | \mathbf{y}_n, x_n, \beta_{ML}, \alpha_{ML}, \mathbf{y}_{ref}, QP) \tag{A.7}$$

$$\approx \arg\max_{\mathbf{m}} \int \prod_{n} P(\mathbf{m}_n | \mathbf{y}_n, x_n, \beta, \alpha, \mathbf{y}_{ref}, QP) P(\beta, \alpha | \mathbf{y}_n, x_n, \mathbf{y}_{ref}, QP) d\alpha d\beta \tag{A.8}$$

Maximization-step for luma:

$$\alpha_{ML}, \beta_{ML} = \arg\max_{\alpha,\beta} \prod_{n=1}^{N} P(\mathbf{y}_n, x_n | \mathbf{m}_n^*, \beta, \alpha, \mathbf{y}_{ref}, QP) \tag{A.9}$$

$$= \arg\max_{\alpha,\beta} \sum_{n=1}^{N} \log P(\mathbf{y}_n, x_n | \mathbf{m}_n^*, \beta, \alpha, \mathbf{y}_{ref}, QP) \tag{A.10}$$

$$= \arg\max_{\alpha,\beta} \sum_{n=1}^{N} \frac{1}{2} \log \beta - \beta d_n(\mathbf{m}_n^*) + \log \alpha - \alpha x_n(\mathbf{m}_n^*)$$

$$\alpha_{ML} = \frac{N}{\sum_{n=1}^{N} x_n(\mathbf{m}_n^*)}, \quad \frac{1}{\beta_{ML}} = \frac{2\sum_{n=1}^{N} d_n(\mathbf{m}_n^*)}{N}$$

$$\lambda = \frac{\alpha_{ML}}{\beta_{ML}} = \frac{2\sum_{n=1}^{N} d_n(\mathbf{m}_n^*)}{\sum_{n=1}^{N} x_n(\mathbf{m}_n^*)} \tag{A.11}$$

Maximization-step for chroma:

$$\alpha_{ML}, \beta_{ML} = \arg\max_{\alpha,\beta} \sum_{n=1}^{N} \frac{1}{2} \log \beta - \beta d_n(\mathbf{m}_n^*) + \log \alpha - \alpha x_n(\mathbf{m}_n^*) + w_0 \beta \tag{A.12}$$

$$\alpha_{ML} = \frac{N}{\sum_{n=1}^{N} x_n(\mathbf{m}_n^*)}, \quad \frac{1}{\beta_{ML}} = \frac{2\sum_{n=1}^{N} d_n(\mathbf{m}_n^*)}{N} - 2w_0$$

$$\lambda = \frac{\alpha_{ML}}{\beta_{ML}} = \frac{2\sum_{n=1}^{N} d_n(\mathbf{m}_n^*)}{\sum_{n=1}^{N} x_n(\mathbf{m}_n^*)} - 2w_0 \frac{N}{\sum_{n=1}^{N} x_n(\mathbf{m}_n^*)} = 2w_1 \tag{A.13}$$

where

$$w_0 = \frac{\sum_1^N d_n(\mathbf{m}_n^*) - w_1 \sum_1^N x_n(\mathbf{m}_n^*)}{N}$$

$$w_1 = \frac{N\sum_{n=1}^{N} d_n(\mathbf{m}_n^*) x_n(\mathbf{m}_n^*) - \sum_{n=1}^{N} x_n(\mathbf{m}_n^*) \sum_{n=1}^{N} d_n(\mathbf{m}_n^*)}{N\sum_{n=1}^{N} x_n^2(\mathbf{m}_n^*) - \left(\sum_{n=1}^{N} x_n(\mathbf{m}_n^*)\right)^2}$$

In (A.5), N is the number of MBs in a picture and two constants α_{ML} and $\left(\frac{\beta_{ML}}{\pi}\right)^2$ are scaling factors for normalization of the Gaussian and Exponential distribution. In (A.6), a uniform prior and a normalization factor are multiplied for the posterior distribution (A.7). In (A.10), we maximize the log likelihood function with respect to α and β and their solution are shown in (A.11). In the M-step for chroma, we can subtract an intercept w_0 at the RD optimization since subtraction of a constant term does not change solutions. The log likelihood function without constant terms

is described in (A.12). λ is 2 times slope w_1 in (A.13) where w_0 and w_1 are the least squares solution of $d_n(\mathbf{m}_n^*) = w_1 x_n(\mathbf{m}_n^*) + w_0$, $n = 1, ..., N$.

A.4 Optimal Solution of Partial Least Squares

$$
\begin{aligned}
\max_{\mathbf{u}} cov(\mathbf{Xu}, \mathbf{Wy}), \quad s.t.\ \mathbf{u^T u} = 1 \\
L = 2(\mathbf{Xu})^{\mathbf{T}}\mathbf{Wy} + \lambda(\mathbf{1} - \mathbf{u^T u}) \\
\nabla_{\mathbf{u}} L = 2\mathbf{X^T W y} - 2\lambda\mathbf{u} = 0 \\
\lambda = \mathbf{u^T X^T W y} \\
\mathbf{u^T X^T W y u} = \mathbf{X^T W y}, \quad (\mathbf{u^T} = \mathbf{y^T W^T X}) \\
\mathbf{u} = \frac{\mathbf{X^T W y}}{\mathbf{y^T W^T X X^T W y}}
\end{aligned}
\tag{A.14}
$$

The equation (A.14) is derived with assumption $(\mathbf{Wy})^{\mathbf{T}}\mathbf{Wy} = 1$ and $\mathbf{Xu} = \mathbf{Wy}$ as in [73], that is, $(\mathbf{Wy})^{\mathbf{T}}\mathbf{Xu} = \mathbf{1} = \mathbf{u^T u}$.

A.5 Deflation of input features for orthogonal projections with respect to W

$$
\begin{aligned}
\mathbf{X}_r = \mathbf{X}_{r-1} - \mathbf{z}_r \mathbf{p}_r^T \\
\mathbf{z}_{r+1} = \mathbf{X}_r \mathbf{u}_{r+1} = (\mathbf{X}_{r-1} - \mathbf{z}_r \mathbf{p}_r^T)\mathbf{u}_{r+1} \\
\mathbf{z}_r^T \mathbf{W} \mathbf{z}_{r+1} = (\mathbf{z}_r^T \mathbf{W} \mathbf{X}_{r-1} - \mathbf{z}_r^T \mathbf{W} \mathbf{z}_r \mathbf{p}_r^T)\mathbf{u}_{r+1} = 0 \\
\mathbf{p}_r = \frac{\mathbf{X}_{r-1}^T \mathbf{W} \mathbf{z}_r}{\mathbf{z}_r^T \mathbf{W} \mathbf{z}_r}
\end{aligned}
$$

A.6 Stochastic Meta Descent

We derive the equations (4.24) and (4.25) based on [75].

$$\begin{aligned}
\log \eta^{r+1} &= \log \eta^r - \mu \sum_{i=0}^{r} \tau^i \frac{\partial J(\mathbf{M}^{r+1})}{\partial \log \eta^{r-i}} \qquad \text{(A.15)} \\
&= \log \eta^r - \mu \frac{\partial J(\mathbf{M}^{r+1})}{\partial \mathbf{M}^{r+1}} \cdot \upsilon^{r+1}, \quad \left(\upsilon^{r+1} = \sum_{i=0}^{r} \tau^i \frac{\partial \mathbf{M}^{r+1}}{\partial \log \eta^{r-i}} \right) \\
\eta^{r+1} &= \eta^r \cdot \exp\left(-\mu \frac{\partial J(\mathbf{M}^{r+1})}{\partial \mathbf{M}^{r+1}} \cdot \upsilon^{r+1} \right) \\
&\approx \eta^r \cdot \max\left(\frac{1}{2}, 1 - \mu \frac{\partial J(\mathbf{M}^{r+1})}{\partial \mathbf{M}^{r+1}} \cdot \upsilon^{r+1} \right) \qquad \text{(A.16)}
\end{aligned}$$

In (A.15), step sizes are adapted in log-space and bi-linearization $e^x \approx \max(\frac{1}{2}, 1+x)$ is applied in (A.16).

$$\begin{aligned}
\upsilon^{r+1} &= \sum_{i=0}^{r} \tau^i \frac{\partial \mathbf{M}^{r+1}}{\partial \log \eta^{r-i}} \\
&= \sum_{i=0}^{r} \tau^i \frac{\partial \mathbf{M}^{r}}{\partial \log \eta^{r-i}} - \sum_{i=0}^{r} \tau^i \frac{\partial}{\partial \log \eta^{r-i}} \left(\eta^r \cdot \frac{\partial J(\mathbf{M}^r)}{\partial \mathbf{M}} \right) \\
&\approx \tau \upsilon^r - \eta^r \cdot \left(\frac{\partial J(\mathbf{M}^r)}{\partial \mathbf{M}} + \tau \frac{\partial^2 J(\mathbf{M}^r)}{\partial^2 \mathbf{M}} \upsilon^r \right) \qquad \text{(A.17)}
\end{aligned}$$

where

$$\begin{aligned}
\mathbf{M}^{r+1} &= \mathbf{M}^r - \eta^r \cdot \frac{\partial J(\mathbf{M}^{\mathbf{r}})}{\partial \mathbf{M}}, \quad \upsilon^r = \sum_{i=1}^{r} \tau^{i-1} \frac{\partial \mathbf{M}^r}{\partial \log \eta^{r-i}} \\
\sum_{i=0}^{r} \tau^i \frac{\partial \mathbf{M}^r}{\partial \log \eta^{r-i}} &= \tau \sum_{i=1}^{r} \tau^{i-1} \frac{\partial \mathbf{M}^r}{\partial \log \eta^{r-i}} + \frac{\partial \mathbf{M}^r}{\partial \log \eta^r} = \tau \upsilon^r \qquad \text{(A.18)}
\end{aligned}$$

The equation (A.17) is derived with assumption $\sum_{i=1}^{r} \tau^i \frac{\partial \log \eta^r}{\partial \log \eta^{r-i}} = 0$ and the equation (A.18) results from $\frac{\partial \mathbf{M}^r}{\partial \log \eta^r} = 0$.

A.7 Acknowledgement

Portions of the text of this appendix are adapted from material that has been submitted for publication as: C. An and T. Q. Nguyen, "Analysis of Lagrange

Multiplier Selection in H.264 and Its adaptation using Classification-Maximization", under review with *the IEEE Transactions on Image Processing*, 2008, and C. An and T. Q. Nguyen, "Iterative Rate-Distortion optimization of H.264 with Constant Bit Rate Constraint" was published in *the IEEE Transactions on Image Processing*, vol. 17, pp. 1605-1615, Sep., 2008. The dissertation author was the primary researcher for these publications, and the co-author directed and supervised the research which forms the basis for this chapter.

Bibliography

[1] "Advanced video coding for generic audiovisual services," ITU-T Recommendation H.264 & ISO/IEC 14496-10 AVC, 2003.

[2] I. E. G. Richardson, *H.264 and MPEG-4 video compression: video coding for next-generation Multimedia*. John Wiley & Sons Press, 2003.

[3] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC Video Coding Standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, pp. 560–576, July 2003.

[4] "ISO/IEC 13818 Information Technology: generic coding of moving pictures and associated audio information," ISO 13818-2 MPEG-2, 1995.

[5] "Video coding for low bit rate communication," ITU-T Recommendation H.263, 1996.

[6] D.P.Bertsekar, *Nonlinear Programming*, 2nd ed. Athena Scientific, 2003.

[7] D. Palomar and M. Chiang, "Alternative distributed algorithms for network utility maximization: Framework and applications," *IEEE Trans. Automat. Contr.*, vol. 52, pp. 2254–2269, Dec. 2007.

[8] B. Johansson and M. Johansson, "Primal and dual approaches to distributed cross-layer optimization," in *Proc. 16th IFAC World Congress*, Prague, Czech republic 2005.

[9] *H.264/AVC reference software (JM11.0)*, HHI. [Online]. Available: http://iphome.hhi.de/suehring/tml/download/

[10] J. T. Tou and R. C. Gonzalez, *Pattern Recognition Principles*. Addison-Wesley, 1974.

[11] A. Ortega and K. Ramchandran, "Rate-Distortion methods for Image and Video Compression: An Overview," *IEEE Signal Processing Mag.*, vol. 15, pp. 23–50, Nov. 1998.

[12] G. J. Sullivan and T. Wiegand, "Rate-Distortion Optimization for Video Compression," *IEEE Signal Processing Mag.*, vol. 15, pp. 74–99, Nov. 1998.

[13] A. Puria, X. Chenb, and A. Luthrac, "Video coding using the H.264/MPEG-4 AVC compression standard," *Signal Processing: Image Communication*, vol. 19, pp. 793–849, Oct. 2004.

[14] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[15] T. Wiegand, M. Lightstone, T. G. Campbell, and S. k. Mitra, "Rate-distortion optimized mode selection for very low bit rate video coding and the emerging h.263 standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 2, pp. 182–190, Apr. 1996.

[16] Y. Yang and S. S. Hemami, "Generalized rate-distortion optimization for motion-compensatedvideo coders," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, pp. 942–955, Sept. 2000.

[17] K. Ramchandran, A. Ortega, and M. vetterli, "Bit allocation for dependent quantization with applications to multiresolution and mpeg video coders," *IEEE Trans. Image Processing*, vol. 3, pp. 533–545, Sept. 1994.

[18] T. Chiang and Y.-Q. Zhang, "A new rate control scheme using quadratic rate distortion model," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 246–250, Feb. 1997.

[19] H. Kim, "Adaptive rate control using nonlinear regression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, pp. 432–439, May 2003.

[20] T. Wiegand and B. Girod, "Lagrange multiplier selection in hybrid video coder control," in *Proc. IEEE ICIP*, 2001.

[21] K. Takagi, "Lagrange Multiplier and RD-characteristics," JVT-C084, ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6, May 2002.

[22] T. Weigand, H. Schwarz, A. Joch, F. Kossentini, and G. J. Sullivan, "Rate-Constrained Coder Control and Comparison of Video coding Standards," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, pp. 688–703, July 2003.

[23] Z. Li, F. Pan, K. P. Lim, G. Feng, X. Lin, and S. Rahardja, "Adaptive basic unit layer rate control for jvt," JVT-G012-r1, ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6, Mar. 2003.

[24] Z. G. Li, W. Gao, F. Pan, S. W. Ma, K. P. Lim, G. N. Feng, X. Lin, S. Rahardja, H. Q. Lu, and Y. Lu, "Adaptive rate control for h.264," *Visual Communication and Image Representation*, vol. 17, pp. 376–406, Apr. 2006.

[25] W. Yuan, S. Lin, Y. Zhang, W. Yuan, and H. Luo, "Optimal Bit Allocation and Rate Control for H.264," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 16, pp. 705–715, June 2006.

[26] H.-J. Lee, T. Chiang, and Y.-Q. Zhang, "Scale Rate Control for MPEG-4 Video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, pp. 878–894, Sept. 2000.

[27] Z. Li, C. Zhu, N. Ling, X. Yang, G. Feng, S. Wu, and F. Pan, "A unified architecture for real-time video-coding systems," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, pp. 472– 487, June 2003.

[28] S. Zhou, J. Li, J. Fei, and Y. Zhang, "Improvement on Rate-Distortion Performance of H.264 Rate Control in Low Bit Rate," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, pp. 996–1006, Aug. 2007.

[29] "Annex B-Hypothetical Reference Decoder, Video Coding for Low Bit Rate Communication," ITU-T Recommendation H.263, 1998.

[30] "Annex C-Video Buffering Verifier, Information Technology-Generic coding of moving pictures and associated audio information: Video," ISO 13818-2 MPEG-2, 2000.

[31] J. Ribas-Corbera, P. Chou, and S. Requnathan, "A Gneralized Hypotheical Reference Decoder for H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, pp. 674–687, July 2003.

[32] S. Ma, W. Gao, F. Wu, and Y. Lu, "Rate control for JVT video coding scheme with HRD considerations," in *Proc. ICIP*, 2003.

[33] Z. G. Li, N. Ling, S. Rahardja, X. Lin, and P. Li, "An iterative method for hypothetical reference decoder," in *Proc. ICME*, 2004.

[34] J. L. H. Webb, "HRD Conformance for Real-time H.264 Video Encoding," in *Proc. ICIP*, 2007.

[35] D. S. Taubman and M. W. Marcellin, *JPEG2000 : Image Compression Fundamentals, Standards and Practice*. Kluwer Academic Publishers, 2002.

[36] G. Bjontegaard, "Calculation of average PSNR differences between RD-Curves," VCEG-M33, ITU-T Video Coding Experts Group (VCEG), Mar 2001.

[37] L. Chen and I. Garbacea, "Adaptive lambda estimation in lagrangian rate-distortion optimization for video coding," in *Proc. VCIP*, 2006.

[38] X. Li, N. Oertel, A. Hutter, and A. Kaup, "Advanced lagrange multiplier selection for hybrid video coding," in *Proc. IEEE ICME*, 2007.

[39] ——, "Adaptive lagrange multiplier selection for intra-frame video coding," in *Proc. IEEE ISCAS*, 2007.

[40] Z. Li and A. M. Tourapis, "An estimation-theoretic interpretation of video rate distortion optimization with lagrangian formulation," in *Proc. IEEE DCC*, 2008.

[41] F. Yang, S. Wan, and E. Izquierdo, "Lagrange Multiplier Selection for 3-D Wavelet Based Scalable Video Coding," in *Proc. IEEE ICIP*, 2007.

[42] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[43] O. Cappé and E. Moulines, "Online EM Algorithm for Latent Data Models," *ArXiv e-prints*, vol. 712, Dec. 2007. [Online]. Available: http://arxiv.org/abs/0712.4273

[44] A. C. Yu, G. R. Martin, and H. Park, "A frequency domain approach to intra mode selection in h.264/avc," in *Proc. EUSIPCO*, 2005.

[45] T. Tsukuba, I. Nagayoshi, T. Hanamura, and H. Tominaga, "H.264 fast intra-prediction mode decision. based on frequency characteristic," in *Proc. EUSIPCO*, 2005.

[46] P. Zhang, D. Zhao, S. Ma, Y. Lu, and W. Gao, "Multiple modes intra-prediction in intra coding," in *Proc. IEEE ICME*, 2004.

[47] C. Chen, P.-H. Wu, and H. Chen, "Transform-domain intra prediction for h.264," *IEEE Internatinal Symposium on Circuits and Systems*, vol. 2, pp. 1479–1500, May 2005.

[48] T. K. Tan, C. S. Boon, and Y. Suzuki, "Intra prediction by template matching," in *Proc. IEEE ICIP*, 2006.

[49] L. K. Tang and K. N. Nqan, "Enhancement techniques for intra block matching," in *Proc. IEEE Multimedia and Expo*, 2007.

[50] J. Yang, B. Yin, Y. Sun, and N. Zhang, "A block-matching based intra frame prediction for h.264/avc," in *Proc. IEEE Multimedia and Expo*, 2006.

[51] S. Knodo, H. Sasai, and S. Kadono, "Tree structed hybrid intra prediction," in *Proc. IEEE ICIP*, 2004.

[52] A. Smola and B. Schöelkopf, "A tutorial on support vector regression," *NeuroCOLT2 Technical Report NC2-TR-1998-030*, 1998. [Online]. Available: citeseer.ist.psu.edu/smola03tutorial.html

[53] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[54] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.

[55] S. Vijayakumar, A. D'Souza, and S. Schaal, "Incremental online learning in high dimensions," *Neural Computation*, vol. 17, no. 12, pp. 2602–2634, Dec. 2005.

[56] J. Kivinen, A. Smola, and R. Williamson, "Online learning with kernels," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 52, no. 8, pp. 2165–2176, Aug. 2004.

[57] G. Cauwenberghs and T. Poggio, "Incremental and decremental support vector machine learning," in *NIPS*, 2000.

[58] J. Ma, J. Theiler, and S. Perkins, "Accurate on-line support vector regression," *Neural Computation*, vol. 15, no. 11, pp. 2683–2703, Nov. 2003.

[59] S. Schaal and C. G. Atkeson, "Constructive incremental learning from only local information," *Neural Computation*, vol. 10, no. 8, pp. 2047–2084, 1998.

[60] T. Evgeniou, M. Pontil, and T. Poggio, "Regularization networks and support vector machines," 2000.

[61] L. Bo, L. Wang, and L. Jiao, "Recursive finite newton algorithm for support vector regression in the prima," *Neural Computation*, vol. 19, no. 4, pp. 1082–1096, 2007.

[62] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Processing Letters*, vol. 9, no. 3, pp. 293–300, 1997.

[63] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, "A practical guide to support vector classification," 2007. [Online]. Available: http://www.csie.ntu.edu.tw/ cjlin/papers/guide/guide.pdf

[64] S. Lessmann, R. Stahlbock, and S. F. Cron, "Genetic algorithms for support vector machine model selection," in *ICJNN*, 2006.

[65] Z. Chunhong and J. Licheng, "Automatic parameters selection for svm based on ga," in *Intelligent Control and Automation*, 2004.

[66] X. Wang, H. Zhang, C. Zhang, X. Cai, J. Wang, and J. Wang, "Prediction of chaotic time series using ls-svm with automatic parameter selection," in *PDCAT*, 2005.

[67] C. Houck, J. Joines, and M. Kay, "A Genetic Algorithm for Function Optimization: A Matlab Implementation," North Carolina State University, Raleigh, NC, Tech. Rep. NCSU-IE-TR-95-09, 1995. [Online]. Available: http://www.ise.ncsu.edu/mirage/GAToolBox/gaot/

[68] R. Liu, E. Liu, J. Yang, M. Li, and F. Wang, *Intelligent Control and Automation: Optimizing the Hyper-parameters for SVM by Combining Evolution Strategies with a Grid Search.* Springer, 2006.

[69] J. Viron, M. Wien, and H. Schwarz, *Joint Scalable Video Model(JSVM) 11 Software, Joint Video Team, Doc. JVT-X203*, 2007.

[70] C.-C. Chang and C.-J. Lin, *LIBSVM: A library for Support Vector Machines*, 2007. [Online]. Available: http://www.csie.ntu.edu.tw/˜cjlin/libsvm

[71] C. E. Rasmussen and C. K. I. Williams., *Gaussian Process for Machine Learning.* MIT Press, 2006.

[72] C. Atkeson, A. Moore, and S. Schaal, "Locally weighted learning," *AI Review*, vol. 11, pp. 11–73, April 1997.

[73] K. P. Bennett and M. J. Embrechts, *An Optimization Perspective on Kernel Partial Least Squares Regression.* In J. Suykens et al., editor, Advances in Learning Theory: Methods, Models and Applications, pages 227–249. NATO Science Series, Series III: Computer and System Sciences - Vol. 190, IOS Press, Amsterdam, The Netherlands, 2003.

[74] C. Gold, A. Holub, and P. Sollich, "Bayesian approach to feature selection and parameter tuning for support vector machine classifiers," *Neural Networks*, vol. 18, pp. 693–701, 2005.

[75] S. V. N. Vishwanathan, N. N. Schraudolph, and A. J. Smola, "Step Size Adaptation in Reproducing Kernel Hilbert Space," *Journal of Machine Learning Research*, vol. 7, pp. 1107–1133, 2006.

[76] D. Palomar, "Convex primal decomposition for multicarrier linear mimo transceivers," *IEEE Trans. Signal Processing*, vol. 53, no. 12, pp. 4661–4674, Dec. 2005.