Reduction of Prediction Side-Information for Image and Video Compression

by

Lucas Nissenbaum

B.S. Cornell University, 2013
S.M. Massachusetts Institute of Technology, 2015

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2021

Author . . . .
Department of Electrical Engineering and Computer Science
January 27, 2021

Certified by . . . .
Professor Jae S. Lim
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by . . . .
Professor Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Reduction of Prediction Side-Information for Image and Video Compression

by

Lucas Nissenbaum

Submitted to the Department of Electrical Engineering and Computer Science on January 27, 2021, in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Electrical Engineering and Computer Science

Abstract

Many compression systems are based on parameterized predictors. In HEVC, directional intra-prediction specifies a prediction direction. Motion compensation forms a motion vector to predict a block's movement from another frame. These parameters lead to side-information bits, which must be encoded. In recent image and video compression standards, the number of intra-prediction directions used has been increasing. Motion vectors are also using finer fractional precision. The side-information bits have therefore become a significant part of the encoder bit-rate.

In this thesis, we will show that there is significant room to use adaptive ways to reduce this side-information. In particular, we will develop a theoretical framework to consider this side-information. In this theoretical framework, we assign a set of possible values of side-information parameter for each block based on information available at the decoder. Based on this framework, two main questions are proposed: How do we find the number of values that compose this set? If we know the cardinality of this set, what values should compose it?

We propose three methods to reduce the intra-prediction side-information bitrate, based on this framework and on prediction inaccuracy modeling. Our first method selects between a set of 7 modes and the full set of 35 modes from HEVC, by thresholding the maximum absolute gradient boundary. Our second method selects between four possible sets, by using scaled thresholds derived from prediction inaccuracy modeling. The third method uses only two sets, but constructs the smaller set adaptively based on neighboring blocks' information. We then present a theoretical and experimental comparison between these three methods.

We then propose a method to adaptively decide whether or not to use fractional precision motion vectors. Our experimental results show there is room to use side-information reduction for the case of motion compensation.

Thesis Supervisor: Professor Jae S. Lim
Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

First and foremost, I would like to thank my research advisor, Prof. Jae S. Lim, for all his mentoring and guidance during my time at MIT. Prof. Lim is a great researcher. The freedom he gives his students to pursue their research ideas and the insight on his research input have been invaluable. I feel very lucky for having been advised by someone who not only has such invaluable knowledge and creativity, but also someone who cares about his students as human beings. Having had the opportuinity to TA for him multiple times, I can say I owe to him a lot of what I have learned about teaching. I am also very thankful to Prof. Vivienne Sze and Dr. John Apostolopoulos for the comments on the research here presented.

My research would not have been completed without the long discussions and dedicated assistance from my friends and labmates Megan Fuller and Xun Cai. Both of them have greatly enriched my experience, with long discussions on all image processing related topics. My graduate studies and my research would not have been the same without them. I would also like to thank Cindy LeBlanc for always looking out for us, and for being there to make our lives a lot easier. Mumin Jin's work as a UROP for intra-prediction side-information reduction has also been fundamental to the results presented in this thesis, and I am very thankful for all the time she has dedicated to it, and for her friendship.

I would like to thank my family. Throughout my whole time abroad, my parents, Selmo and Genny, have made a significant effort to always be present and attentive, and have been more helpful than I believe they realize. Throughout this whole long experience, there has never been a moment where I doubted their love and support. I would also like to thank my sister Rafaela for being a great source of love and laughter in the moments I needed it.

I wish also to thank my friends for being next to me throughout this whole experience. I especially thank Alfredo, Antonio, Bernardo, Carlo, Hugo, Leonardo, and Rafael for always keeping in touch and remaining close while I have been abroad. I am also very thankful to Camille, Carlos, Christina, Lorenzo, Nicole, Renata and Van for the dinners, discussions, and trips we have had. Our meetings were some of the highlights of my weeks, and I am very grateful for having shared these moments with you.

I should also thank some of my friends I have met through my studies at MIT. Specifically, I would like to thank Catherine, Eren, Eric, Gregory, James, Jessica and Pablo for all the interesting discussions and fun times they have given me throughout this experience.

Thank you all.

Contents

List of Figures

List of Tables

Chapter 1

Thesis Overview

In prediction-based image and video coding, parameterized predictors are used to produce prediction residuals and reduce the signal variance. This is true for both intra-prediction and motion compensated prediction residuals, for example. The residuals are then transformed, quantized and entropy encoded. With a proper choice of prediction parameters, a significant fraction of the bit-rate can be reduced.

Recent compression standards have significantly increased the number of possible values allowed for the parameterized predictors' parameters. The number of intra-prediction directions has significantly increased. Motion vectors now use quarter or even eighth pixel interpolation. This leads to a significant increase in the amount of side-information required to encode the signal. This amount of encoding side-information is likely to increase as future standards are developed.

In this thesis, we consider a theoretical framework to evaluate the role of side-information in prediction. In particular, we consider under which contexts we should use a larger number of intra-prediction directions in a block, or in which blocks we should use longer precision motion vectors. We use the prediction inaccuracy modeling to justify our statistical assumptions, as this model relates the imprecision in your prediction parameter to the variance in the prediction residuals. Through this new framework, we are able to improve encoder performance by choosing a better set to use in a block-by-block basis.

As an example of why this framework can work, we note that in the case of intra-prediction, if we have a large edge, we wish the direction to be as precise as possible, since even a small angular error will lead to a large overall error. On the other hand, if we have a small edge, we wish the direction to be precise, but do not require it to be too precise. In the second case, we can use a smaller number of directional intra-prediction modes. We also adapt such framework to the scenario of motion compensation, and show how we could also adaptively choose to use a larger or smaller number of bits in terms of prediction directions.

We will discuss the theoretical framework and its applications for the rest of this thesis. The thesis is structured as follows:

- In Chapter 2, we review some of the fundamental concepts of image and video compression systems. In particular, we will go over the main modules that most image and video compression systems are composed of.

- In Chapter 3, we present the prediction inaccuracy model, which will allow us to provide a statistical analysis of our problem.

- In Chapter 4, we consider the problem of reducing prediction side-information from a theoretical perspective. We will propose a framework that adaptively decides a set of side-information parameter values for each block.

- In Chapter 5, we propose a simple algorithm to reduce intra-prediction side-information adaptively. This first algorithm will focus on deciding between whether to use a large set or a small set of prediction directions when encoding a block.

- In Chapter 6, we will discuss the results obtained when implementing the algorithm from Chapter 5. In particular, we observe that using a side-information reduction system can lead to significant improvement in HEVC's performance.

- In Chapter 7, some alternatives to the original algorithm from Chapter 5 will be considered. We will propose alternatives by considering using more than two possible set sizes, and by considering using adaptive sets instead.

- In Chapter 8, we will propose adapting our theoretical framework to the case of motion compensation residuals. We will derive an algorithm based on prediction inaccuracy modeling to decide whether or not to use fractional pixel precision when encoding a motion vector.
- In Chapter 9, we will discuss the results found from the motion compensation residuals.
- In Chapter 10, we will conclude by summarizing the contributions of the thesis and discussing possible future extensions.

Chapter 2

Introduction

In this chapter, we will present most background information required. We review the background of prediction-based image and video compression systems in Section 2.1, and then consider side-information reduction problem in Section 2.2.

2.1 Prediction-Based Image and Video Encoders

A typical video sequence in its raw format contains a large amount of data. This raw format is not viable for many storage, transmission, and processing applications. In image and video compression, our goal is to represent this image or video with as few bits as possible, while preserving a certain level of video quality. There are two broad categories of compression processes: lossy and lossless compression. In lossless compression, the signal must be perfectly preserved. In lossy compression, the goal is to preserve important visual information within the signal, but some distortion is allowed. In this thesis, our focus will be on lossy compression systems.

Among the lossy compression systems, many of these are prediction and transform based. Figure 2-1 shows a high-level diagram of a prediction and transform-based compression system. The encoder side of the system, shown in the upper branch of the figure, has as input a raw signal and as output a sequence of bits. This sequence of bits is in turn stored or transmitted through a network. The sequence is then decoded by a decoder, shown in the lower branch of Figure 2-1. This decoder has as input a bit-sequence and reconstructs a signal. Since this encoder is lossy, this signal will not necessarily be the same as the original signal, but should be similar to some degree. The system should be designed in such a way that the number of bits transmitted after the encoding process is significantly smaller than the number of bits for the raw signal.

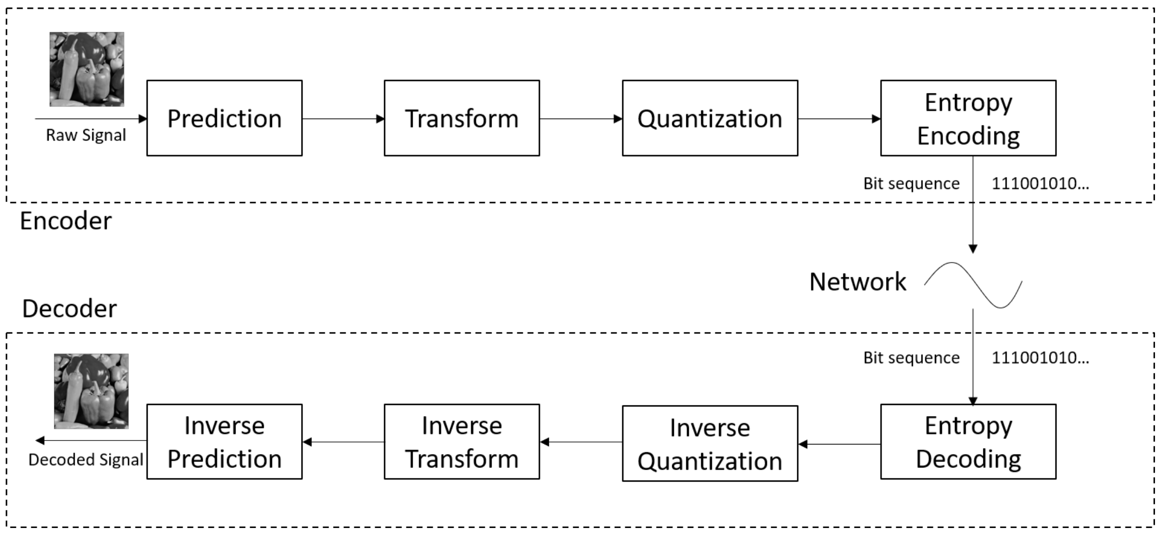


Figure 2-1: Block diagram of generalized prediction and transform-based compression system.

The objective of this encoding is to reduce the amount of redundant information transmitted. This redundant information may come from many sources. In image compression,neighboring pixels tend to contain similar information. These pixels are highly correlated, which is referred to as spatial redundancy. In video compression, not only do we have spatial redundancy, but we also have temporal redundancy, as two frames in a video sequence will contain very similar information.

Figure 2-1 also shows a typical structure of a prediction and transform-based compression system. The encoder is formed by four modules: prediction, transform, quantization, and entropy coding. Many video compression standards such as HEVC [1] and H.264 [2] follow this structure. In this chapter, we will consider these modules, as well as other associated components, in more detail.

2.1.1 Block Partition

A typical image and video compression system encodes one frame at a time. For each frame, the signal is partitioned into blocks. Block processing allows the encoder to take into account the local signal characteristics. Smaller regions have higher degree of similarity and local stationarity, and therefore can be efficiently encoded using fewer bits. On the other hand, larger regions may encode relatively large areas in a small number of bits; as long as these areas exhibit a high degree of redundancy, they will be more efficient to encode than the smaller areas.

A compression system may choose to use a fixed block size, or a variable block size. In JPEG image compression system [3], each image is segmented into blocks of fixed size 8×8. In H.264/AVC video compression system [2], each video frame is fragmented into macroblocks of size 16×16, and may be adaptively segmented into smaller blocks, including non-square rectangular blocks such as blocks of size 8×16.

More recent compression systems such as High Efficiency Video Coding (HEVC) and Versatile Video Coding (VVC) allow for even more flexible block partition strategies. HEVC [1] allows a block to be partitioned recursively using a coding tree unit structure [4]. Smaller blocks are used to adapt to finer features in an image, such as the detailed information in the poster. Meanwhile, larger blocks are used to encode large smooth regions in the original image, such as the blue background. In this sense, block partitioning is chosen to ensure areas within blocks have high spatial redundancy.

2.1.2 Prediction

After partitioning into smaller blocks, prediction is applied to each block. When encoding a series of blocks, a prediction is formed based on previously decoded samples, and we encode the residual signal given by the difference between this signal and its prediction. By subtracting the prediction, we attempt to exploit the dependency between neighboring blocks to reduce the variance within each block by removing redundant information.

Prediction-based compression systems often uses a parameterized predictor. This predictor takes some parameter α, and forms a prediction $\widehat{x}(n)$ as a function of α and the previous decoded samples. In this case, α does need to be encoded, and incurs in an additional cost. We will now consider two of the main classes of parameterized predictors used in image and video compression.

Intra-Prediction

Intra-prediction is a type of prediction in which we wish to predict a block from its known spatial neighboring pixels. The idea is to use neighboring pixels to predict the content in the current block and reduce spatial redundancy.

A common mode of intra-prediction used in many standards is DC mode. In DC mode, the boundary pixels are averaged, and the current block is estimated as a constant given by this average. Another mode used in HEVC is planar mode, where a plane is fit to the values of the boundary pixels.

An important class of parameterized predictors used for intra-prediction is directional intra-prediction. The potential boundary pixels used in HEVC are shown in Figure 2-2. Note that not all boundary pixels may be available, and missing pixels are therefore estimated. In this case, a direction is chosen from a set of valid directions, and pixels are predicted by copying the values according to the given direction. Figure 2-3 shows an example of intra-predicted block for vertical direction.

Intuitively, intra-prediction can perform especially well in a block with edges if it uses the exact direction of an edge in a block. This effect can be seen in the previous example in Figure 2-3. Due to this behavior, a large set of prediction directions can be used in most standards. Figure 2-4 shows the set of valid directional intra-prediction directions for H.264. HEVC allows DC and planar modes to be used, as well as a total of 33 intra-prediction directions shown in figure 2-5. VVC allows for an even larger set of 67 possible intra-prediction modes [5].

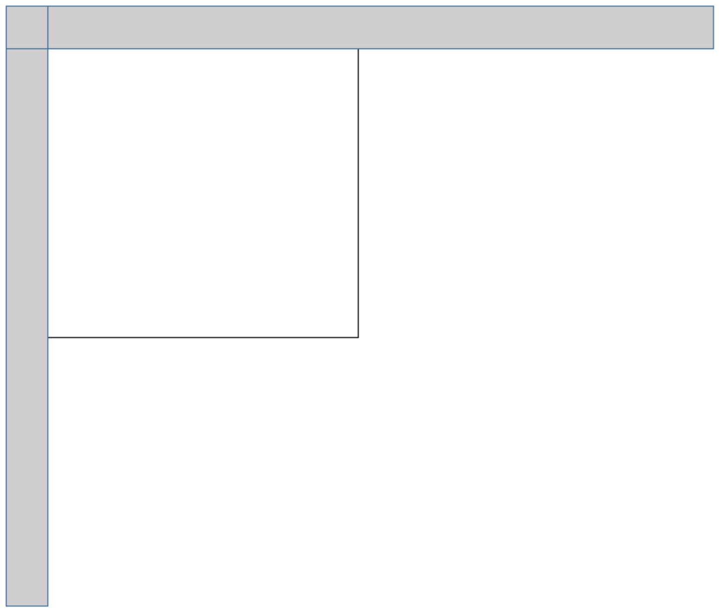


Figure 2-2: Block boundaries that may be available for intra-prediction.

Motion Compensation

In video compression, it is very important to reduce the temporal redundancy between frames. The process of reducing the redundancy between frames is called inter-prediction. When observing adjacent frames, they tend to be closely related as they are representing the same scene, except for moving objects and cameras. Motion compensation aims to take such correlation and motion into account [6].

In this case, a block is predicted from previous frames. A motion vector indicates which block from the previous frame should be copied to the current block, as shown in Figure 2-6. This vector is obtained by a process known as block matching. In this method, a block in the previously processed frame which is similar to the current block is chosen. Intuitively, motion compensation captures the motion between the different blocks, and thus achieves smaller prediction residuals. The motion vectors in H.264 and HEVC are allowed to have fractional prediction, as it allows to more precisely capture finer motion. This motion vector parameter is computed in the encoder, and must be transmitted to the decoder, leading to some side-information bit-rate.

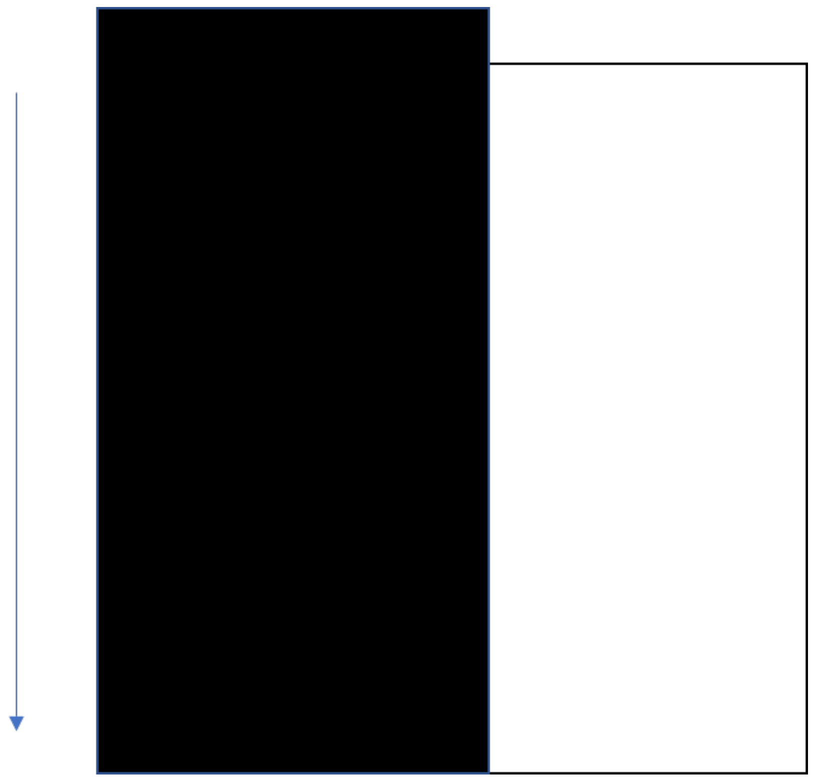


Figure 2-3: Example of vertical intra-prediction. Pixels get copied along fixed prediction direction to form prediction block. As long as prediction direction is close to edge direction, associated error will be small.

In the case of motion compensation, the error will be smaller in areas with higher temporal correlation, while it should be larger in areas with lower temporal correlation. This means it will perform especially poorly in edges or rotating objects, where motion inaccuracy will be larger, or in areas that were occluded in previous frames.

2.1.3 Transform Encoding

Prediction reduces spatial redundancy, but these residuals still are still highly correlated. For instance, neighboring pixels within a frame may still contain information that was not found before in the boundary or in previous frames. The transform step reduces this redundancy by representing the signal in a sparse manner in the transform domain.

A large amount of signal energy can be represented with a small number of coefficients. This property is known as energy compaction as has been widely studied in

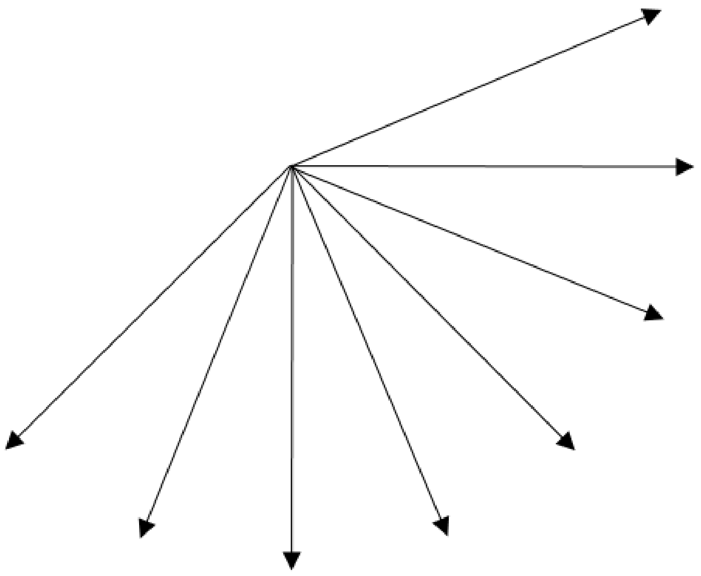


Figure 2-4: Intra-prediction directions used in H.264.

signal processing literature [7]. Figure 2-7 shows the energy of a signal as a function of the number of Fourier transform coefficients kept from it.

As can be seen, most of the energy is concentrated in a small number of coefficients. From a theoretical perspective, the Karhunen-Loeve transform [8] has been found to be the optimal linear transform from an energy compaction perspective in the case where the distribution of the signal is known. This transform can be obtained by computing the eigenvector decomposition of the signal's correlation matrix, and keeping the eigenvectors corresponding to the largest eigenvalues, as these indicate the orthogonal components with largest variances.

By considering a two-dimensional Markov-1 correlation model, it has been found that as neighboring pixel's correlation converges to 1, the KLT will converge to the 2-D DCT [9]. In most image and video compression systems a transform based on the 2-D DCT is used, such as the integer DCT. The objective of the integer DCT is to concentrate the energy in as few coefficients as possible, but using only computationally efficient integer operations. The basis functions are chosen to satisfy orthogonality and normality properties as close as possible. This transform is used

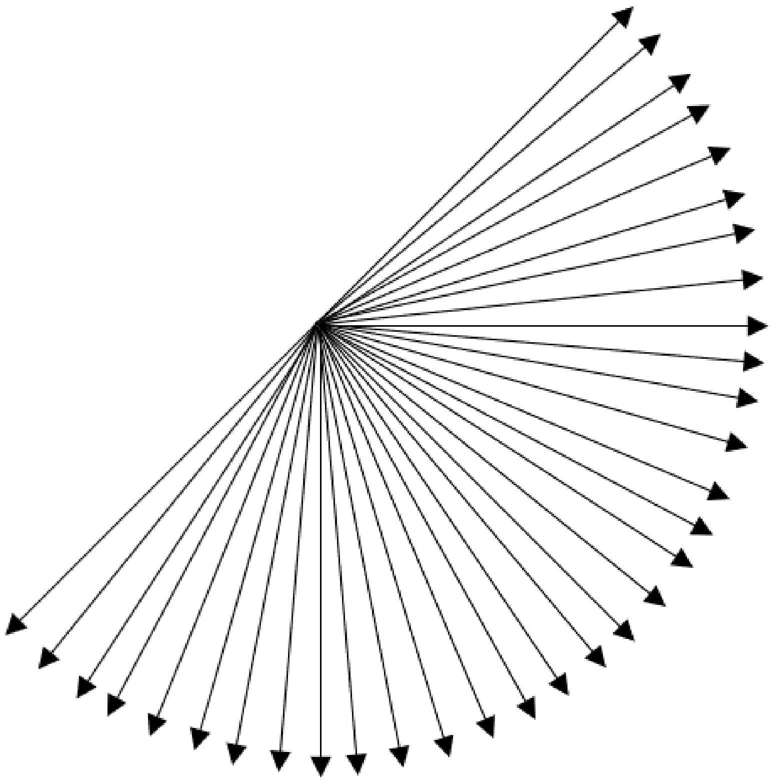


Figure 2-5: Intra-prediction directions used in HEVC.

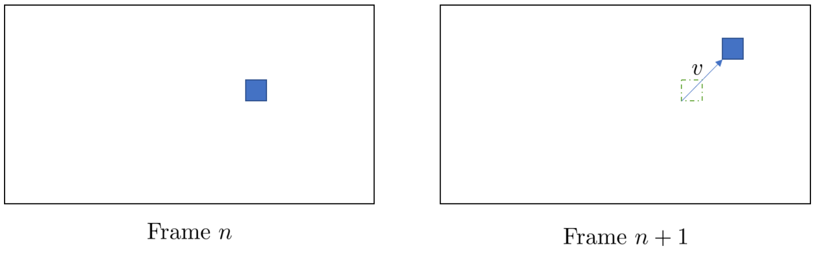
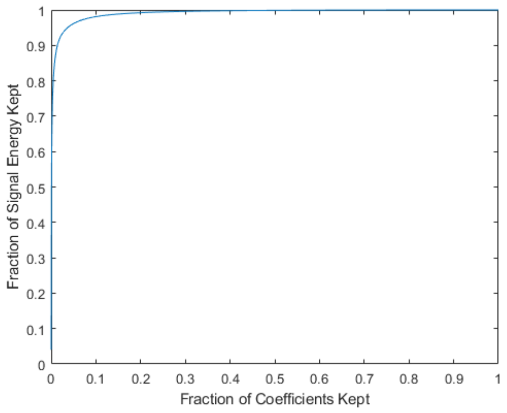


Figure 2-6: Blocks tend to move between frames. In this example, the blue block can be traced back to the previous frame by taking its movement a vector of v. By transmitting v instead of the original block, a system is able to significantly reduce the bit-rate used to encode it as it can simply encode the residual between the current block and the previously decoded block.

in HEVC.

Many other transforms have been proposed and used in compression systems. In [10, 11], one-dimensional DCTs are proposed to encode edge blocks. By using one-dimensional transforms in the direction of an edge, the signal will be encoded with fewer coefficients. In HEVC, the asymmetric discrete sine transform (ADST) [12] can

(a) Peppers



(b) Energy Compaction.

Figure 2-7: The fraction of energy retained in non-DC coefficients when only a small fraction of coefficients is kept from the image in (a) is shown in (b). As can be seen, most of the signal's energy is retained in a small number of coefficients.

be used to encode 4×4 intra-prediction residuals.

2.1.4 Quantization

After a block is transformed, the resulting coefficients usually consist of floating point numbers. These coefficients are quantized so they map to a small number of values, usually integers. Uniform scalar quantization is typically used, with different step sizes for each of the transform coefficients. Figure 2-8 exemplifies the input/output mapping of a scalar quantizer.

Note that the quantizer is the main lossy step of the encoder, and is therefore responsible for establishing the rate-distortion trade-off. This is done by choosing larger or smaller step sizes in the uniform quantizer. The step sizes are controlled by a quantization parameter, denoted as QP. The larger the QP value, the worse quality you will obtain out of the compressed signal. Individual coefficients are also quantized with larger or smaller step sizes based on their relevancy to human perception, and their relative frequency in most sequences.

A related problem to the scalar quantization we described is vector quantization. In this case, you wish to quantize a vector of possibly correlated random variables into a small number of levels. A natural question that has been widely studied is the

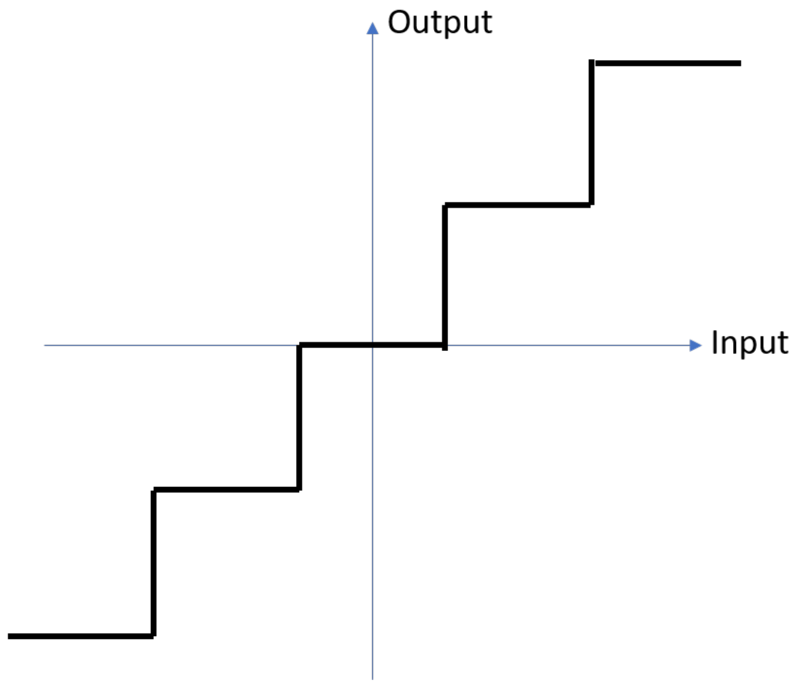


Figure 2-8: Example of a quantizer function. This function maps inputs to outputs, reducing a large set of possible input values to a much smaller set of output values, which should require fewer bits to encode.

design of a non-uniform but optimal quantizer from a Mean-Squared Error (MSE) perspective [13]. The Lloyd-Max algorithm [14] assumes knowledge of the probability distribution of the original signal to solve this problem.

2.1.5 Entropy Coding

After quantization, entropy encoding maps the quantization levels obtained into bits. This is ideally done taking into account the likelihood of each quantization level. To reduce the average length of the bit-stream, a more likely keyword has fewer bits, while a less likely keyword uses more bits. In theory, the optimal prefix-free uniquely decodable codewords are given by the Huffman algorithm [15].

One may reduce the number of bits used even further by using joint encoding [16]. This is done in HEVC by using arithmetic encoding [17], where a context is defined to estimate the probability of each bit being 1 or 0, and multiple bits are jointly encoded taking the distribution into account.

2.2 Side-Information Reduction

One of the main issues with parameterized predictors is that they require side-information to be transmitted. This adds an encoding overhead. In H.264, a total of 10 intra-prediction modes was used. In HEVC, a total of 35 modes is used. This number is increasing even further in the versatile video coding (VVC) standard to a total of 67 modes [5]. Similarly, motion vector precision has gone from half-pixel to quarter pixel, and further down to eigth-pixel precision. This implies a larger number of bits used to transmit the side-information. As side-information bit-rate becomes more significant, we believe there should be better ways to deal with the increase in side-information.

We note that not all of the blocks require the same precision for their side-information parameter. Let us consider an example to see why this is true. Let us take the block with an edge from our previous example in Figure 2-3, and assume we are off from the true edge by a small angle θ as shown in Figure 2-9. Then the area where the error happens will be small, and the overall residual energy will be proportional to this area and the magnitude of the edge. In fact, using a small angle approximation, the error will be proportional to the error term θ.

Note this gives us a simple rule: If the edge has a large magnitude, then we should use more modes, as encoding inaccuracy area will be significant and we wish it to be as small as possible. If the edge has a small magnitude, then we should use fewer modes, as even if the area doubles in size, the increase will not be as significant.

A similar idea comes up in the context of motion compensation. If we are encoding a block with an edge and the motion vector is off orthogonal to the direction of the edge, the error will be proportional to the motion vector imprecision. An example of this is shown in Figure 2-10. A similar idea as before may thus be derived, where we choose our motion vector precision based on the magnitude of the edge in the previous block.

In this thesis, we will develop a theoretical framework to take this into account. We will then propose methods to select between different precisions for these side-

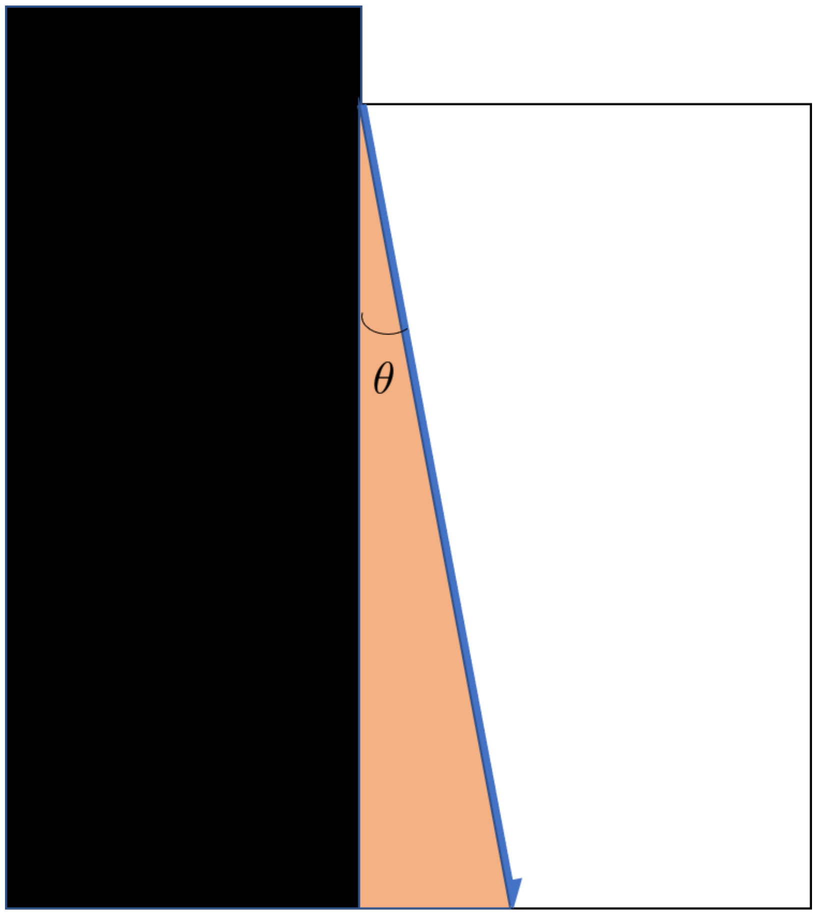


Figure 2-9: Prediction error as a function of angular inaccuracy θ is shown in orange. In this image, a black region and a white region are separated by a vertical edge. The prediction direction chosen has an inaccuracy of θ. As can be seen, the inaccurately predicted area will be proportional to $sin(\theta)$, or θ using a small angle approximation.

information parameters in a block-by-block basis.

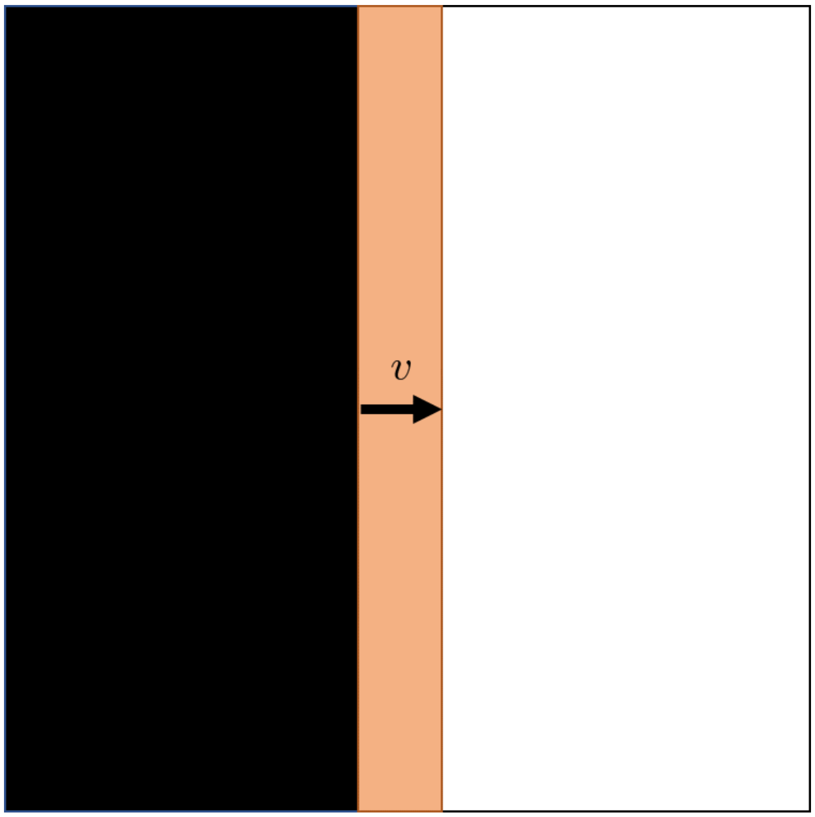


Figure 2-10: Prediction error as a function of motion vector inaccuracy v is shown in orange. In this image, a black region and a white region are separated by a vertical edge. The motion vector chosen has an inaccuracy to the true motion vector of v. As can be seen, the inaccurately predicted area will be proportional to v.

Chapter 3

Previous Research

In this thesis, the main goal is to understand how adapting the bitrate of the prediction parameter may be useful to reduce the overall encoding bit-rate. In this Chapter, our focus will be on prediction inaccuracy modeling, as the variance derived by these models will be used in the algorithms developed in Chapters 5, 7, and 8. Specifically, we will consider how prediction inaccuracy has been modeled in the context of intra-prediction residuals in Section 3.1, and in the context of motion compensation residuals in Section 3.2.

3.1 Prediction Inaccuracy Modeling for Intra-Prediction Residuals

Consider a typical intra-prediction residual frame for a given image, as shown in Figure 3-1. As can be seen, the areas with high residual magnitude are typically highly correlated to the areas where edges occur in the image. In smooth areas of the image, the spatial correlation between neighboring pixels is high. Therefore intra-prediction is able to reduce most of the energy of the original signal. In the figure, this can be seen by noting that the residual magnitude is small in the smooth background area. Meanwhile, in non-smooth areas such as edge regions or textures, the spatial correlation between neighboring pixels is much smaller. Therefore, intra-prediction

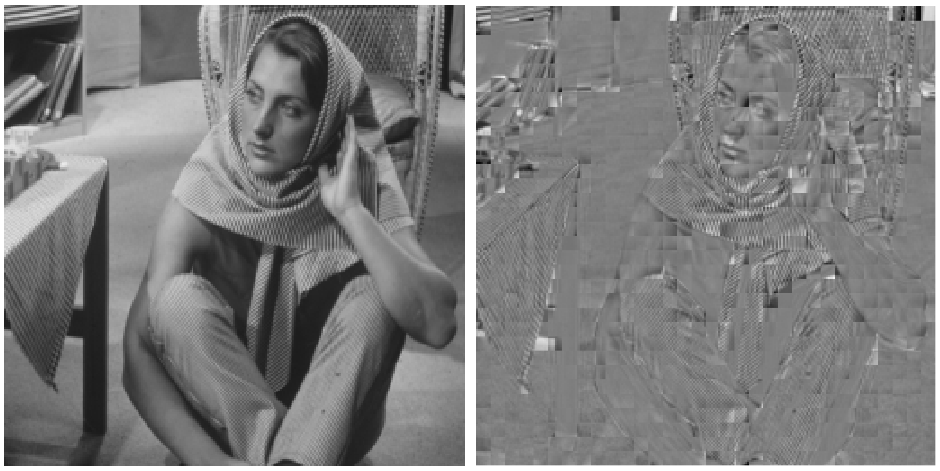

(a) Barbara Image. (b) Intra-prediction Residuals.

Figure 3-1: Image (b) shows the intra-prediction residuals for image (a). As can be seen, intra-prediction residuals tend to be large in noisier areas in the original images, such as edges.

will not be able to reduce as much of the energy. An example of an area where this can be seen in the figure is the texture on Barbara's clothes.

The prediction inaccuracy model proposes a model to explain this effect. In [18, 19], this model is proposed and a transform is chosen so the transform coder is optimized around this model. We will describe this statistical model in this section. Let $f(m,n)$ denote the block we wish to encode, for $0 \leq m,n < N$ and some fixed block-size parameter N. Assume we wish to predict pixel (m,n), and that we use fixed horizontal prediction. Then we predict it from the boundary pixel $f(0,n)$. This is represented graphically in 3-2.

In this case, the prediction residual will be:

$$r(m,n) = f(m,n) - f(0,n) \tag{3.1}$$

If we consider the block as containing a single linear edge, which holds for small enough block-size N, then the block may be encoded with a perfect predictor by using the edge direction as our prediction direction. Therefore, for the correct prediction direction, we can take the pixel to be approximately equal to the optimal boundary estimate $(0,n_a)$. Note that n_a is not equal to n, as we are simply approximating the

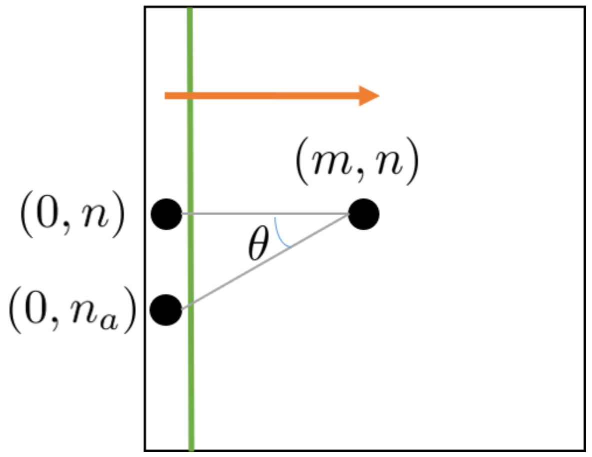


Figure 3-2: Geometric representation used to find the intra-prediction residuals at pixel (m, n) as a function of the block boundary.

true prediction direction, but should be around n if the horizontal direction was the appropriate choice from our given set of directions. We thus write $f(m, n) \approx f(0, n_a)$. Using the previous equation, and linear approximation of f around $(0, n)$, we find:

$$r(m, n) \approx f(0, n_a) - f(0, n) \approx (n_a - n)\frac{\partial f(0, n)}{\partial n} \tag{3.2}$$

Now assume the optimal prediction direction for this pixel is characterized by a random angular variable θ as we show in Figure 3-2, randomly distributed over all directions close to horizontal. Using a small angle approximation, we find that:

$$n_a = n + m\tan(\theta) \approx n + m\theta \tag{3.3}$$

Thus the prediction inaccuracy model gives us an explicit formula for our residual $r(m, n)$ as follows:

$$r(m, n) \approx m\theta\frac{\partial f(0, n)}{\partial n} \tag{3.4}$$

Using this formula, we can find $\sigma_r^2(m, n)$, the variance of the residual signal $r(m, n)$ at pixel (m, n). This is given as a function of the variance of parameter θ, denoted here by σ_θ^2, where we obtain:

$$\sigma_r^2(m,n) = m^2\sigma_\theta^2 \left(\frac{\partial f(0,n)}{\partial n}\right)^2 \tag{3.5}$$

We could extend this model to find the correlation between pixels given the used prediction direction, as is done in [18, 19]. A similar model can be derived for non-horizontal prediction directions, obtaining similar results. Through this correlation matrix, the Karhunen-Loeve transform (KLT) was used, and thus an adaptive transform was obtained. In [20], this transform was applied to H.264 intra-prediction residuals, and this significantly improved compression performance for 4×4 blocks as well as other block-sizes. As will be shown in chapter 5 of this thesis, we may use the intra-prediction inaccuracy model as a way to characterize how the accuracy of our prediction parameter, here represented by the term σ_θ^2, affects the prediction residual variance.

3.2 Prediction Inaccuracy Modeling for Motion Compensation Residuals

We similarly consider an equivalent model proposed for motion compensation residuals instead in [21]. In the case of motion compensated residuals, we note that the areas of high residual magnitude are still highly correlated with edges in the frames. In large edge areas of a frame, the motion vector components need to be precise to take into account the finer edge structure.

Let $f(m,n)$ be a pixel we wish to predict, where m denotes the horizontal coordinate and n denotes the vertical coordinate. Let $\widehat{f}(m,n)$ be the previous block used to predict it by assuming affine motion with a motion vector $v = (m_v, n_v)$. Let $r(m,n)$ denote the residual pixel obtained via:

$$r(m,n) = f(m,n) - \widehat{f}(m - m_v, n - n_v) \tag{3.6}$$

Now, assume the motion vector was not chosen perfectly. Let the vector $\widehat{v} = (\widehat{m}_v, \widehat{n}_v)$ represent the true motion vector. Then we have a small displacement from

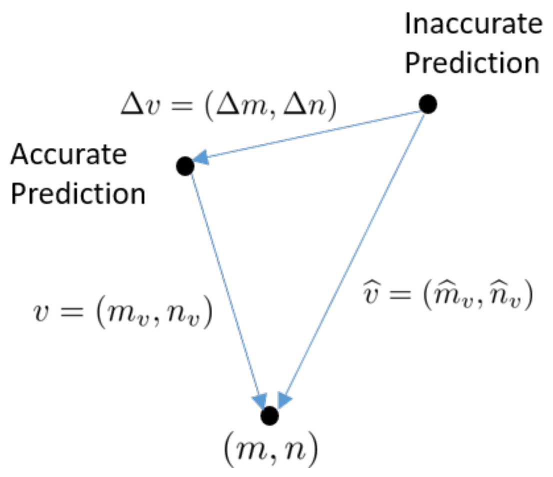

Inaccurate Prediction
$\Delta v = (\Delta m, \Delta n)$
Accurate Prediction
$v = (m_v, n_v)$
$\widehat{v} = (\widehat{m}_v, \widehat{n}_v)$
(m, n)

Figure 3-3: Used motion vector $\widehat{v}$ is off from the true motion vector v by Δv.

the correct motion vector given by:

$$\Delta v = \widehat{v} - v \tag{3.7}$$

We also write $\Delta v = (\Delta m, \Delta n)$. These vectors are represented graphically in Figure 3-3. If we assume the correct motion vector to be an accurate representation of the current scene:

$$\widehat{f}(m - \widehat{m}_v, n - \widehat{n}_v) \approx f(m, n) \tag{3.8}$$

Then using the two-dimensional Taylor series expansion for a small $(\Delta m, \Delta n)$ we find:

$$r(m, n) \approx \widehat{f}(m - \widehat{m}_v, n - \widehat{n}_v) - \widehat{f}(m - v, n - v) \approx \Delta m \frac{\partial \widehat{f}}{\partial m} + \Delta n \frac{\partial \widehat{f}}{\partial n} \tag{3.9}$$

Under this formula, we can find the residual variance to be equal to:

$$\sigma_r^2(m, n) \approx E[(\Delta m)^2] \left(\frac{\partial \widehat{f}}{\partial m}\right)^2 + E[(\Delta n)^2] \left(\frac{\partial \widehat{f}}{\partial n}\right) \tag{3.10}$$

This model has been used to derive a correlation matrix as in the intra-prediction case, and thus a new transform for motion compensation residual was derived based on previously processed information. Note the dependence of the residual variance on the prediction inaccuracy for both motion vector components. This will be considered in the context of defining the motion compensation parameter accuracy in Chapter 8.

Chapter 4

Theoretical Framework For Side-Information Reduction

In this Chapter, we will propose a theoretical framework through which we will analyze side-information reduction. The objective of this framework is to provide a theoretical setup, through which one can design algorithms that adaptively trade-off side-information accuracy and overall encoding bit-rate.

First, we will consider a general framework for prediction with side-information. Our setup has some signal $x(n)$ as input, which we wish to transmit. This signal may consist of rectangular blocks from an image or blocks of an audio signal for example. The encoder then forms a prediction of this signal $\widehat{x}(n)$, from which a residual signal $r(n)$ can then be computed as:

$$r(n) = x(n) - \widehat{x}(n) \tag{4.1}$$

We require this prediction to be known in both the encoder and the decoder by design. This residual signal is then encoded and transmitted. In the case of image or video compression, the residual signal is encoded by transforming, quantizing and entropy encoding, as described in Chapter 2. This generates a binary string $b_r(n)$. This string is then decoded, forming an approximation to this residual $\widetilde{r}(n)$. The decoder knows the prediction $\widehat{x}(n)$, and forms a decoded signal $\widetilde{x}(n)$ via:

$$\widetilde{x}(n) = \widehat{x}(n) + \widetilde{r}(n) \tag{4.2}$$

This signal $\widetilde{x}(n)$ is known at both the encoder and the decoder, as $\widetilde{r}(n)$ and $\widehat{x}(n)$ are known at both the encoder and the decoder. Therefore, the prediction $\widehat{x}(n)$ is formed based on the decoded blocks $\widetilde{x}(n-1)$, $\widetilde{x}(n-2)$ and so forth, and on a prediction parameter $\alpha(n)$. We denote the predictor as H_α, or:

$$\widehat{x}(n) = H_\alpha(\widetilde{x}(n-1), \widetilde{x}(n-2), \ldots) \tag{4.3}$$

In the case of intra-prediction, $\alpha(n)$ refers to the prediction angle θ in each block, while in the case of motion compensation, $\alpha(n)$ refers to the motion vector used for each block. This setup is shown in Figure 4-1. Intuitively, the goal of this setup is to highlight the role of prediction in the encoder process.

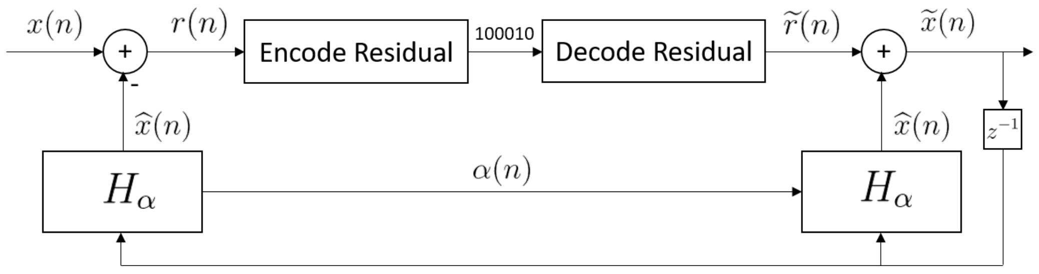


Figure 4-1: Simplified prediction system based on side-information.

As mentioned before, this encoding is called lossless if $\widetilde{x}(n) = x(n)$, or equivalently $r(n) = \widetilde{r}(n)$. Otherwise it is called lossy. In this thesis, we will focus on lossy encoding. For lossy encoding, note that there are two parameters we wish to optimize. First, we wish to minimize the error between $x(n)$ and $\widetilde{x}(n)$. Second, we wish to minimize the encoding bit-rate, given by:

$$b = \sum_n |b_r(n)| \tag{4.4}$$

where $b_r(n)$ is the number of bits used to encode the residual in each block. The joint optimization is generally done through a rate-distortion optimization:

$$\sum_n ||x(n) - \widetilde{x}(n)|| + \lambda b \tag{4.5}$$

The setup so far does not consider any side-information bit-rate. In other words, it assumes transmitting the value of $\alpha(n)$ does not add any cost. This is not a valid assumption, as if that were the case, we could just make $\alpha(n) = x(n)$, and we would have a total of bits $b = 0$ and perfect reconstruction as $x(n) = \widetilde{x}(n)$.

To take the side-information bit-rate into account, we let $b_\alpha(n)$ be the number of bits required to encode $\alpha(n)$, then our metric becomes:

$$\sum_n ||x(n) - \widetilde{x}(n)|| + \lambda b + \lambda \sum_n b_\alpha(n) \tag{4.6}$$

The main question is how we encode $\alpha(n)$. We begin with a set of valid values for $\alpha(n)$, denoted by A. The value $\alpha(n)$ is encoded based on a prior from the previous values of $\alpha(n)$. In HEVC, this is done through most probable modes (MPMs) for intra-prediction bits. The 3 most likely modes are encoded using 2 or 3 bits, while each of the other 32 intra-prediction modes uses 6 bits instead.

Another issue is choosing the value of $\alpha(n)$. The encoder needs to have a decider block, which observes the current $x(n)$ and selects a value of $\alpha(n)$ to encode it. There is a wide literature in how to choose the side-information parameter for intra-prediction and motion compensation, especially in terms of how this process can be made computationally efficient. We will not discuss it in detail here.

Figure 4-2 shows the extended prediction system from Figure 4-1 when we take the side-information bits into account. In this setup, our main objective is to make clear the role side-information plays in the encoder, and how we can take into account the fact that this may add up to a significant amount of the bitrate.

This setup makes one fundamental assumption about the side-information parameter: It is a discrete value from the fixed set A. Although we may change our prior to try to encode it more efficiently, there are still better ways to encode it. In this thesis, we consider adapting the set A instead. For example, in the case of directional intra-prediction, the set of possible angles is $\Omega = \left[-\frac{3\pi}{4}, \frac{\pi}{4}\right]$, but we limit ourselves to

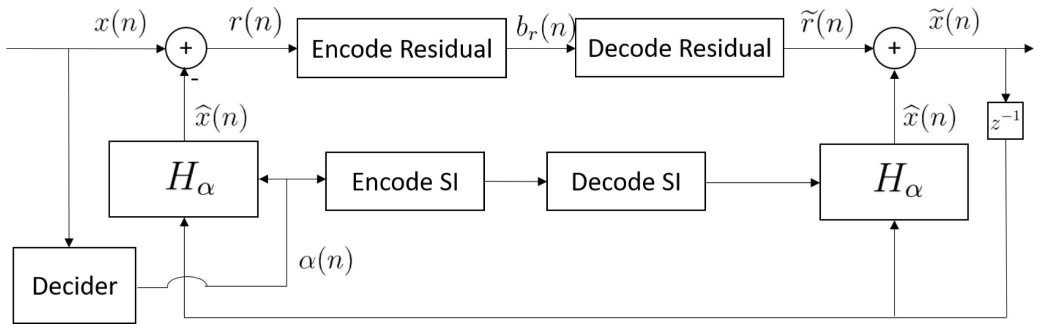


Figure 4-2: Simplified prediction system based on side-information taking side-information bitrate into account.

a set A of 33 prediction directions from Ω.

In practice, this set A is not required to be a fixed set. For example, A can depend on information from the previous block, as long as it is information that is synchronized in the encoder and the decoder. If we do not expect the current block to have much varying information, we can use a smaller set of prediction directions for example, or use fewer bits in motion compensation. In other words, we can allow ourselves to have two deciders, one for the set A and one for the side-information α. This gives us a sequence $A(n)$ of sets used in each block

For our encoder design, we assume we do not wish to add any side-information bits to encode the set $A(n)$, as this would be another source of side-information. In other words, we require the new set to be decided only using information based on the decoder. By imposing this constraint, we simplify the system to be designed, and avoid an aggregating effect of adding multiple layers of side-information on side-information. This setup is shown in Figure 4-3, and proposes manipulating the set of available prediction directions as a function of the information in the previously processed blocks.

The main problem in this setup is how to find the ideal set $A(n)$ for each block, observing only previously encoded blocks. It is very hard to find what the optimal set $A(n)$ would be, so throughout this thesis we will derive a few setups that are optimal under some simplifying conditions. The main goal of these setups is to show that methods of this class can lead to encoder improvement, and indicate how further studies in side-information reduction may lead to even larger improvements in encoder

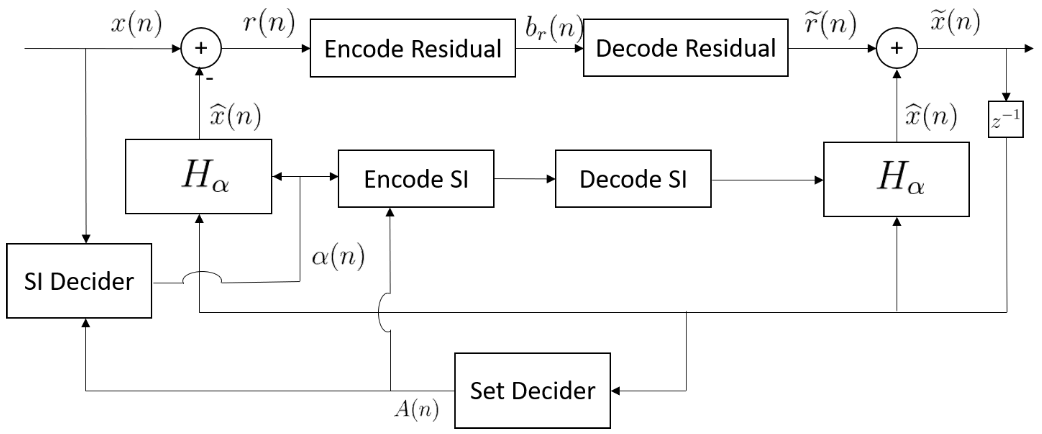


Figure 4-3: Prediction system based on side-information with decider for set of prediction parameters.

performance.

As a simplification of how to choose the optimal A in a given block, we will consider this problem in two steps. First, what is the optimal size of A, here denoted by $N = |A|$. Once we know the value of N, what is the best way to choose N possible values for our side-information parameter?

An initial way to observe this system is via rate-distortion optimization. We have the objective of minimizing, over all possible values of the set A:

$$||r(n)||^2 + \lambda \left(b_{SI} + b_r\right) \tag{4.7}$$

where b_r are the bits required to encode this residual r. Let us assume the bits used to encode the residual are approximately constant. Although this assumption is not true in practice, we will use it in this section as an approximation to ensure our problem is mathematically tractable. Under this condition, we can simply focus on minimizing:

$$||r(n)||^2 + \lambda b_{SI} \tag{4.8}$$

In other words, we are approximating the previous optimization problem with minimizing the residual energy while still having as few bits as possible for side-information. If the choice of intra-prediction modes is uniformly distributed within a set A, we will have $b_{SI} = \log_2(N)$. Assuming this approximately holds, our problem can be written as:

$$\min_{N}\left(\min_{|A|=N}||r(n)||^2\right)+\lambda\log_2(N) \tag{4.9}$$

In this thesis, our main focus will be on finding the value of N. We will now relate this problem to quantization. Assume we have the true optimal value of the side-information parameter α to be denoted by α^* as:

$$\alpha^* = argmin_{\alpha}||r(n)||^2 \tag{4.10}$$

Assume the encoder chooses to use an inaccurate prediction parameter $\widetilde{\alpha}$ instead. We say the residual energy has proportional error to the prediction parameter inaccuracy if we can write, for some constants $c()$:

$$E[||r(n)||^2|\widehat{\alpha},\alpha^*] = c(\widehat{\alpha})(\widehat{\alpha}-\alpha^*)^2 \tag{4.11}$$

In other words, this means the residual energy is proportional to how far our parameter is from the true value. In particular, assume $c(\widehat{\alpha})$ is a constant. This condition is interesting, as our problem becomes equal to classical quantization. In quantization, if we wish to encode a variable x as $\widehat{x}$, the error will be:

$$r^2 = (x-\widehat{x})^2 \tag{4.12}$$

This problem has been widely studied, as we have presented before in Chapter 2, and solutions have been presented in many different conditions. As a particular case, if x is uniformly distributed, then the solution is to divide the energy uniformly.

Going back to our original problem, we can see our problem as quantizing the parameter α, but where instead of caring about the true value of α, we really wish to obtain $r(n)$. We therefore refer to this problem as an indirect quantization problem. An interesting way in which our problem differs from classical quantization is that the Voronoi regions from the α may not be connected as they were before. This leads to our problem being a lot harder to optimize. Even under the simplifying condition in (4.11), finding the solution to Equation (4.9) is non-trivial, as it also depends on the prior on α^* and the constants $c(\alpha)$.

A natural question would be how restrictive the residual energy having proportional error to the prediction parameter inaccuracy is. In fact, we will show that, under the prediction inaccuracy model, this will be true both for intra-prediction and motion compensation residuals.

To summarize our theoretical development, there are two main issues with finding the optimal set to be used for a different block, from Equation (4.9):

(a) How do we find the size N of the set $A(n)$?

(b) Once we know $N = |A(n)|$, how do we choose $A(n)$ for a given block?

In this thesis, we will consider some examples of solutions to both of these problems. Our first approach will consider two possible sets (one with many elements, and one with few elements), and show that there is some gain to be had when using a method to reduce intra-prediction side-information based on a simplified setup. It will assume that, given N, the intra-prediction modes will still be distributed mostly uniformly. This method is presented in Chapter 5, and its results in Chapter 6.

Chapter 7 presents two methods to further reduce side-information for intra-prediction. The first method will consider using more than two sets, and offer multiple alternatives to the value of N, while still assuming uniform distribution. The second method will consider two sets with fixed values of N, but instead propose adapting $A(n)$ based on the boundaries. Finally, Chapter 8 will present a solution to the issue of finding the value of N in the context of motion compensation, and its results will be shown in Chapter 9.

Summary

In this Chapter, we first observed how many compression systems use a parameterized predictor to reduce the energy in the transmitted signal. This parameter is chosen at the encoder from a fixed set of parameter values A, and transmitted as side-information bits. As mentioned in Chapter 2, this set has been increasing in recent compression standards both in the case of intra-prediction and motion compensation. This increase leads to an increase in the number of side-information bits used.

We proposed to consider instead a new setup for this set A. Instead of having the set be fixed, we proposed adapting the set in a block-by-block basis. The decision of which set to use must be done using only information available in the decoder, as we do not wish to add additional side-information bits. We compared the decision of this set to the quantization problem. We then summarize the decision of this set $A(n)$ based on two questions. First, what should be the cardinality N of this set $A(n)$? Given a cardinality $N = |A(n)|$, what should be the parameter values that compose $A(n)$?

Chapter 5

Intra-Prediction Side-Information Reduction

In this section, we consider a preliminary algorithm, through which we show that side-information reduction can lead to an overall improvement in image encoding. We first presented this algorithm in [22]. We begin by reconsidering equation (3.5) with respect to intra-prediction. This equation gives us:

$$\sigma_r^2(m,n) = m^2\sigma_\theta^2\left(\frac{\partial f(0,n)}{\partial n}\right)^2 \tag{5.1}$$

In this equation, $f(m,n)$ denotes the current block, where $f(0,n)$ is one of its boundaries. The angle θ represents the difference between the true direction of the edge in the image and the prediction direction.

Note that, if the gradient boundary has a small magnitude, the residual will have a smaller variance. This is consistent with our intuition that if the boundary is smooth, our block is most likely also smooth, and the prediction direction inaccuracy will not be as significant. On the other hand, if the gradient boundary has a large magnitude, the residual will have larger variance. This is consistent with the observation that there is likely an edge in such a block, and residuals are concentrated along edges.

Note also that θ is related to the predicted direction chosen. We can assume initially that the true edge is an angle $\phi \in [\frac{-3\pi}{4}, \frac{\pi}{4}]$. In theory, we can see choosing the prediction direction as trying to find the angle from our given set that is closest to ϕ. Therefore, we can see the prediction inaccuracy σ_θ^2 as decreasing with the number of prediction directions used.

We will considering this equation from a rate-distortion perspective. In rate-distortion optimization, we wish to minimize:

$$\sigma_r^2(m,n) + \lambda b_{SI} \tag{5.2}$$

Thie idea behind rate-distortion optimization in this case is that λ will estabilish a trade-off between bits of side-information and the residual variance. Since we wish to minimize this sum, adding a bit of side-information will only be correct if we reduce the residual variance by a value that is greater than λ. Applying what we had from the previous equation, we find:

$$\sigma_r^2(m,n) + \lambda b_{SI} = m^2 \sigma_\theta^2 \left(\frac{\partial f(0,n)}{\partial n} \right)^2 + \lambda b_{SI} \tag{5.3}$$

This therefore establishes a trade-off between the bits of side-information needed to encode θ and the gradient boundary. If the gradient is large, we compensate for its large value by using a larger number of prediction directions, or in other words a large value for λb_{SI}. If the gradient is small, we take that into account by using a smaller number of prediction directions, reducing the value of λb_{SI} instead.

Therefore, we conclude that we should adapt the precision of the parameter θ as a function of its gradient. When the gradient boundary has a small magnitude, we will use a smaller set of 7 prediction modes. The seven modes are given by 5 directional modes (2, 10, 18, 26, and 34 from HEVC) shown in Figure 5-1, plus planar mode and DC mode. On the other hand, when the gradient boundary has a large magnitude, we will use the full HEVC set of 35 intra-prediction modes. This will reduce variance of θ when appropriate, and we can use such a method to keep the overall variance as small as possible. To determine whether a block has large or small magnitude, we simply compute its gradient boundary and compare the largest magnitude to a fixed threshold parameter β. We compute the gradient by using single differences, as in

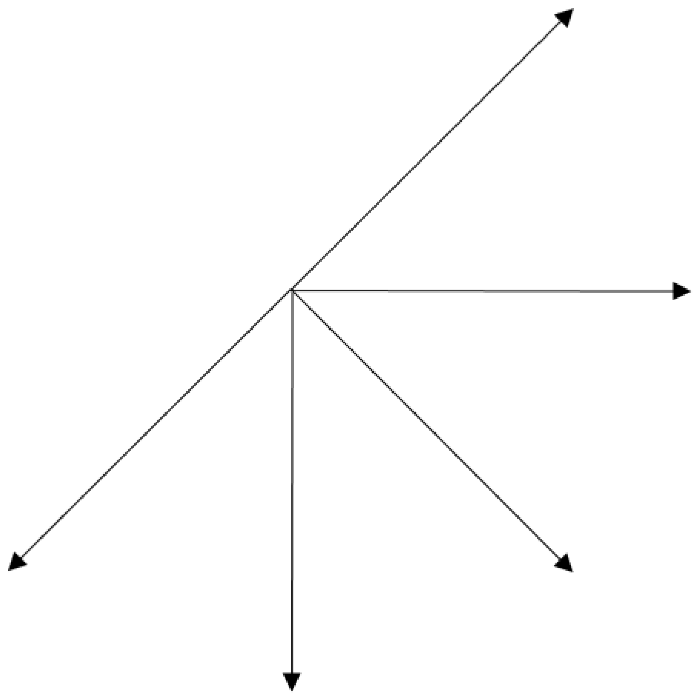


Figure 5-1: Set of reduced intra-prediction directions used in blocks with small gradient boundary magnitude.

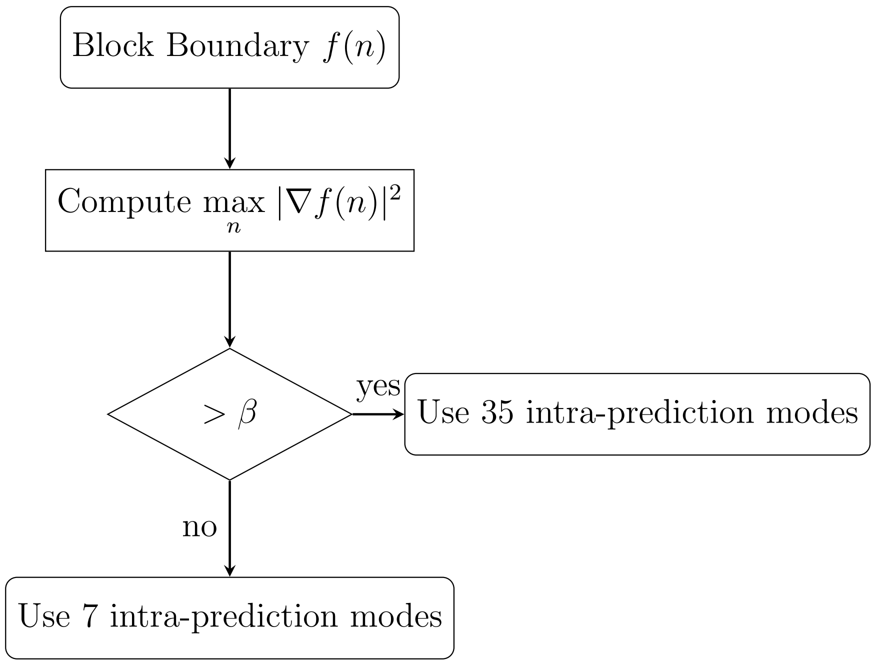

Block Boundary $f(n)$
Compute $\max_n |\nabla f(n)|^2$
$> \beta$
yes
Use 35 intra-prediction modes
no
Use 7 intra-prediction modes

Figure 5-2: Algorithm decides whether or not to use a smaller set of intra-prediction modes based solely on the gradient boundary and threshold parameter β.

[19]. This algorithm is displayed in Figure 5-2.

There are a few design questions that may be raised here. First and foremost, why did we specifically choose the maximum absolute gradient? We consider a block's boundary to consist of an edge with noise added to it. From a theoretical perspective, in this case we can model the gradient as a noisy impulse:

$$A\delta(n-k,0)+w(n_1,n_2) \tag{5.4}$$

where $\delta(n)$ represents the one-dimensional discrete impulse function, A is an amplitude term referring to the magnitude of the edge, k is the position of the edge, and $w(n_1, n_2)$ is a noise term. This noise can be added to it due to textures or simple camera noise. By observing the maximum value of our gradient boundary, we would approximately be observing the value of A, as long as our noise has zero mean. This is especially relevant when we are adding the variance contributed by each of the pixels in the image.

Under this perspective, what is the magnitude of the predicted edge inside the block that we are trying to encode? If this magnitude is large, we need to be as precise as possible, as the area for which the direction does not match the true edge will lead to high-magnitude residuals. If this magnitude is small, we may still need to use directional intra-prediction, as an edge could be present, but we do not need to have it as precise, as capturing an approximate edge direction will already encapsulate most of the associated energy.

Second, what is the effect that the most probable modes would have in our system? As is well known, not all codewords use the same number of bits. Therefore, we need the codebook to adapt accordingly. To do so, we need to compute the optimal MPM for the previous block for the decoded residuals. This will update our estimate of the MPMs. This will increase our computations significantly. There are alternative designs that can be done to reduce this computation. For instance, we could estimate the MPMs based on edge detectors, or on alternative fast methods. We should also note that if the MPMs were always correct, there would be no reason to use our method, as side-information would already be reduced. This method provides an adaptive approach to be able to improve in the blocks where MPMs do not capture the correct mode.

Another effect the MPMs will have is that sometimes the best mode may not be selected due to their large code size. In this scenario, the first MPM could have a higher probability of being selected simply by having a smaller number of bits. This is especially true when the value of λ is small. Our method will in fact increase the chance a directional mode is selected, as it requires fewer bits to be encoded.

Finally, we note how the system illustrated here relates to the theoretical setup presented in Chapter 4. As described in Chapter 4, one of the main issues is finding the optimal set size to be used in each block. This model assumes that the set size N can take two levels (7 or 35), based on the known boundaries. The set $A(n)$ is obtained from N by dividing the interval approximately uniformly, and adding DC and planar modes. This assumes the modes will have approximately uniform probabilities, if we do not consider the effect of the most-probable modes.

Summary

In this Chapter, we proposed an algorithm to reduce side-information adaptively in a block-by-block basis for intra-prediction. We considered selecting whether a block should use the full set of 35 intra-prediction modes, or a reduced fixed set of 7 intra-prediction modes based only on thresholding the absolute gradient boundary. We designed this decision criterion based on prediction inaccuracy modeling. We also explained how most probable modes will interact with this algorithm. The results of applying this algorithm to an HEVC-based image compression system is shown in the next Chapter.

Chapter 6

Intra-Prediction Side-Information Reduction: Initial Results

In this section, we will discuss the main results we obtained in our initial experiments. Section 6.1 will go over the experimental setup used, while Section 6.2 will go over the results and our insights, as well as suggestions for future improvement. Section 6.3 will offer a discussion on the computational complexity of our method.

6.1 Experimental Setup

In the experiments performed, we implement our method in a simplified version of HEVC. This version assumes a fixed block-size used to encode an image, chosen from the possible sizes of 4×4, 8×8, 16×16 or 32×32. Each block is then intra-predicted by choosing the block that minimizes the rate-distortion based sum of absolute transformed differences (SATD) plus λb_{SI} metric, as in HEVC. The SATD is chosen as it approximates the overall encoder performance, and is computationally efficient [23]. The rate-distortion metric is used to take into account the fact that often one should not choose the mode that minimizes the SATD, as it can incur a significant side-information bit-rate addition.

The codewords chosen for each intra-prediction mode are determined by an MPM flag, and uniform encoding for non-MPMs. The non-MPMs are encoded using uniform encoding, while the MPMs use a single additional bit, and the second and third MPMs use two additional bits.

Once we have obtained the prediction, we subtract it from the given block. We then encode the prediction residuals. To encode it, first we transform the residuals using the integer 2-D ADST for 4×4 blocks or the integer 2-D DCT for larger blocks. The transformed blocks are then quantized and entropy encoded as determined by the HEVC standard. The blocks are decoded using the inverse entropy encoder and inverse quantizer, followed by the inverse predictor. For our analysis, we did not include the deblocking filter, as we do not believe it will significantly change whether such a method is viable.

In the experiments we perform, we test sequences in the kodak and derf datasets, with resolutions of 256×256, 512×512, and 1280×720. We process only the Y channel, and apply our method to only this channel for the first frame of the video sequences. The sample sequences are shown in Figures 6-1 and 6-2.

6.2 Results

We ran the algorithm in Figure 5-2 on the previously defined experimental setup. We also have to choose a value of β for each configuration. To do so, we iterate over values of β with small increment, and choose the β that best fits the associated rate-distortion metric for each image and QP level. This would be equivalent to transmitting a value of β for each image, which would result in a negligible increase in bit-rate as it is sent once per sequence. We evaluate our algorithm in a large set of QP values, from 25 to 50 with an increment of 2, so we can understand how our algorithm behaves in multiple scenarios. We assume transmitting a single value of β for an image would be negligible with respect to the total bit-rate.

We now consider in which scenarios we would expect our algorithm to perform better. In very smooth regions of an image, both our method and the original encoder should just choose a most probable mode, as it will use fewer side-information bits and any further prediction will not lead to a significant prediction residual reduction.

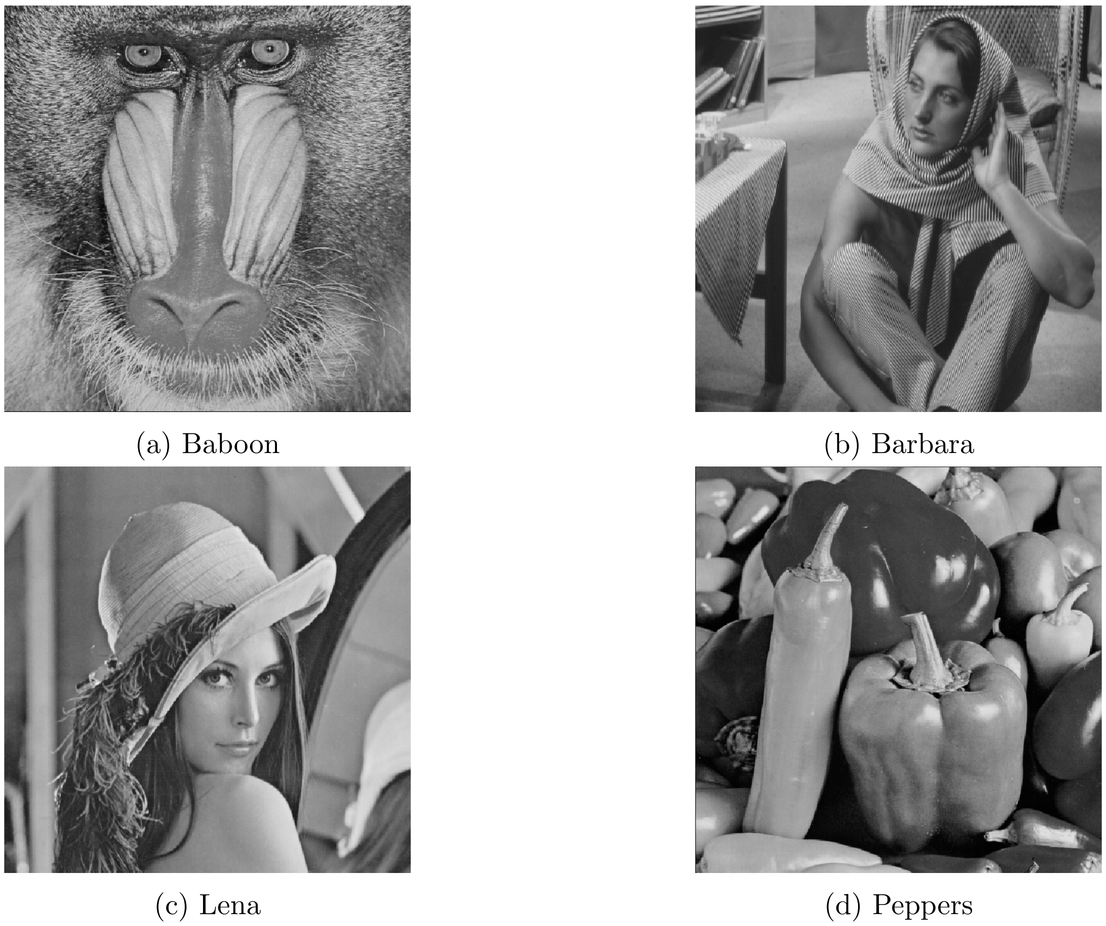
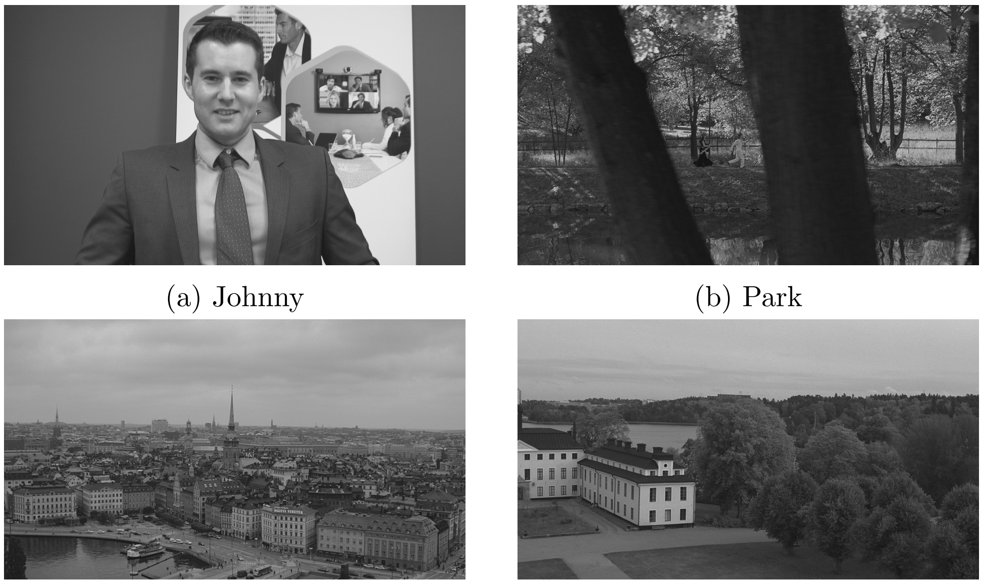


Figure 6-1: Test images chosen to represent many possible features the algorithm could adapt to. These images will be used in resolutions of 256×256 and 512×512.

In regions with small-magnitude edges, while using some sort of directional prediction is valuable, we do not need to be extremely precise with our prediction algorithm; therefore, as we predicted before, we expect our method to profit in these regions. In regions with large-magnitude edges, we will need to use a larger number of prediction directions, as error caused by prediction inaccuracy will be large. These scenarios can also be considered from the viewpoint of equation (3.5), and the same results may be observed.

We now examine in more detail why our algorithm should lead to an improvement in encoder performance from the perspective of small magnitude edges and the most probable modes (MPMs). If we consider small magnitude edges in the image, using a directional intra-prediction mode that is reasonably close to the true mode will


(a) Johnny (b) Park

(c) Town (d) Tree

Figure 6-2: Test images of size 1280×720, chosen to represent many possible features the algorithm could adapt to.

already give us most of the energy reduction. In this scenario, using a smaller number of directional intra-prediction modes would encode most of the energy of the residuals, while using a small number of side-information bits when compared to the full set of 35 intra-prediction modes. Similarly, in many blocks using a direction that is reasonably close to the true direction may require the use of a non-MPM. In this case, this means that a larger codeword will be required to encode such a direction. As the encoder decides the intra-prediction direction in a block through rate-distortion optimization [24], this may lead to selecting the first MPM instead of another mode as it has the smallest codeword. When using a small set, the previous direction may use a shorter codeword instead. Although this codeword might still be larger than that of the MPM, the cost for selecting it instead of the first MPM is often smaller. In this case, using a smaller set of intra-prediction modes will increase the side-information bit-rate, but will lead to a smaller rate-distortion metric, as it can capture more of the energy of the signal through an appropriate intra-prediction direction.

To better explain the effect MPMs have on a particular block when using our method, we will consider the metric of SATD plus λb_{SI}. Let A be the set of modes

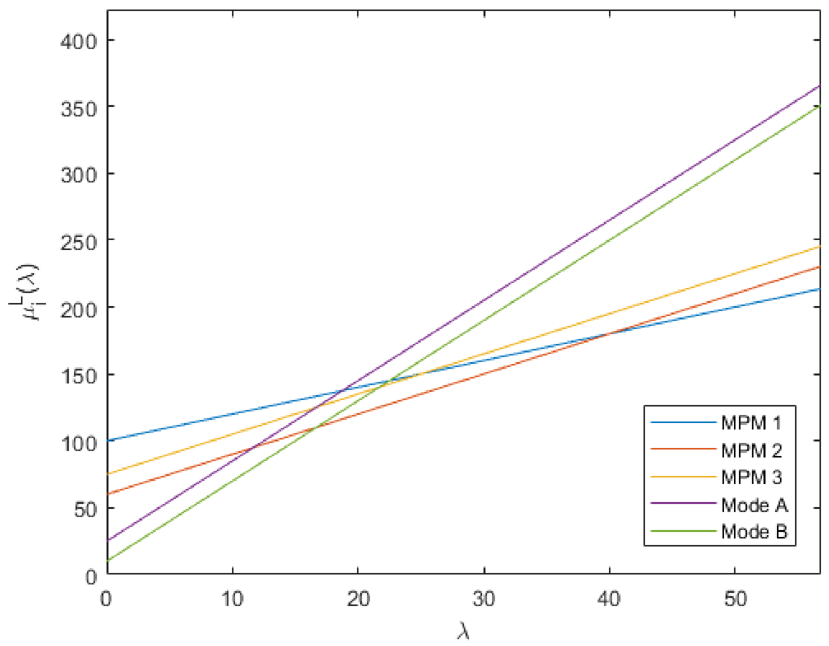


Figure 6-3: Example of metric μ_i^L for different modes. The MPMs use a smaller number of bits (2 or 3) than the other modes, therefore they will tend to be chosen as λ increases.

used in a block. For this section, A may take the values of L or S, indicating the large and small set respectively. Let $\mu_i^A(\lambda)$ denote the metric of SATD plus λb_{SI} when the mode $i \in A$ is used. For this analysis, we will assume this metric directly relates to the encoder performance. Take a fixed mode $i \in A$, then either i is an MPM, in which case b_{SI} will either be 2 or 3, or i is not an MPM, in which case $b_{SI} = \log_2(|A|-3)+1$. This means that $\mu_i^A(\lambda)$ will be a line that intersects the y axis at the SATD metric when using mode i, and incline given by $b_S I$ of either 2, 3 or $\log_2(|A|-3)+1$. An example of what these curves would look like for the large set L is shown in Figure 6-3.

For each block, we then have $|A|$ possible modes to be considered. Given a value of λ, the mode that should be chosen i_A^* is given by:

$$i_A^* = \arg\min_i \mu_i^A \tag{6.1}$$

In this plot, If we choose to use this mode as a function of λ, this also gives us a curve of the optimal metric, which would be:

$$\mu^A(\lambda) = \min_i \mu_i^A(\lambda) \tag{6.2}$$

This curve is continuous and piecewise linear, as it is the minimum of linear curves. It simply represents how small the metric can get by choosing the mode appropriately from our given set as a function of λ. Intuitively, this curve tells us that for small λ, the number of bits used by the side-information does not matter, as the y-intercept will be much more significant due to low variation for large values of λ. For large values of λ, you should simply choose an MPM (and the first MPM for the very large values of λ), as the incline will be much more significant then the y-intercept. In other words, the MPMs will dominate the metric for large values of λ, while we will choose the most precise mode for low values of λ, as expected from a rate-distortion trade-off. Note that there is also no option other than using an MPM or the most precise mode.

If we consider the small and large sets, we can then compare $\mu^L(\lambda)$ and $\mu^S(\lambda)$. In this case, for very small values of $\lambda \approx 0$, only the SATD will matter. Since L contains S, we note that the metric in set $\mu^L(\lambda)$ will be smaller than $\mu^S(\lambda)$ as it is simply the minimum of a larger set of numbers.

As we start increasing λ, note the incline will be more important. If the MPMs were not chosen for either of the sets, the incline for the large set will be 6, while the incline for the small set will be 3. This means it is quite likely that $\mu^S(\lambda)$ will become smaller than $\mu_L(\lambda)$ for a value of λ in this range before an MPM is reached in either block. In this case, the use of a smaller set helped offset the side-information costs.

As we increase λ further, an interesting effect happens. The large set will tend to reach the MPMs first. In this case, $\mu^L(\lambda)$ will start increasing at a factor of 2 or 3, while $\mu^S(\lambda)$ is much slower, but will increase with λ at a factor of 3. In this case, the smaller set can allow for a smaller metric, as it will be able to hold a more precise direction for a larger range of values of λ. This will end as both sets converge to MPMs and have the same incline, where they will mostly match as long as the blocks have the same MPMs for both sets.

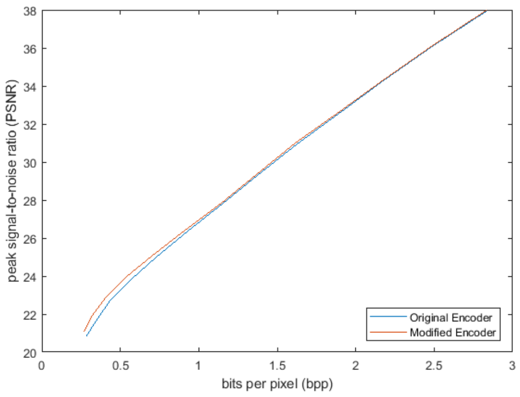
We may also consider our algorithm from the perspective of the variation of the quantization parameter QP. The definition of a large edge is based on the QP value, as the definition of the rate-distortion parameter λ is a function of QP in HEVC. As we expect from our theoretical development and was observed with our experiments, our algorithm will perform better for medium QP levels. For small QP parameters, most edges will be considered large edges, and therefore using the large set will be better. For large QP parameters, most edges will be considered small enough such that the MPM should just be used instead, therefore our method will not lead to a significant improvement. For medium values of QP, there will be a significant improvement, as these are exactly the blocks where we will see the improvement described in the previous paragraph.

Another significant consideration is the fact that we need our boundary to express whether or not an edge is contained in the current block. In this case, we should expect a higher performance to our method in situations where there is more correlation between boundary pixels and the pixels to be predicted.

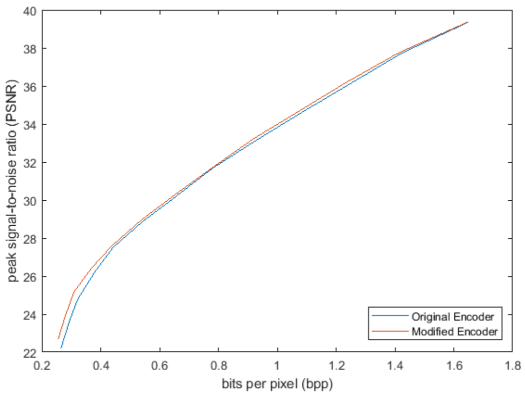
Figure 6-4 shows the peak signal-to-noise ratio (PSNR) with bit-rate for our 256×256 images when using 4×4 blocks. For a significant bit-rate range, we observe a reduction in bit-rate for the same PSNR value. As we can see in this figure, the relative improvement of our method is in lower bit-rates, as for these values, side-information is more relevant to the overall bit-rate, and thus using a smaller set of modes is more beneficial. Although we do not directly show it here, the same trends can be observed for other block-sizes and resolutions.

We now consider the results for all previously defined images, resolutions, and block-sizes. Table 6.1 indicates the BD-rate [25] improvement over the originally encoded image using our simplified version of HEVC. As a first note, we observe there is a non-negligible improvement in BD-rate on average over all images and resolutions, consistent with what we had found in Figure 6-4.

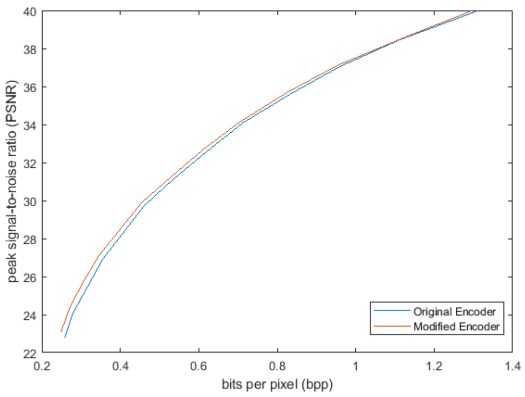
We also consider this trade-off in terms of block-sizes. As we increase the block-size, the boundary pixels will be less correlated with the pixels inside. Therefore,



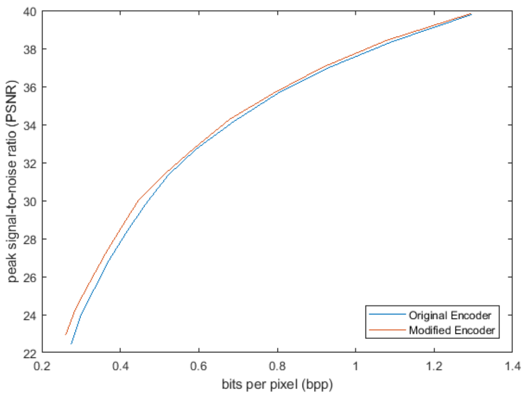
(a) Baboon



(b) Barbara



(c) Lena



(d) Peppers

Figure 6-4: PSNR as a function of bit-rate for the original simplified HEVC-based encoder and our method for 4×4 blocks on 256×256 images showed in figure 6-1. As can be seen, there is a significant improvement when using our method, especially for lower bit-rates.

they will be a worse predictor of whether or not the current block should use more prediction directions. Also, in this case, the side-information will account for a less significant fraction of the bit-rate. Thus we conclude using multiple direction sets should be less impactful as we increase blocks-size. This is verified in most sequences shown in Table 6.1.

In terms of resolution, we note this method presents a significant higher improvement for higher resolution sequences than lower resolution sequences. This can be explained from the fact that higher resolution images will have a smaller number of features in each block. We expect this to lead to even larger gain if we consider higher resolutions, such as 4k sequences.

At this point, we consider how our method would interact with the non-constant

Image Resolution	Sequence	4×4	8×8	16×16	32×32
256×256	Baboon	3.15 %	1.17 %	0.50 %	0.24 %
	Barbara	2.81 %	2.18 %	1.40 %	1.40 %
	Lena	2.90 %	2.34 %	1.03 %	0.91 %
	Peppers	3.71 %	1.53 %	1.32 %	0.70 %
256×256	Average	3.14 %	1.81 %	1.06 %	0.81 %
512×512	Baboon	2.26 %	0.80 %	0.45 %	0.36 %
	Barbara	1.86 %	1.03 %	0.93 %	0.67 %
	Lena	3.45 %	1.51 %	2.56 %	0.88 %
	Peppers	4.33 %	2.47 %	1.90 %	1.27 %
512×512	Average	2.98 %	1.45 %	1.46 %	0.80 %
1280×720	Johnny	2.09 %	0.54 %	1.42 %	0.85 %
	Park	3.76 %	0.96 %	0.57 %	0.35 %
	Town	4.76 %	1.67 %	1.05 %	0.45 %
	Tree	7.17 %	1.72 %	2.05 %	0.96 %
1280×720	Average	4.45 %	1.22 %	1.27 %	0.65 %

Table 6.1: BD-rate improvement when utilizing HEVC-based encoder with selection based only on two sets of intra-prediction directions with respect to original HEVC-based encoder. As we can see, for most of our image sequences and resolutions, there is a non-negigible improvement in BD-rate by using our method for fixed number of intra-prediction directons.

block-sizes used in HEVC. Figure **??** shows an example of how blocks could be partitioned in HEVC. As can be seen, block sizes tend to be chosen so that features tend to be the same within its pixels. This may mean that the color is the same within a block, or that an edge has constant direction within a block. We believe that there are two advantages of using a system to reduce intra-prediction side-information in the non-constant block-size case. First, there are still smaller or larger blocks that have more content than others of the same size, and we believe using such information may lead to a reduction of side-information in many of these blocks. Second, allowing blocks to use a smaller set will give the encoder more flexibility when choosing the block-sizes from small magnitude edges.

6.3 Computational Complexity

In this section, we analyze the increase in computational complexity from the current method with respect to HEVC, and propose ways to reduce it. To consider the computational complexity, we note that the typical intra-prediction system will have to compute the optimal mode once for each block. This optimal mode may be found using brute force, or using a more efficient method.

In our method, we note that there will be a decrease in the search space for modes. If 35 modes were searched before, we will have a total of 7 modes searched in the blocks with the smaller set. Meanwhile, we will still need to recompute the intra-prediction directions whenever there is a set change for the neighbors, as we do not wish our method to be fixed to directions from previous blocks.

Let α_0 denote the fraction of blocks where we compute the algorithm for full 35 modes, and α_1 denote the fraction of blocks where we compute the algorithm for 7 modes. Also let β_0 be the fraction of 35 mode blocks with a predictor neighbor using fewer modes, and β_1 be the fraction of 7 mode blocks with a predictor neighbor using the smaller set. Then the average number of modes tested per block will be:

$$35\alpha_0 + 70\beta_0 + 7\alpha_1 + 14\beta_1 \tag{6.3}$$

Therefore, we conclude our method will be more computationally efficient than the original method of always using 35 modes as long as:

$$35\alpha_0 + 70\beta_0 + 7\alpha_1 + 14\beta_1 < 35 \tag{6.4}$$

This will depend on the QP value used, as it will determine the fractions α_0, α_1, β_0 and β_1. For high QP values, we note that α_1 will tend to be much larger than α_0, so it should be more computationally efficient than using the full set in every block.

We also note that there are other ways to find an approximate optimal mode in different blocks [26, 27]. Brute force search will lead to the optimal solution, but other methods may lead to locally optimal solutions with fewer computations. In this case, it may not be always true that our method will be more computationally efficient.

Summary

In this Chapter, we presented the results when applying the algorithm in Chapter 5 to HEVC. We applied this algorithm to images from the kodak and derf datasets. As can be seen in Table 6.1, our method leads to significant encoder improvement over traditional HEVC using the fixed prediction direction set. We presented an intuitive reason of why this method leads to improvement, as well as an intuitive analysis of how this improvement should behave as a function of the QP value. We concluded with a short computational analysis, showing under which conditions our method should lead to an increase in complexity.

Chapter 7

Intra-Prediction Side-Information Reduction: Alternative Methods

In Chapter 4, we presented a theoretical setup to consider the impact of side-information on the encoder. To do so, we proposed a parameter $A(n)$, given by the set of side-information parameter values to be used in each block n. We enforced $A(n)$ to be chosen based solely on the information found in previously processed blocks. Our argument was that significant gain could be found by choosing the set $A(n)$ appropriately and adaptively. The set $A(n)$ would be optimized in a two-step fashion, where first the cardinality of the set $N = |A(n)|$ would be chosen, then the specific set $A(n)$ that satisfied $N = |A(n)|$.

In Chapter 5, we considered a first approach to obtaining this set in the context of intra-prediction. In this algorithm, we were alternating between $N = 7$ and $N = 35$, with the edge magnitude in the gradient boundary being the main deciding factor. This was shown in Chapter 6 to improve HEVC's encoder performance.

In this Chapter, we will consider alternative ways to reduce intra-prediction side-information. We note that the methods proposed here are not comprehensive, and simply meant to indicate the reader some possible directions, as well as to show that there is room for further improvement in such a system. Specifically, we will consider two approaches to the questions we defined in Chapter 4, based on the set $A(n)$ and its size $N = |A(n)|$:

(a) We initially approached our problem using only two values of N. Can we improve our system even further by having a larger set of values? In particular, if we allow N to be 7, 11, 19 and 35, would this improve the encoder performance? How would we redesign our algorithm to take this into account? This problem will be considered in Section 7.1.

(b) We chose $A(n)$ to be uniformly spaced samples from our set Ω given N. Are there better ways to design $A(n)$? This problem will be considered in Section 7.2.

We will then compare the results obtained with these two methods to the ones found in Chapter 6. In this case, we will observe how the multiple set is the method with largest improvement.

7.1 Multiple Sets Of Prediction Directions

In this section, we will consider a way through which the work presented in Chapter 5 can be generalized to include multiple sets. Our objective is to allow the number of modes used to be more flexible, as a function of the content in the previously processed blocks.

Let us consider the problem we are trying to solve here in more detail. Before, we had two sets of possible prediction directions, denoted as the large set L and the small set S. The idea was that by using the sets at the appropriate blocks, we could adapt appropriately to the magnitude of given edges. A model was derived based on the prediction direction model, and we verified that it gives us positive results.

We consider generalizing the previous setup to the case where we allow multiple sets. Assume we have a large set L, and allow ourselves to have multiple increasing sets S_1, S_2 and S_3 for instance. Under this scenario, we should at least match the results when using a single small set, as it simply will reduce to the two set case in a worst case scenario.

We will consider the approach of using 4 possible small sets, where we use 3, 4 or

5 bits of side-information in the non-MPMs and have our modes uniformly divide the set of possible modes. In other words, $|S_1| = 7$, $|S_2| = 11$, and $|S_3| = 19$. How do we generate a criterion to select among these sets? This section addresses this problem by reconsidering the prediction inaccuracy model, and proposes a new algorithm from it.

7.1.1 Theoretical Analysis

In this section, we will consider a theoretical framework to minimize the prediction residual as a function of side-information.

First, let us assume we have our signal divided into N blocks. For each block, indicated by the index n, we will use a certain number of bits b_n to encode its prediction residuals. These bits can also be written in vector form as the N-dimensional vector $\vec{b}$. Now, let us consider the prediction error in block n when using b_n bits of side-information in this block to be denoted by $e_n(b_n)$. In this case, we define the overall error $E(\vec{b})$ to be given by:

$$E(\vec{b}) = \sum_{n=1}^{N} e_n(b_n) \tag{7.1}$$

Effectively, what this means is that we wish to consider the overall prediction error as a function of the number of side-information bits used in each block. This overall error is an additive function of the error in each block. Through this framework, we can pose the side-information bit allocation as an optimization problem. How do we choose the number of side-information bits to be used in each block in order to minimize the total error $E(\vec{b})$?

We assume we have a bit-rate constraint given by a total number of bits B. We can write this as the following optimization problem:

$$\begin{aligned} \underset{\vec{b}}{\text{minimize}} \quad & E(\vec{b}) \\ \text{subject to} \quad & ||\vec{b}||_1 \leq B \end{aligned}$$

We simply wish to minimize the error over all blocks subject to a bit-rate constraint. Using a Lagrange optimization or rate-distortion approach, we rewrite our problem as:

$$\underset{\vec{b}}{\text{minimize}} \quad E(\vec{b}) + \lambda ||\vec{b}||_1$$

In this case, we are effectively associating a cost λ to the action of adding a bit in a rate-distortion approach. Using the definition of norm and the overall error, our problem becomes:

$$\underset{\vec{b}}{\text{minimize}} \quad \sum_{n=1}^{N} (e_n(b_n) + \lambda b_n)$$

We have effectively separated our problem into a block-based problem. In this case, we now need to solve individual problems of the form:

$$\underset{b_n}{\text{minimize}} \quad e_n(b_n) + \lambda b_n$$

Now, define the return $r_n(b_n)$, given by the difference in error found when adding the b_n-th bit to block n, or:

$$r_n(b_n) = e_n(b_n - 1) - e_n(b_n) \tag{7.2}$$

We now discuss two important properties for $r_n(b_n)$. First, we have the non-negative return property, which simply states that adding a bit will never have negative return. Mathematically, this means $r_n(b_n) \geq 0$ for all b_n. This property is quite obvious in most scenarios, as, if done in a reasonable manner, adding a bit can never increase the error.

Second, we have the diminishing returns property. This states that adding bit b_n to a block will never have greater return than adding bit $b_n + 1$. Mathematically, this can be written as:

$$r_n(b_n) \geq r_n(b_n + 1) \tag{7.3}$$

This condition is not always true, but will be satisfied in the intra-prediction scenario we are considering as we will show later. Assuming the diminishing return condition holds true, we can then show that the solution to our optimization problem is given simply by adding bits as long as $r_n(b_n) \geq \lambda$. To see this, note that we can write:

$$e_n(b_n) + \lambda b_n = e_n(b_n) + \sum_{m=1}^{b_n} (e_n(m) - e_n(m)) + \lambda b_n \tag{7.4}$$

Using the definition of $r_n(m)$ and manipulating the sums, we find:

$$e_n(b_n) + \lambda b_n = e_n(0) + \sum_{m=1}^{b_n} (\lambda - r_n(m)) \tag{7.5}$$

Note that $e_n(0)$ does not depend on b_n, so our optimization problem can be written simply as:

$$\underset{b_n}{\text{minimize}} \quad \sum_{m=1}^{b_n} (\lambda - r_n(m)) \tag{7.6}$$

As long as we have diminishing returns, note that this minimization problem becomes quite trivial, as we can stop adding terms as soon as we find the first positive $(\lambda - r_n(m))$ term. Intuitively, this just means that we stop as soon as our return is more expensive than our cost λ. Since our return is always decreasing, we do not need to worry about possible high returns after we find the first non-profitable bit increase.

7.1.2 Multiple Set Algorithm

We now specialize our theoretical setup specifically taking into account the prediction inaccuracy model. Then we show that such a framework can lead to an algorithm with a significant reduction of bit-rate on top of what we had already observed.

We first return to the prediction inaccuracy model in the case where we are using a horizontal predictor. According to this model, intra-prediction residuals may be written as:

$$r(m,n) = m\theta \frac{\partial f(0,n)}{\partial n} + \epsilon(m,n) \tag{7.7}$$

where $\epsilon(m,n)$ is a signal-independent noise term. We will assume for the sake of this analysis that this noise is white. We observe that the only random variable in this scenario is the parameter θ, as the gradient is mostly known from previous boundaries, and (m,n) is deterministic for each pixel. In this case, we note that the variance of these same residuals $\sigma_r^2(m,n)$ is:

$$\sigma_r^2(m,n) = m^2 \sigma_\theta^2 \left(\frac{\partial f(0,n)}{\partial n} \right)^2 + \sigma_\epsilon^2 \tag{7.8}$$

Now we can consider the residual variance as a function of the number of bits used in side-information for each block. In this case, we note that σ_θ^2 is the main term that is changed when we add a bit to θ. Let us assume that the intra-prediction angle θ is effectively quantized via its representation, and adding a bit to θ implies we are doubling the number of quantization regions. Assuming θ is uniform in each of its quantizer regions will in turn imply that adding a bit will divide σ_θ^2 by 4. In fact, let b_0 be the smallest number of bits we can use and σ_θ^2 denote the angle variance for the prediction inaccuracy when we use b_0 bits. Our error $e(b)$ as a function of the number of bits b used will be:

$$e(b) = \left(\frac{1}{4} \right)^{b-b_0} m^2 \sigma_\theta^2 \left(\frac{\partial f(0,n)}{\partial n} \right)^2 + \sigma_\epsilon^2 \tag{7.9}$$

In this case, the return $r(b)$ will be:

$$r(b) = e(b-1) - e(b) = 3 \left(\frac{1}{4} \right)^{b-b_0} m^2 \sigma_\theta^2 \left(\frac{\partial f(0,n)}{\partial n} \right)^2 \tag{7.10}$$

It is simple to see that in this case the diminishing returns condition will be satisfied as the return $r(b)$ is exponential with b. Therefore we should add bits to our encoder as long as $r(b) \geq \lambda$. From section 7.1.1, we should choose b to be as large as possible but satisfying:

$$\lambda < 3 \left(\frac{1}{4}\right)^{b-b_0} m^2 \sigma_\theta^2 \left(\frac{\partial f(0,n)}{\partial n}\right)^2 \tag{7.11}$$

From this equation, we conclude that we should choose how many bits allocated to this block as a function of its maximum absolute gradient squared. The maximum absolute gradient squared should be thresholded based on thresholds that scale by 4 with each added bit, starting at the smallest set $b_0 = 3$.

Our algorithm is summarized in Figure 7-1. We fix a threshold β. If a block's maximum absolute gradient squared is smaller than β, we use the 7 intra-prediction modes. On the other hand, if it is between β and 4β, we add a bit to the non-MPMs and use 11 prediction modes. If it is between 4β and 16β, we add a bit to the non-MPMs and use 19 prediction modes. Finally, if it is greater than 16β, we add a bit to the non-MPMs and use the full set of 35 prediction modes, as is currently done in HEVC.

Intuitively, let us consider what the algorithm is accomplishing. It takes into account the magnitude of the edge from the information of the neighboring block. Depending on how large the edge is, it decides how fine we should approximate it by using a larger set of intra-prediction modes. This reduces the area where the prediction is inaccurate when we have larger edge magnitude, and will therefore reduce the residual energy in those cases, while having a smaller number of side-information bits in the cases where the edge has a smaller magnitude.

We also consider this algorithm from the perspective of the previously derived metric functions $\mu^A(\lambda)$. Let S_1, S_2, S_3, S_4 denote the four possible increasing sets of intra-prediction modes. In this case, we have four possible curves $\mu_k^S(\lambda)$. By having these, each of them has a different incline, and we can take into full effect how the difference in incline will lead to a different set being optimal as a function of λ, slowly reducing side-information bits when we increase λ to indicate the relative weight of side-information in the rate-distortion trade-off.

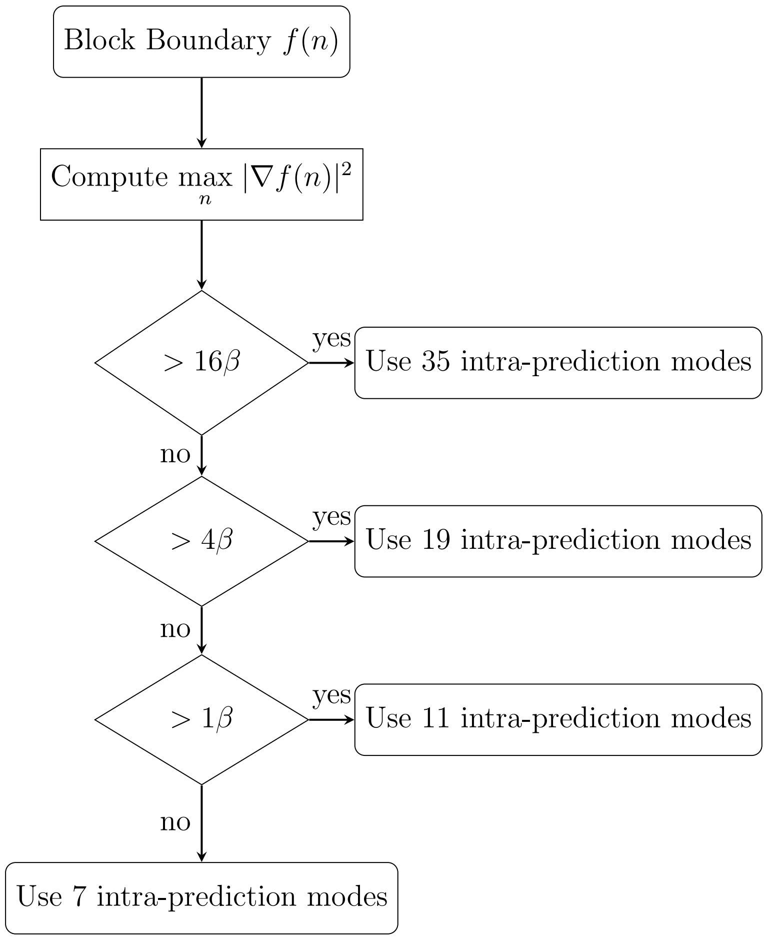

Block Boundary $f(n)$
Compute $\max_n |\nabla f(n)|^2$
$> 16\beta$
yes
Use 35 intra-prediction modes
no
$> 4\beta$
yes
Use 19 intra-prediction modes
no
$> 1\beta$
yes
Use 11 intra-prediction modes
no
Use 7 intra-prediction modes

Figure 7-1: Summary of multiple set algorithm to compute number of intra-prediction directions used in block as a function of the block boundary $f(n)$ and the threshold parameter β.

7.1.3 Results

To compare these results, we use the same experimental setup as defined in the previous chapter. Our first four images are 256×256 images from the kodak dataset, shown in Figure 6-1. We also compare our results with the same images but in 512×512 resolution. Furthermore, we consider standard 1280×720 images from derf dataset in Figure 6-2. As in the previous chapter, we take a single block-size to be used in the whole image. If the image cannot be exactly divided, we crop a few pixels to ensure that it fits.

To encode the intra-prediction modes, we use a code based on a choice of either the three most probable modes or a fixed length code, as is done in HEVC. These are then bypass coded. If the previous block used a different set of prediction directions, we still derive the optimal direction it should have used, so the most probable modes are consistently independent of the chosen set in each block.

The bit-rate reduction when using the new method with respect to a set of fixed intra prediction directions is computed for multiple images, and shown in Table 7.1. A comparison with the results presented in Chapter 6 is then shown in Table 7.2.

Image Resolution	Sequence	4×4	8×8	16×16	32×32
256×256	Baboon	3.41 %	1.13 %	0.49 %	0.19 %
	Barbara	3.22 %	2.36 %	1.74 %	1.59 %
	Lena	3.49 %	2.47 %	1.17 %	0.83 %
	Peppers	4.37 %	2.11 %	1.24 %	0.37 %
256×256	Average	3.62 %	2.02 %	1.16 %	0.75 %
512×512	Baboon	2.39 %	0.85 %	0.44 %	0.29 %
	Barbara	1.46 %	0.72 %	0.81 %	0.60 %
	Lena	3.81 %	1.43 %	2.99 %	0.63 %
	Peppers	4.88 %	2.53 %	1.93 %	1.15 %
512×512	Average	3.14 %	1.38 %	1.54 %	0.67 %
1280×720	Johnny	2.30 %	0.96 %	1.66 %	1.18 %
	Park	3.80 %	1.56 %	0.96 %	0.04 %
	Town	4.69 %	1.43 %	0.97 %	0.49 %
	Tree	5.68 %	1.67 %	0.96 %	0.20 %
1280×720	Average	4.12 %	1.41 %	1.14 %	0.48 %

Table 7.1: BD-rate improvement when utilizing HEVC-based encoder with selection based only on four sets of intra-prediction directions and geometrical thresholding with respect to original HEVC-based encoder.

We begin our analysis by reporting the similar conclusions between both algorithms. First, we observe that both algorithms present overwhelmingly positive results for all resolutions and block-sizes. With the exception of images with extremely high frequencies, such as Baboon and Raft, our method has reduced the amount of intra-prediction side-information for the same SATD level. In these images, increasing the number of prediction directions does not significantly affect the SATD, as the blocks consist of mostly high frequencies, and any prediction ends up having similarly poor results.

Second, we note that the improvement is much higher in both methods as we

Image Resolution	Sequence	4×4	8×8	16×16	32×32
256×256	Baboon	0.26 %	-0.04%	-0.01 %	-0.05 %
	Barbara	0.40 %	0.18 %	0.35 %	0.19 %
	Lena	0.59 %	0.13 %	0.14 %	-0.33 %
	Peppers	0.66 %	0.58 %	-0.08 %	-0.33 %
256×256	Average	0.48 %	0.21 %	0.10 %	-0.13 %
512×512	Baboon	0.13 %	0.04 %	-0.01 %	-0.06 %
	Barbara	-0.40 %	-0.32 %	-0.12 %	-0.77 %
	Lena	0.35 %	-0.08 %	0.44 %	-0.26 %
	Peppers	0.55 %	0.06 %	0.04 %	-0.12 %
512×512	Average	0.16 %	-0.08 %	0.09 %	-0.30 %
1280×720	Johnny	0.21 %	0.43 %	0.33 %	1.18 %
	Park	0.04 %	0.60 %	0.39 %	-0.39 %
	Town	-0.08 %	-0.24 %	-0.08 %	0.05 %
	Tree	-1.49 %	-0.06 %	-1.09 %	-0.76 %
1280×720	Average	-0.33 %	0.18 %	-0.11 %	0.02 %

Table 7.2: BD-rate difference when utilizing HEVC-based encoder with selection based only on four sets of intra-prediction directions and geometrical thresholding instead of the two-set case.

reduce block-sizes. There are a few reasons why that would be the case. Using smaller block sizes means our boundaries are more effective predictors. They will also contain better edge information for the current block. Therefore, the probability our block has an edge and should be encoded with more directions will be more accurate. Note also that the intra-prediction side-information bit-rate will be higher as we reduce the block-size, as we have to add more bits over the whole image. Thus, reducing it will have a bigger impact in compression results.

Third, we note that as the resolution increases, the improvement may also increase. This comes from the notion that our predictors will be more accurate, as happened when we previously discussed reducing the block-sizes. This will be especially true in images where the low frequencies are more predominant. Unfortunately, we do not have enough evidence yet to make broader claims about this effect.

We can now observe the difference between using only two sets and using multiple small sets. Theoretically, using only two sets will be worse than using multiple small sets as long as our thresholds are appropriately chosen. These results do manifest in this scenario. As may be seen in both tables, using multiple sets of intra-prediction directions do in fact significantly reduce the intra-prediction side-information BD-rate.

A natural question is why this improvement is present in some images more than in others. Based on the images used in the previous examples, we believe the main reason is related to the presence of edges in the image with medium magnitude. These edges may be better captured by having multiple thresholds that scale, so we can take into account their contribution to the oveall error by reducing it proportionally. This could not be finely captured by the previous algorithm, since only two sets were used.

Another natural question is a comparison of this improvement in terms of different resolutions. From our experiments, our multi-set method tends to result in more improvement for smaller resolutions of 256×256 than our 720p images. This can be attributed to the fact that the higher quality images have larger smooth areas. These areas should be encoded more effectively by using the smaller set, reducing the importance of using multiple sets with geometric thresholds.

We have also evaluated different geometric factors to separate sets, but none has given us better results than what we currently have. The main suggestions for future approaches to the question of which set cardinality to use would be to evaluate different selection criteria instead of the maximum absolute gradient squared, possibly based on different image models.

7.2 Adaptive Method

We now reconsider the scenario presented in the previous chapter. When we designed our set of prediction directions, it was chosen as a reasonable set, but in no way was it optimal. It was simply a uniform split of the set of possible prediction directions. A natural question here comes up: can we choose a better set of directions to use in our small set? Specifically, can we do this by taking into account the modes neighbors used or should have used?

7.2.1 Proposed Method

From the prediction inaccuracy model for intra-prediction and equation (7.7), we can write the expected prediction error e as a function of the mode used $\widehat{\alpha}$ and the true angle α^* as:

$$e(\alpha, \widehat{\alpha}) = c(\widehat{\alpha})(\alpha^* - \widehat{\alpha})^2 \tag{7.12}$$

where $c(\widehat{\alpha})$ is some constant dependent on $\widehat{\alpha}$ as well as the known gradient boundary. This is the same equation as the one found in equation (4.11), and we can propose an analysis similar to the one proposed there. From the prediction inaccuracy model and the discrete gradients, we can assume $c(\widehat{\alpha})$ to be relatively smooth. This indicates that, for each angle $\widehat{\alpha}$, we have a different width parabola, depending on our gradient boundary. The goal of this problem becomes to place the parabola such that the area covered, weighted by the probability that each α is the true α^* is non-zero.

This problem is non-trivial to solve, as the coefficients are not well defined, but we here propose an interesting approach. Assume that the coefficient $c(\widehat{\alpha})$ is constant, or $c(\alpha) = c$ for all α. Then our problem is equivalent to finding the optimal quantizer levels for minimum mean-squared error. In this scenario, the optimal quantizer depends on the distribution $p_{\alpha^*}(\alpha^*)$.

An interesting model for $p_{\alpha^*}(\alpha^*)$ is to assume that it is given by fast decaying exponentials or Gaussians centered around the modes used in the left and top neighboring blocks. This distribution will lead us to wish to choose modes that are close to these MPMs for our quantizers. Assuming the exponentials are relatively fast decaying, we would simply like to take the MPM modes and modes around it.

We propose a method by simply approximating the most probable modes taking an increment factor into account. This method is defined based on the intra-prediction modes of the left and top neighbors, denoted as (L, T). We start with what we call the base set B, which consists of DC and planar modes, as our observations indicate these modes should always be present. We form an additional set of $N - 2$ modes, according to the following rules with respect to some increment value i.

(a) Add L to the additional set if L is not in the base set.

(b) Add T to the additional set if T is not in the base set.

(c) If the additional set is empty, both neighbors use planar or DC, so no directional features imply we should just use our previous equally spaced set.

(d) Otherwise, for each element x in the additional set, add $x + i$ and $x - i$ to the additional set. Repeat this step until it has $N - 2$ elements.

(e) The set of modes allowed should be the union of the base set B and the additional set.

Intuitively, what this algorithm does is to expand the directional modes. For each directional mode used in a neighbor, we add the modes adjacent to it. If there are still modes available to add, we can add the adjacent modes to the adjacent modes, and so forth. If no neighbors used directional modes, we then choose to use the previously designed equally spaced set. Note that we don't necessarily add the directly adjacent modes. It may be more advantageous to give some spacing, even if small, between directional modes to be added.

We now consider an example of how this method would work. Assume the left neighbor uses DC mode ($L = 1$) and the top neighbor uses directional mode $T = 25$. Also, let us use increment $i = 1$ in this example. Then the base set for $N = 7$ would consist of DC and planar, and the additional set would be $\{25, 26, 24, 27, 23\}$. In this case, our set of modes would be $\{0, 1, 23, 24, 25, 26, 27\}$. If $i = 2$ instead, then our additional set would be $\{25, 27, 23, 29, 21\}$, and the set of modes would be $\{0, 1, 21, 23, 25, 27, 29\}$. In other words, parameter i can incorporate the edge directional variance, and could be transmitted as a choice for each image as well.

This method ensures that the neighboring modes are always in the set, and uses the increment i to characterize the fast decay of the exponentials. Depending on the image resolution or other statistical properties, the correct choice of the increment term i will be different.

As in our previous algorithm, we also use the maximum absolute gradient squared as a way to decide whether or not to use the adaptive set of 7 modes instead of the full set of 35 modes. This is done as we still wish to use a smaller number of modes only on blocks without large edges, where higher precision will be necessary.

7.2.2 Results

We apply this method to the images in Figure 6-1 and Figure 6-2, as we did for the previous methods. We test blocksizes of 4×4, 8×8, 16×16 and 32×32, where the same blocksize is used over the whole image.

How did this method perform when compared to our previous method? We used an increment of $i = 2$ fixed. Table 7.3 shows the result with the new adaptive method, while Table 7.4 shows the relative improvement to the original method. As can be seen, the adaptive algorithm does lead to some improvement over the original method, especially for 256×256 images. The decay for larger sized images may be related to the suboptimality of the value of i. Optimizing this may lead to better results.

Another note is that the images that this method had the most improvement, such as Barbara and Johnny, were the opposite from the ones in which we had the most improvement in the multiple set method. We believe this can be attributed to the notion that the adaptive method better adapts to longer edges which are approximately linear, as it can better encode these. It does not perform as well in images where such structures do not appear as much, such as Lena image.

7.3 Comparison

In this section, we compare the three methods presented for intra-prediction side-information reduction. The first method, which we here refer as the original method, uses the maximum absolute gradient boundary squared as a selector between a large set of 35 intra-prediction modes, and a fixed set of 7 modes. This method was described in Chapter 5, and its results presented in Chapter 6.

The second method, which we here refer as the multiple set method, uses the

Image Resolution	Sequence	4×4	8×8	16×16	32×32
256×256	Baboon	4.52 %	3.19 %	1.83 %	1.00 %
	Barbara	4.18 %	3.59 %	1.95 %	1.10 %
	Lena	3.54 %	1.39 %	0.62 %	0.88 %
	Peppers	2.85 %	1.70 %	1.61 %	1.12 %
256×256	Average	3.77 %	2.47 %	1.50 %	1.03 %
512×512	Baboon	2.91 %	1.06 %	0.80 %	0.59 %
	Barbara	3.73 %	2.42 %	2.38 %	1.09 %
	Lena	2.81 %	1.30 %	0.46 %	0.09 %
	Peppers	3.47 %	2.36 %	1.21 %	0.68 %
512×512	Average	3.23 %	1.79 %	1.21 %	0.61 %
1280×720	Johnny	6.15 %	2.87 %	0.80 %	0.61 %
	Park	3.36 %	0.68 %	0.62 %	0.46 %
	Town	4.28 %	0.68 %	0.32 %	0.00 %
	Tree	4.04 %	1.37 %	0.43 %	0.34 %
1280×720	Average	4.46 %	1.40 %	0.54 %	0.35 %

Table 7.3: BD-rate improvement when utilizing HEVC-based encoder with selection based only on two sets of intra-prediction directions but one of our sets is generated by the adaptive method with $i = 2$. This leads to a significant improvement in performance, especially for 256×256 images.

maximum absolute gradient boundary squared as a selector between four possible sets. These sets are uniform partitions of the set of directions, as well as DC mode and planar mode. This method was described and evaluated in Section 7.1.

The third method, which we here refer as the adaptive method, uses the maximum absolute gradient boundary squared as a selector between our full set of 35 directions, and a set of 7 directions chosen adaptively based on the block's neighbors. This method was defined and evaluated in Section 7.2.

Theoretically, when we compare these methods to the two questions presented in Chapter 4, the original method chooses the value of the size of the set N between two possibilities. The multiple set method chooses the value of N between four possibilities. Both of these methods assume a uniform distribution is the best distribution for the set of modes. The adaptive method tackles the question of which directions should compose the set $A(n)$ using the criterion from the original method to decide its cardinality. The theoretical comparison between these methods is summarized in

Image Resolution	Sequence	4×4	8×8	16×16	32×32
256×256	Baboon	1.38 %	2.02 %	1.32 %	0.77 %
	Barbara	1.37 %	1.41 %	0.55 %	-0.30 %
	Lena	0.64 %	-0.95 %	-0.41 %	-0.03 %
	Peppers	-0.85 %	0.17 %	0.29 %	0.42 %
256×256	Average	0.64 %	0.66 %	0.44 %	0.22 %
512×512	Baboon	0.65 %	0.26 %	0.35 %	0.23 %
	Barbara	1.87 %	1.39 %	1.45 %	0.41 %
	Lena	-0.64 %	-0.21 %	-2.10 %	-0.79 %
	Peppers	-0.86 %	-0.12 %	-0.68 %	-0.58 %
512×512	Average	0.26 %	0.33 %	-0.25 %	-0.18 %
1280×720	Johnny	4.06 %	2.27 %	-0.62 %	0.24 %
	Park	-0.40 %	-0.28 %	0.05 %	0.11 %
	Town	-0.48 %	-0.99 %	-0.73 %	-0.44 %
	Tree	-3.13 %	-0.33 %	-1.62 %	-0.62 %
1280×720	Average	0.01 %	0.17 %	-0.73 %	-0.18 %

Table 7.4: BD-rate improvement when utilizing HEVC-based encoder with selection based on two adaptive sets in relation to original two set method presented in Chapter 6. This leads to a significant improvement in performance in the case of 256×256 images.

	Original Method	Multiple Set Method	Adaptive Method
Alternates between sets of prediction directions	X	X	X
Has more than two possible sets of directions		X	
Sets of prediction directions designed uniformly	X	X	
Adapts set of directions based on local content			X

Table 7.5: Summary of theoretical comparison between the three different algorithms presented in the context of intra-prediction side-information reduction.

Table 7.5.

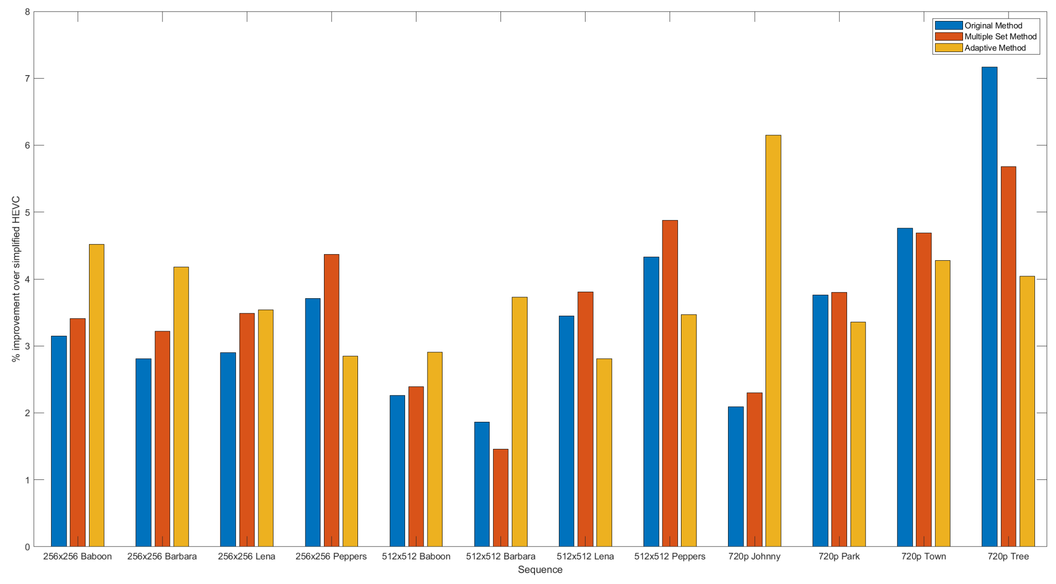
Figure 7-2 compares the results obtained from the three different methods for our test images in Figure 6-1 and Figure 6-2 for 4×4 and 8×8 blocks. Figure 7-3 does the same comparison for 16×16 and 32×32 blocks.

There are a few things to note from this comparison. First of all, all methods provide more improvement for smaller blocks. As block sizes grow larger, the effect of side-information on the bit-rate is smaller. Therefore, reducing it will not lead to as much improvement.

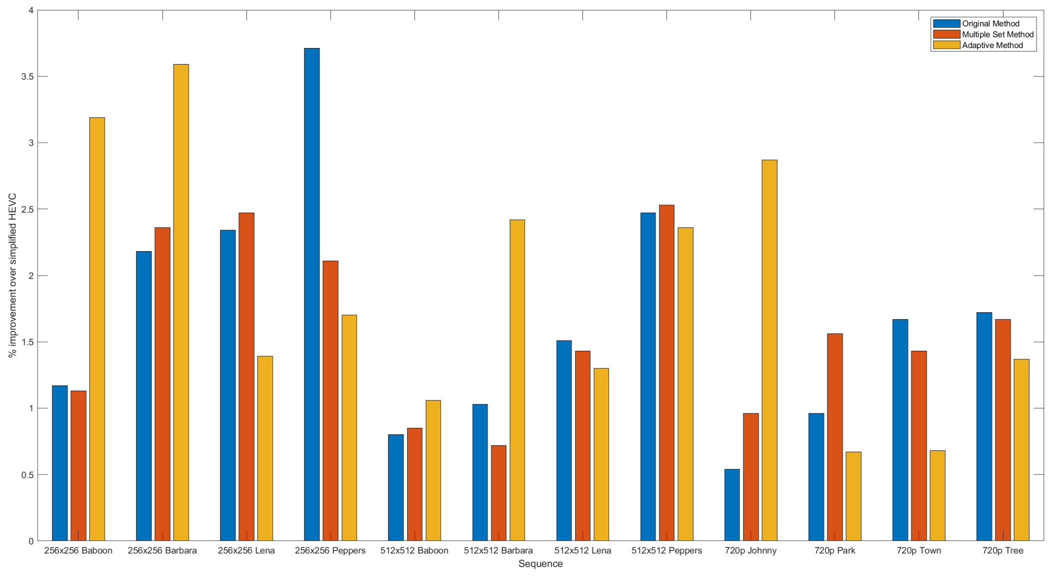
Second, the adaptive method performs very well for 256×256 images, but does not have similar results for larger image resolutions. We believe the value of i could be appropriately adapted for higher resolutions, but were not able to find a consistent way to adapt this value.

The multiple set method tends to perform better than the original method. Over many sequences, we can see that it has a small edge. We believe this difference is more significant as most of the gain with respect to HEVC will come from the blocks that should be using the smaller set, as in these blocks there is more room for bit-rate savings.

To summarize, there is still significant room for research in side-information reduction for intra-prediction. The initial method presented in Chapters 5 and 6 showed that you can in fact find an improvement to a simplified version of HEVC by adapting intra-prediction side-information to local content available at both the encoder and the decoder. The approach presented in Section 7.1 leads us to conclude that there is room to choose intermediary cardinalities to the set of prediction directions, and that we do not need to be restricted to only the values of 7 or 35 intra-prediction modes. The approach presented in Section 7.2 shows that adaptive approaches also have high potential to improve encoder performance, but are significantly harder to design. Since future standards should increase the number of prediction directions even further, we believe developing methods that can adaptively reduce the amount of side-information such as the ones presented here will be very useful. In the next chapters, we will explore how side-information reduction can also be applied to motion compensation side-information reduction.



(a) 4×4 blocks

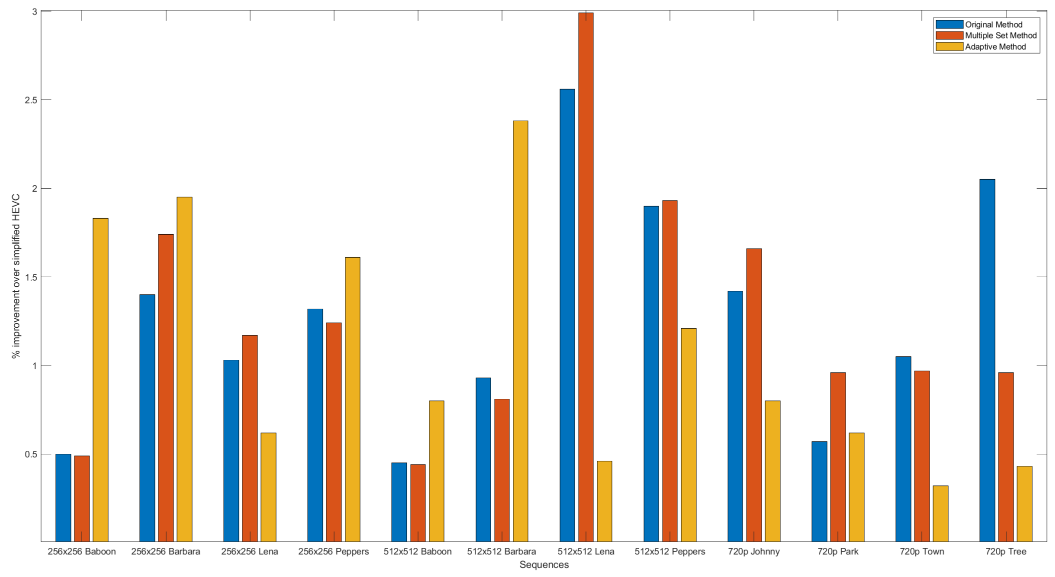


(b) 8×8 blocks

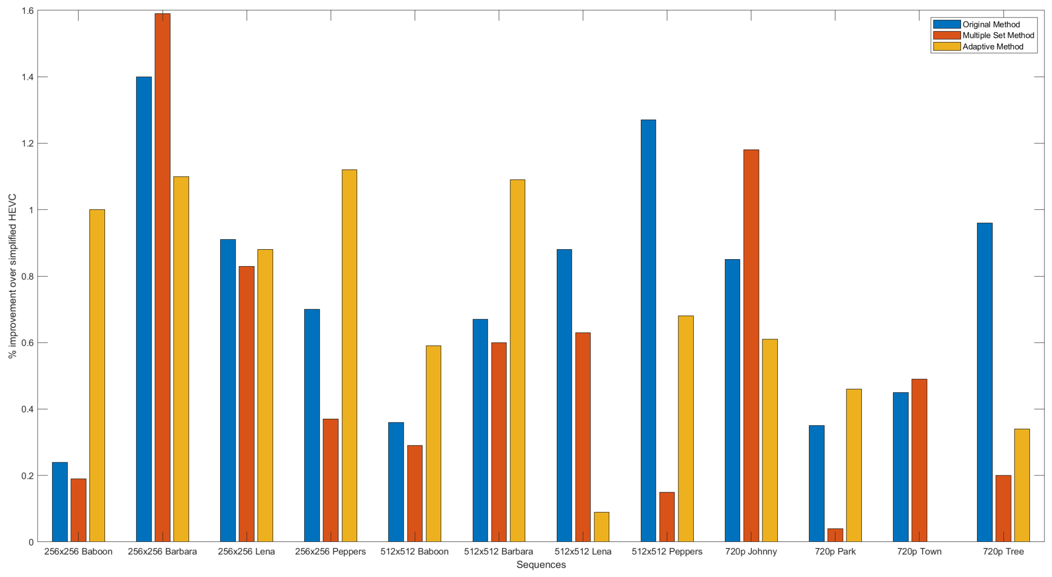
Figure 7-2: Improvement in BD metric when using three methods designed in Chapters 5 and 7 instead of the simplified HEVC-based encoder for 4×4 and 8×8 blocks. As can be seen, the multiple set method tends to perform better than the original method. The adaptive method tends to perform better than the others for lower resolution images, but worse for higher resolution images.

Summary

In this Chapter, we presented two additional methods to reduce intra-prediction side-information. Our first method proposes to use a larger number of sets than just two.



(a) 16 × 16 blocks



(b) 32 × 32 blocks

Figure 7-3: Improvement in BD metric when using three methods designed in Chapters 5 and 7 instead of the simplified HEVC-based encoder for 16×16 and 32×32 blocks. Similar to the previous cases, the multiple set method tends to perform better than the original method. The adaptive method tends to perform better than the others for lower resolution images, but worse for higher resolution images.

Instead of choosing between a set of 7 and a set of 35 prediction modes, we propose to choose between sets of 7, 11, 19 or 35 uniformly spaced prediction modes. This is still done by thresholding the maximum absolute gradient squared, as theoretically derived from prediction inaccuracy modeling, where the thresholds scale by a factor of 4 per set. This algorithm was shown to lead to some improvement over the results in the previous Chapter.

Our second method attempts to design the optimal set based on the modes used in the top and left neighbors. The idea is to construct a set to approximate the current block's directions by adding the directions used in the neighbors and their neighbors according to an increment factor. This method was our main attempt at using a variable set $A(n)$ given a fixed cardinality of 7 modes for the blocks where we opt to use a smaller number of sets. This lead to an improvement in the case of 256×256 images, but not a significant improvement in higher resoltuions.

Chapter 8

Motion Compensation Side-Information Reduction

In this section, we design an algorithm to reduce motion compensation prediction side-information. We begin by reviewing motion compensation prediction inaccuracy modeling presented in Section 3.2, and how it relates to side-information precision. In particular, we estabilish a trade-off between side-information and the residual prediction energy in a very similar manner to how we did before in Section 7.1.1.

We will then show how we can use this model to derive an algorithm, similar to the one in Chapter 5. This algorithm is based on thresholding the maximum absolute gradient in the predictor block. Finally, we will conclude by explaining how this is another application of our setup presented in Chapter 4. This will highlight the versatility of our setup as a mathematical framework when designing a side-information based encoder for applications other than intra-prediction.

8.1 Theoretical Analysis

In Section 3.2, it was found that the residual signal $r(m, n)$ is related to the original signal $f(m, n)$ via:

$$r(m,n) = f(m,n) - \widehat{f}(m,n) \approx \widehat{f}(m_a,n_a) - \widehat{f}(m,n) \approx \Delta m \frac{\partial \widehat{f}}{\partial m} + \Delta n \frac{\partial \widehat{f}}{\partial n} \tag{8.1}$$

where Δm and Δn denote how much the transmitted motion vector is off from the true motion vector. Since the gradients of the previous frame $\widehat{f}$ are known, we then have the residual variance equal to:

$$\sigma_r^2(m,n) \approx E[(\Delta m)^2] \left(\frac{\partial \widehat{f}}{\partial m}\right)^2 + E[(\Delta n)^2] \left(\frac{\partial \widehat{f}}{\partial n}\right)^2 \tag{8.2}$$

This equation highlights two notions: First, if the motion vector is chosen incorrectly or imprecisely, or mathematically if we have large Δm and Δn, then we will have larger variance in our prediction residuals. Second, if we have large gradient in the previous frame, we will have larger prediction residuals. This is related to the observation that edges and textures tend to have more residual inaccuracy.

Now, a natural question is how this relates to the side-information used to encode the motion vector. Assume we encode the vector without any half or quarter pixel interpolation. Effectively what this means is that we only allow the motion vector to have integer coordinates. Unfortunately, true motion is not required to be an integer. In fact, it can be respresented as a continuous random variable. The inaccuracy between true motion and its integer representation will correspond to inaccuracy values of Δm and Δn. Assume we add a bit to the motion vector by implementing half-precision, this will concentrate the values of Δm and Δn closer to 0.

Assuming Δm is fine enough such that its distribution is approximately uniform within the range given by the motion vector step-size, we note that adding a bit will simply divide its variance by 4, as it will divide its range by half. In other words, by adding a bit and forming a new prediction error vector $\Delta m'$, we have:

$$E[(\Delta m')^2] = \frac{1}{4} E[(\Delta m)^2] \tag{8.3}$$

This is the same behavior for the other component of the motion vector, by a simple symmetry argument. Assume we have a prediction vector encoded uniformly using b_m bits to encode the fractional part of the first motion vector component instead, here denoted by Δm_{b_m}, then we will have:

$$E[(\Delta m_{b_m})^2] = \frac{1}{4^{b_m}} E[(\Delta m)^2] \tag{8.4}$$

Similarly, let b_n denote the number of bits used to encode the fractional part of the second motion vector component instead, here denoted by Δn_{b_n}, we have:

$$E[(\Delta n_{b_n})^2] = \frac{1}{4^{b_n}} E[(\Delta n)^2] \tag{8.5}$$

We replace the variances in Equation 8.2, to find the variance if we use a fractional motion vector representation with b_m fractional bits in the first component and b_n fractional bits in the second component. The resulting residual variance will be equal to:

$$\sigma_r^2(m,n) \approx \frac{1}{4^{b_m}} E[(\Delta m)^2] \left(\frac{\partial \widehat{f}}{\partial m} \right)^2 + \frac{1}{4^{b_n}} E[(\Delta n)^2] \left(\frac{\partial \widehat{f}}{\partial n} \right)^2 \tag{8.6}$$

Now we consider from a rate-distortion perspective, where we wish to minimize the residual variance plus a trade-off variable λ times the number of side-information bits used. The result we have is:

$$\sigma_r^2(m,n) + \lambda(b_m + b_n) = \frac{1}{4^{b_m}} E[(\Delta m)^2] \left(\frac{\partial \widehat{f}}{\partial m} \right)^2 + \frac{1}{4^{b_n}} E[(\Delta n)^2] \left(\frac{\partial \widehat{f}}{\partial n} \right)^2 + \lambda(b_m + b_n) \tag{8.7}$$

We can separate this equation in two terms, referring to the horizontal and vertical components of the motion vector:

$$\sigma_r^2(m,n) + \lambda(b_m + b_n) = \left(\frac{1}{4^{b_m}} E[(\Delta m)^2] \left(\frac{\partial \widehat{f}}{\partial m} \right)^2 + \lambda b_m \right) + \left(\frac{1}{4^{b_n}} E[(\Delta n)^2] \left(\frac{\partial \widehat{f}}{\partial n} \right)^2 + \lambda b_n \right) \tag{8.8}$$

From this equation, we first note that each direction can choose to add a bit at a time, independently in each direction. To add a bit to m, we need to have:

$$\frac{1}{4^{b_m}} E[(\Delta m)^2] \left(\frac{\partial \widehat{f}}{\partial m} \right)^2 + \lambda b_m > \frac{1}{4^{b_m+1}} E[(\Delta m)^2] \left(\frac{\partial \widehat{f}}{\partial m} \right)^2 + \lambda (b_m + 1) \tag{8.9}$$

We can isolate the terms to find:

$$\frac{3}{4^{b_m+1}} E[(\Delta m)^2] \left(\frac{\partial \widehat{f}}{\partial m} \right)^2 > \lambda \tag{8.10}$$

Define the variable β_m to be independent of b_m and given by:

$$\beta_m^2 = \frac{4\lambda}{3E[(\Delta m)^2]} \tag{8.11}$$

Then we should add a bit as long as:

$$\left(\frac{\partial \widehat{f}}{\partial m} \right)^2 > 4^{b_m} \beta_m^2 \tag{8.12}$$

Therefore, we should add bits to the m index as long as the gradient squared is greater than $4^{b_m} \beta_m^2$. Another way to see this equation is to take b_m to be the largest value to satisfy:

$$b_m < \log_2 \left(\frac{\left| \frac{\partial \widehat{f}}{\partial m} \right|}{\beta_m} \right) \tag{8.13}$$

The same argument applies to the n index of the motion vector, by symmetry. In this case, we would have:

$$\left(\frac{\partial \widehat{f}}{\partial n} \right)^2 > 4^{b_n} \beta_n^2 \tag{8.14}$$

where β_n is defined symmetrically as:

$$\beta_n^2 = \frac{4\lambda}{3E[(\Delta n)^2]} \tag{8.15}$$

8.2 Proposed Algorithm

Based on Section 8.1, we will propose an algorithm, similar to that presented in Chapter 5. As we can see in equations 8.12 and 8.14, we should add bits for additional precision to a motion vector direction as long as the gradient in that direction is large. Therefore, a natural approach would be one similar to that in Section 7.1, but that would not hold.

There is one main issue with this setup: the motion vectors are not encoded uniformly. In HEVC, to encode a motion vector, first you form a prediction $v_p(m,n)$. This prediction is transmitted out of a few options on a list, based on neighboring blocks. Then you compute the inaccuracy of this prediction $\Delta v(m,n)$ given by:

$$\Delta v(m,n) = v_p(m,n) - v(m,n) \tag{8.16}$$

This inaccuracy is then encoded, using a non-uniform codebook. Each component is encoded independently. First, a flag indicating whether or not the component of this motion vector is 0 is signaled. Then a flag indicating the sign of this motion vector component is signaled, as well as a flag indicating whether this component is greater than 1. If the motion vector component is absolutely larger than 1, then the absolute component minus two is encoded using an Exponential Golomb code of first order. These additional bits are also encoded using a context model based on the neighboring blocks. In areas where the Δv tends to be large, codeword length takes this into account by assuming a wider tail, and vice versa.

This code is highly optimized for motion compensation residuals encoded in HEVC. We will compare our method and base the analysis in this section and the next chapter on a simpler code. The main reason we opt to do this is that we simply wish to demonstrate the potential in using a side-informaton reduction method for motion compensation. In our case, we will assume a fixed motion search window size of $2W+1$ in each direction, where W is fixed for Δv. The motion vector components will in this case be encoded independently. First a flag is sent indicating whether it is 0, as this is very often the case. Then the other $2W$ values are encoded uniformly.

Given this code, we propose a simple procedure to decide whether to use quarter-pixel interpolation or no interpolation for Δv. For each motion vector component, we interpolate it as long as the absolute gradient of the block given by the integer part of Δv is larger than some threshold β. Otherwise we simply use the integer motion vector. The rest of the encoding is performed in the same way, using only fewer bits.

Note that this algorithm does not require any new information in the decoder. In fact, the decoder will know the previous decoded frames, and can decide whether or not the next bit belongs to which motion vector at a time, without any additional side-information overhead. This will not generate any issues as such decision is synchronized in the encoder and the decoder.

A natural question is how this algorithm relates to what was shown in Chapter 4. The two main questions in Chapter 4 are how we find the size N of the set $A(n)$, and, once we know N, how we choose $A(n)$ for a given block. This method basically increases the size of $A(n)$ based on the processed block's variance. This variance is estimated by observing the gradient in the previously decoded block given by the predictor. In the case of motion compensation residuals, each direction is done separately as a large horizontal gradient does not imply a large vertical gradient and vice versa. The choice of $A(n)$ is still based on a uniformity assumption other than the very common value of 0 given by when the motion vector is correctly predicted. This assumption is not true, and we will discuss more about it at the end of the next chapter.

Summary

In this Chapter, we presented an algorithm to reduce the motion compensation side-information. This algorithm is based on the prediction inaccuracy modeling for motion compensation residuals. It utilizes thresholds based on the maximum absolute gradient to use a larger interpolation factor to either the horizontal or vertical direction. This algorithm is designed based on the simplifying assumption that motion vectors are transmitted using a uniform encoding on the last bits.

Chapter 9

Motion Compensation Side-Information Reduction Results

In this section, we will first present the experimental setup and results when applying the algorithm we designed in the previous chapter. Through this initial approach, we will conclude that there is significant room for an approach to be developed that adapts motion vector precision as a function of local content in previous frames.

9.1 Experimental Setup

In the experiments we performed for motion compensation side-information reduction, we implemented our method in an adapted version of H.264. We opt to use a version of this standard, as it has a simpler way to estimate the motion vector predictor: In H.264, each of the motion vector predictor's components is given by the median of its previously decoded neighbors. In HEVC, a list of predictors is formed, and an index to one of these predictors is transmitted. For our initial, simplified analysis, we did not wish to consider the effect this encoding would have on the theoretical setup presented, as we were not as focused on the number of bits used to encode each particular possible motion vector difference. Note that using a predictor from this list should change the result in the sense that very often this predictor will serve as an approximation to the correct motion vector. This will lead to smaller codewords on average, and future practical algorithms that attempt to solve the motion compensation side-information reduction problem will have to be designed with a codebook that takes this into account.

For motion compensation residuals, we still assume fixed block-size used to encode a video frame, chosen from the possible sizes of 4×4, 8×8, 16×16 or 32×32. A single previous frame is assumed to have been encoded perfectly, so the accumulated error from lossy encoding of an I-frame or P-frame will not affect our analysis.

The motion vector is chosen via brute-force search, by optimizing a rate-distortion metric of the SATD plus λ times the side-information bit-rate. Overall, we will compare our method to the original by analyzing the different values of this metric produced over the whole image. This metric is an approximation to encoder performance, but it is simpler in computational and analysis terms, and will allow us not to focus as much in the interaction between different components of the encoder.

In the experiments we performed, we tested sequences in the derf dataset, with resolutions of 352×288 and 1920×1080 at a rate of 50 frames per second, also known as common intermediate format (CIF) and 1080p respectively. We process only the Y channel, and apply our method to only this channel for the second frame of the video sequences, assuming perfect knowledge of the first frame. The CIF sequences used are shown in Figure 9-1, and the 1080p sequences used are shown in Figure 9-2.

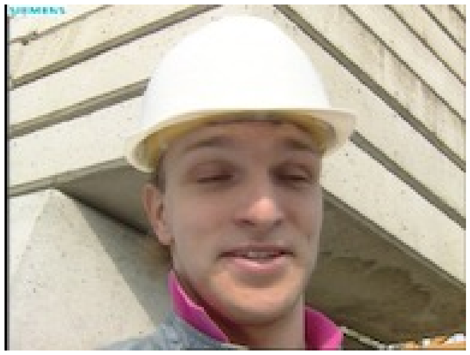
9.2 Results

We ran the algorithm to encode the second frame of each of the sequences in the two previous video sequences. The results are shown in Table 9.1. In this case, we first observed that our algorithm has a significant improvement over the naive strategy presented in the previous chapter. This improvement is especially significant for 4×4 blocks, as there are more blocks with small edges where capturing in detail isn't as important, and where the large number of blocks means there is greater need to

(a) Akiyo

(b) Coastguard

(c) Flower



(d) Foreman

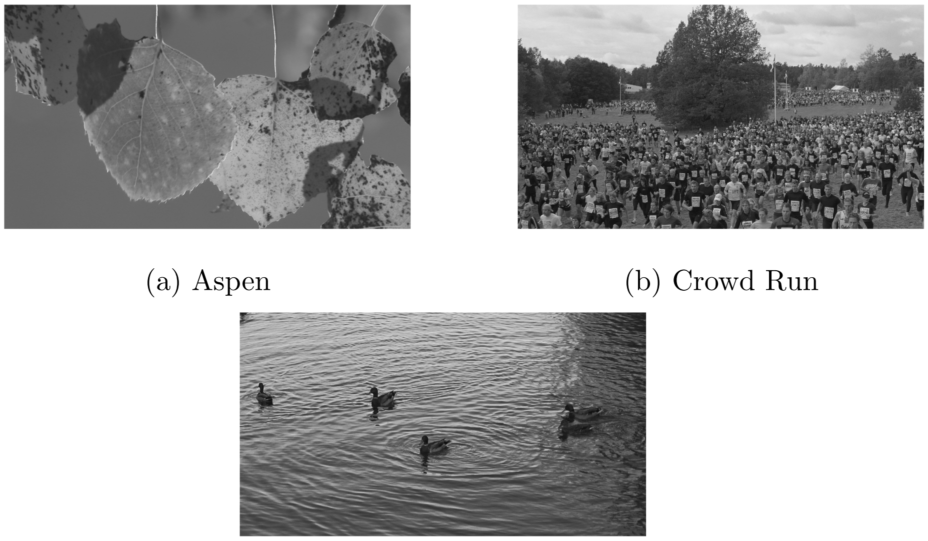
(e) Paris

(f) Silent

Figure 9-1: CIF test sequences, chosen to represent many possible features the algorithm could adapt to.

reduce side information.

A natural question when considering such a method is where we expect our method to provide the most improvement. This argument is actually very similar to the one presented in the case of intra-prediction residuals. When we considered motion compensation in the previous Chapter, we saw that most of the energy in residuals comes from edges. In fact, the error can be given by the magnitude of the edge times



(a) Aspen

(b) Crowd Run

(c) Ducks

Figure 9-2: 1080p test sequences, chosen to represent many possible features the algorithm could adapt to.

the inaccuracy area. This was the main observation we used when designing our method.

Based on this observation, we conclude most of our improvement will happen in images where there are many small magnitude edges. For these edges, we do not require as tight a bound in the inaccuracy area to reduce the overall error. When observing these sequences, we notice that we have larger improvement in images with the presence of small magnitude edges, such as Foreman in the CIF set, and Ducks in the 1080p set. We believe these are the areas a method such as the one proposed here has the greatest potential to improve.

In terms of QP values, we also noted the improvement was a lot higher for high values of QP. To observe this, Figure 9-3 shows the improvement obtained for the metric as a function of the value of the QP parameter for the sample sequence 'Silent' from Figure 9-1. The same behavior is also present in all other sequences. For high values of the QP parameter, the high value of λ in the associated rate-distortion trade-off increases the importance of reducing side-information bits.

In terms of resolution, we note that as we increase the resolution, there is a smaller gain from our method. It may at first seem counter-intuitive, but there is

Image Resolution	Sequence	4×4	8×8	16×16	32×32
352×288	Akiyo	9.35 %	4.85 %	7.76 %	0.60 %
	Coastguard	9.39 %	9.98 %	2.23 %	0.19 %
	Flower	9.40 %	6.21 %	1.82 %	0.98 %
	Foreman	19.06 %	11.99 %	3.76 %	0.10 %
	Paris	14.83 %	6.71 %	2.96 %	0.38 %
	Silent	11.74 %	10.06 %	5.88 %	0.98 %
352×288	Average	12.30 %	8.30 %	4.07 %	0.54 %
1920×1080	Aspen	6.91 %	5.74 %	1.80 %	1.26 %
	Crowd	4.42 %	1.48 %	0.42 %	0.56 %
	Ducks	18.38 %	9.30 %	2.92 %	0.77 %
1920×1080	Average	9.90 %	5.51 %	1.71 %	0.86 %

Table 9.1: Average improvement in sum of absolute differences plus lambda times bit-rate metric when comparing our method with the code that uses 1 bit for a 0 motion vector component, and uniform encoding for the other values.

an explanation. As we increase the resolution, smaller movement from the motion vector predictor may be captured. Note the probability of smaller movement from the predictor is larger than that of larger movement. This probability is much more significant than the chance of having larger motion. In HEVC, this is captured by using a flag to indicate if the motion vector has absolute magnitude equal to the interpolation factor. Unfortunately, this is not something our method accounts for as we assume uniform codewords over all possible non-zero motion vector differences.

In terms of computation, we note that there are some savings in terms of interpolation costs and reduction of search area by using this algorithm. In that case, we conclude there is some computational advantage to use a method to reduce motion compensation side-information. This does depend on how some parameters are computed, and the overall method can be more or less effective in this same sense based on implementation.

From a practical perspective, a main issue with this method is that it cannot be combined with the current standard encoder directly, as explained in the previous chapter. The main issue in this case will be that the codewords are designed in a less uniform manner, and most of the gain obtained from this method will come from the same source as the entropy encoding. Furthermore, the entropy encoding

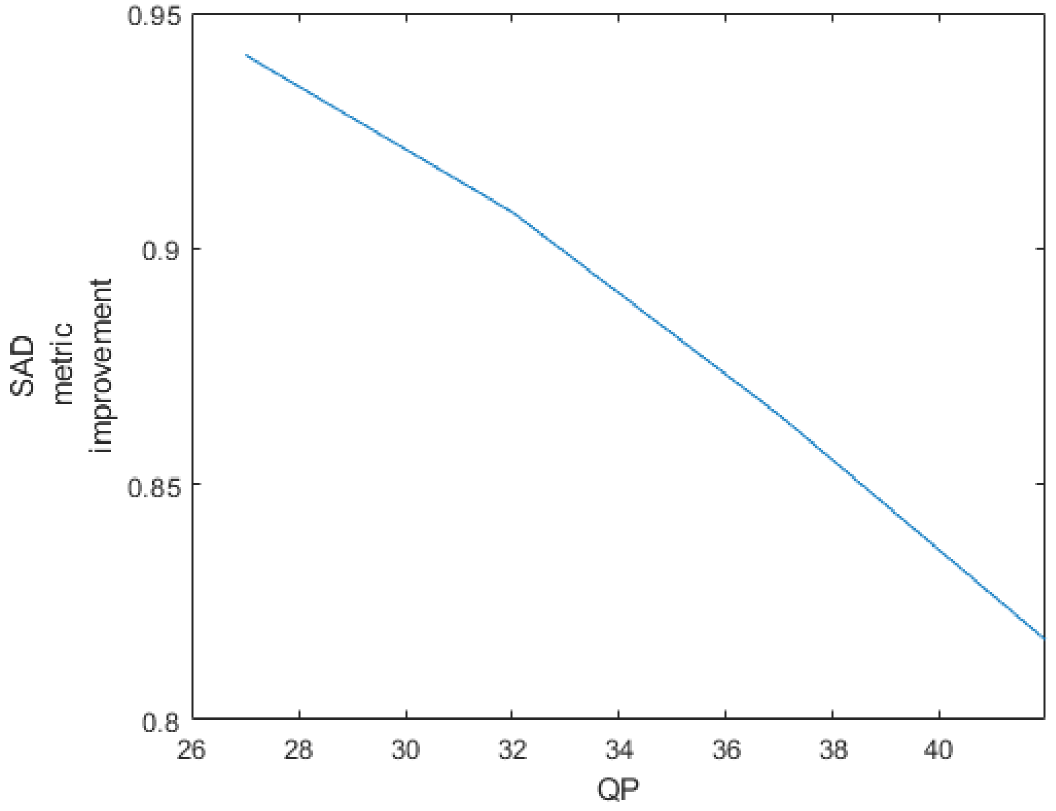


Figure 9-3: Average improvement as a function of QP value for motion compensation metric when using our method instead of the naive method for 'Silent' sequence.

is already highly optimized for motion vector encoding. In some simple attempts to improve using the codewords that have been designed for motion vectors, we realized the improvement was small compared to what is obtained by most standards.

From a theoretical perspective, our results do show that an algorithm can be designed to adapt the side-information precision based on information available at the decoder. In our setup defined in Chapter 4, we proposed two questions, the first being finding the size N of the set $A(n)$ of side-information parameters, and finding the ideal set $A(n)$ given that length. We assumed that, in the case of motion compensation, most motion vectors have the same probability of being chosen. That is definitely not true. Optimizing the codebook to be used in the case where we opt not to use fractional interpolation will be a significant step in the design of any future motion compensation side-information reduction system.

Summary

In this Chapter, we observed the results of applying our method to motion compensation side-information. In the simplified scenario of assuming motion vectors are uniformly encoded, we found that we can reduce the side-information bit-rate significantly for images of 720p and 1080p resolution. This does not remain true in the scenario where we do not use uniform codewords for our motion vectors, as they are highly optimized, and not optimized to the multiple set scenario. In this case, using uniformly designed set of prediction parameters is highly suboptimal, so there is still significant room for growth in the fully designed scenario.

Chapter 10

Summary and Future Directions

In this thesis, we began by considering image and video compression. Most modern encoders follow a pattern of prediction, followed by transform, quantization and entropy encoding. The objective of the prediction is to reduce the energy in the signal by reducing its redundancy with previous blocks. The transform is used to concentrate the energy in as few coefficients as possible. Quantization represents these coefficients with a small set of integer (or fixed) values, while entropy encoding represents these values as a series of bits that can be transmitted.

Many prediction systems use some kind of parameterized predictor. By transmitting a certain prediction parameter, the encoder tends to significantly reduce the energy in prediction residuals. For example, directional intra-prediction is typically done by selecting a prediction direction and forming a prediction by copying pixels down in that direction. Motion compensation is typically done by selecting and transmitting a motion vector referring to the previous frame. The difference between the referred block in the previous frame and the current block is then transmitted in this case.

A significant issue with parameterized predictors is that the predictor must be encoded. This means a fraction of the bit-rate will now be used to account for intra-prediction mode or motion vectors. As current standards tend to use finer intra-prediction modes and more precise fractional motion vectors, these will occupy an ever higher fraction of the bit-rate. In this thesis, we reconsidered this trade-off, and proposed to select whether or not to use a larger set adaptively, in order to gain an advantage.

In Chapter 4, we first presented a framework to consider this parameterized prediction and its side-information domain. We take our parameter to be chosen from a set of prediction parameters. We note that, although this set is often assumed to be fixed, there is no reason for this to be the case. We propose that this set should adapt to local content from previous blocks. In the context of this thesis, we require the set to be decided only on previously encoded blocks, since this will not incur in any additional side-information bits. We related the selection of this set to the problem of quantizer design, and the concept of an indirect quantizer, where we quantize a certain parameter in order to encode something else, and the error is given by the error obtained when decoding the estimated value.

A natural question is what optimal set should be used? The design of the adaptive set to be chosen in each block boils down to two questions, to be answered in a block-by-block basis and based on data from previously decoded blocks:

(a) What is the number of possible side-information levels that we should use in a particular block?

(b) Given the number of side-information levels we wish to use, what are the optimal levels that should compose this set?

In Chapter 5, we first considered applying this framework to intra-prediction, by deriving an algorithm based on the intra-prediction residual inaccuracy model as presented in [22]. Our first method for intra-prediction selects between a set of 7 and a set of 35 intra-prediction directions. The set of 7 is given by uniformly spaced directions shown in Figure 5-1, while the set of 35 directions is given by the full set of directions in HEVC. We choose between these two sets by observing the maximum absolute gradient in the decoded block-boundary. For blocks with small gradient boundary, we use the small set. For blocks with large gradient boundary, we use the large set. This means we attempt to encode our block more precisely whenever we expect to have a large edge in this block, as in this case the imprecision would lead to larger residuals, and encode our block less precisely when we expect it to have a large edge.

In terms of our theoretical framework, what this meant is that we were focusing on the question of determining the number of side-information levels to be used in each block. We restricted this to two possibilities, and simply designed a criterion based on previously decoded data to select between them. We assumed for now that the ideal answer for the levels to be used would be approximately given by the uniformly designed levels.

When applying this algorithm, we observed that it was indeed able to significantly improve the encoder performance, as shown in Chapter 6. There were two main advantages that we found in this method. The first one is that in many blocks, using a smaller set of prediction directions will mean a smaller number of possible side-information parameter values, and therefore a smaller number of bits used in these blocks. The other advantage is what we referred to as the most-probable mode (MPM) effect. Since blocks are encoded using non-uniform codewords where the smaller codewords are given to more probable levels, often the optimal level from an energy reduction perspective will not be optimal from a rate-distortion perspective. For these blocks, using a smaller set will lead to a smaller cost to non-MPM modes. This means the optimal mode from an energy reduction perspective will be encoded using fewer bits, and our method may indeed select a codeword that uses more bits, leading to a larger number of bits when using the smaller set, but improved overall encoder performance.

In Chapter 7, we considered two ways we could improve the side-information reduction system designed in Chapter 5. First, we proposed to select from a larger number of possible cardinalities to the set of intra-prediction directions. Instead of restricting ourselves to 7 or 35 modes, we allow ourselves to use 7, 11, 19 or 35 modes. We select among these by thresholding the maximum absolute gradient squared with scaling factors by 4 in between sets, based on the prediction inaccuracy model for intra-prediction. In terms of our theoretical setup, we are merely extending a bit the range of possibilities in question (a), while still keeping the uniformity assumption for question (b). As we observed, this led to a small improvement in relation to what was previously established in Chapter 6.

We then proposed to consider the second question, by using only two possible cardinalities, but adapting the prediction directions to the content in previous blocks. Specifically, we design these directions by adding the neighboring directions to the directions used in the block's neighbors. This takes into account the fact that edges tend to extend between multiple blocks with smaller variations. This unfortunately did not lead to any gain when compared to what was found before.

In Chapter 8, we considered how we could adapt side-information reduction to the case of motion compensation. For motion compensation, motion vectors are encoded using fractional precision. This fractional precision is what leads to larger codewords. Therefore, we propose to use fractional precision only for some of the blocks, decided by looking at the gradient of the block given by the predicted motion vector in the previous frame. The idea is again to look for whether or not the current block has a large edge or a small edge, and to use a fine set of motion vectors only in the blocks with large edges.

In Chapter 9, we observed that, in a simplified scenario, reducing the side-information can lead to an improvement in encoder performance. The main issue is that it does not lead to the same improvement when using the codebook designed for HEVC. HEVC uses a non-uniform codebook, which exploits the same redundancy in between blocks as our method does, to lead to more efficient encoding. To apply side-information reduction directly to HEVC would require a significant amount of work in optimizing the non-uniform codewords, and designing the set of possible directions to be used within the adaptive set.

To conclude, we believe side-information reduction should be an active research direction for future standards. As we use more precise motion vectors and intra-prediction directions, we will require adaptive ways to deal with the large amount of associated side-information. Future work should be done in future standards such as VVC [5], to consider different ways side-information is contributing to the overall bit-rate, and how adaptive methods can be used to reduce it.

New types of multimedia applications which utilize video compression may also be directions where methods to reduce side-information can lead to significant compression improvement. For example, in the case of three-dimensional stereoscopic video sequences or multiview video sequences [28, 29], frames accounting for different views are highly correlated. Using the side-information parameters in a view as a way to reduce side-information in the other view can lead to significant encoding gains. Similarly, in the case of scalable video encoding [30, 31], one can adapt the side-information precision for higher resolutions based on the parameters used in lower resolutions. Given the increasing use of sophisticated prediction mechanisms in the coding techniques for these new types of media, and therefore the increasing need for side-information and the need to compress this side information, this suggests that the research directions identified in this thesis may increase in importance in the future.

Bibliography

[1] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, “Overview of the High Efficiency Video Coding (HEVC) Standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, pp. 1649–1668, dec 2012.

[2] I. E. G. Richardson, *H.264 and MPEG-4 Video Compression.* Wiley, 2003.

[3] A. Skodras, C. Christopoulos, and T. Ebrahimi, “The JPEG 2000 Still Image Compression Standard,” *IEEE Signal Processing Magazine*, vol. 18, pp. 36–58, sep 2001.

[4] I. Kim, J. Min, T. Lee, W. Han, and J. Park, “Block partitioning structure in the hevc standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1697–1706, 2012.

[5] B. Bross, “Versatile video coding (draft 10) document jvet-s2001,” *Joint Video Experts Team (JVET)*, June 2020.

[6] J. Jain and A. Jain, “Displacement Measurement and Its Application in Interframe Image Coding,” *IEEE Transactions on Communications*, vol. 29, no. 12, 1981.

[7] J. S. Lim, *Two-Dimensional Signal and Image Processing.* Prentice Hall, 1989.

[8] K. Karhunen, *Uber lineare Methoden in der Wahrscheinlichkeitsrechnung*, vol. 37. 1947.

[9] N. Ahmed, T. Natarajan, and K. R. Rao, “Discrete Cosine Transform,” *IEEE Transactions on Computers*, vol. 100, pp. 90–93, jan 1974.

[10] F. Kamisli and J. S. Lim, "1-D Transforms for the Motion Compensation Residual," *IEEE Transactions on Image Processing*, vol. 20, no. 4, pp. 1036–1046, 2011.

[11] H. Zhang and J. S. Lim, "Analysis of One-Dimensional Transforms in Coding Motion Compensated Prediction Residuals for Video Applications," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012.

[12] A. Saxena and F. C. Fernandes, "DCT/DST-based transform coding for intra prediction in image/video coding," *IEEE Transactions on Image Processing*, vol. 22, no. 10, pp. 3974–3981, 2013.

[13] R. G. Gallager, *Principles of Digital Communication.* Cambridge University Press, 2008.

[14] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, no. 28, pp. 129–137, 1982.

[15] D. A. Huffman, "A Method for the construction of minimum redundancy codes," *proc. IRE*, pp. 1098–1101, 1952.

[16] T. M. Cover and J. A. Thomas, *Elements of Information Theory.* Wiley, 2 ed., 2006.

[17] V. Sze, M. Budagavi, and G. J. Sullivan, *High Efficiency Video Coding (HEVC).* Springer International Publishing, 2014.

[18] X. Cai and J. S. Lim, "Transforms for intra prediction residuals based on prediction inaccuracy modeling," *International Conference on Image Processing (ICIP)*, pp. 4401–4405, 2015.

[19] X. Cai and J. S. Lim, "Transforms for Intra Prediction Residuals Based on Prediction Inaccuracy Modeling," *IEEE Transactions on Image Processing*, vol. 24, no. 12, pp. 5505–5515, 2015.

[20] X. Cai and J. S. Lim, "H.264 intra coding with transforms based on prediction inaccuracy modeling," *International Conference on Image Processing (ICIP)*, pp. 2380–2384, 2016.

[21] X. Cai and J. S. Lim, "Transforms for Motion-Compensated Residuals Based on Prediction Inaccuracy Modeling," *Data Compression Conference Proceedings*, p. 583, 2016.

[22] L. Nissenbaum, J. S. Lim, and M. Jin, "Intra-prediction side-information reduction based on gradient boundary," in *Data Compression Conference*, 2019.

[23] Y. Piao, J. Min, and J. Chin, "Encoder Improvement of Unified Intra Prediction," in *JCT-VC207*, 2010.

[24] A. Ortega and K. Ramchandran, "Rate-Distortion Methods for Image and Video Compression," *Signal Processing Maganize*, vol. 15, pp. 23–50, nov 1998.

[25] G. Bjontegaard, "Calculation of Average PSNR differences between rd-curves," *VCEG Contribution VCEG-M33*, apr 2001.

[26] M. Zhang, C. Zhao, and J. Xu, "An adaptive fast intra mode decision in hevc," in *2012 19th IEEE International Conference on Image Processing*, pp. 221–224, 2012.

[27] T. Zhang, M. T. Sun, D. Zhao, and W. Gao, "Fast Intra-Mode and CU Size Decision for HEVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 8, pp. 1714–1726, 2017.

[28] L. Guan, *Multimedia image and video processing.* CRC press, 2017.

[29] Y. Ho and K. Oh, "Overview of multi-view video coding," in *2007 14th International Workshop on Systems, Signals and Image Processing and 6th EURASIP Conference focused on Speech and Image Processing, Multimedia Communications and Services*, pp. 5–12, 2007.

[30] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the h.264/avc standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 9, pp. 1103–1120, 2007.

[31] J. M. Boyce, Y. Ye, J. Chen, and A. K. Ramasubramonian, "Overview of shvc: Scalable extensions of the high efficiency video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 1, pp. 20–34, 2016.