University of California
Santa Barbara

Towards Prediction Optimality in Video Compression and Networking

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

Doctor of Philosophy
in
Electrical and Computer Engineering

by

Shunyao Li

Committee in charge:

Professor Kenneth Rose, Chair
Professor B. S. Manjunath
Professor Shiv Chandrasekaran
Professor Nikil Jayant

March 2018

The Dissertation of Shunyao Li is approved.

Professor B. S. Manjunath

Professor Shiv Chandrasekaran

Professor Nikil Jayant

Professor Kenneth Rose, Committee Chair

February 2018

Towards Prediction Optimality in Video Compression and Networking

by

Shunyao Li

To my beloved family

Acknowledgements

The completion of this thesis would not have been possible without the support from many people.

I would like to thank my advisor, Prof. Kenneth Rose, for his guidance and support throughout the entire course of my doctoral research. He always encourages us to think about problems from the fundamental theory of signal processing, and to understand deeply the reasons behind the experiments and observations. He also gives us enough freedom and support so that we can explore the topics that we are interested in. I enjoy every constructive and inspiring discussion with him, and am genuinely grateful for developing a solid, theoretical and critical mindset under his supervision.

I would also like to thank Prof. Manjunath, Prof. Chandrasekaran and Prof. Jayant for serving on my doctoral committee and providing constructive suggestions on my research. The research in this dissertation has been partly supported by Google Inc, LG Electronics and NSF. Thanks to our sponsors.

I thank all my labmates for the collaborations and discussions. Special thanks to Dr. Tejaswi Nanjundaswamy, who has been a great mentor and collaborator throughout most of my doctoral study. He was actively engaged in almost all of my research projects, and offered practical suggestions and guidance on academic writing. I also want to thank Yue for her help during my first two years at UCSB, both inside and outside the lab. My thanks also go to other SCL members and alumni for all the encouragement and help in my research, and my lovely roommates and friends for making these years special and unforgettable.

Finally, deeply gratitude to my dearest family for their unconditional love.

Curriculum Vitæ

Shunyao Li

Education

2018	Ph.D. in Electrical and Computer Engineering (Expected), University of California, Santa Barbara.
2014	M.S. in Electrical and Computer Engineering, University of California, Santa Barbara.
2013	B.S. in Electronic Engineering, Tsinghua University, Beijing, China

Professional Employment

2013-2018	Graduate Researcher, Department of Electrical and Computer Engineering, University of California, Santa Barbara.
2015	Research Intern, Chrome Media Team, Google Inc., CA
2014	Teaching Assistant, Department of Electrical and Computer Engineering, University of California, Santa Barbara.
2014	Research Intern, Mobile Research Lab, LG Electronics, CA
2013	Undergraduate Researcher, Tsinghua University, Beijing, China

Publications

- S. Li, T. Nanjundaswamy, B. Li, and K. Rose, "Towards Optimality in Transform Domain Temporal Prediction", to submit to *IEEE Trans. on Image Processing.*
- S. Li, T. Nanjundaswamy, B. Li, and K. Rose, "On Generalizing the Estimation-theoretic Framework to Scalable Video Coding with Quadtree Structured Block Partitions", *Proc. IEEE Intl. Conf. on Image Processing (ICIP)*, 2017.
- D. Mukherjee, S. Li, Y. Chen, A. Anis, S. Parker, J. Bankoski, "A Switchable Loop-restoration with Side-information Framework for the Emerging AV1 Video Codec", *Proc. IEEE Intl. Conf. on Image Processing (ICIP)*, 2017.
- S. Li, T. Nanjundaswamy, and K. Rose, "Jointly Optimized Transform Domain Temporal Prediction and Sub-pixel Interpolation", *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, Mar. 2017.
- S. Parker, Y. Chen, J. Han, Z. Liu, D. Mukherjee, H. Su, Y. Wang, J. Bankoski, S. Li, "On transform coding tools under development for VP10", *SPIE Optical Engineering+ Applications.* International Society for Optics and Photonics, 2016.
- S. Li, T. Nanjundaswamy, and K. Rose, "Transform Domain Temporal Prediction with Extended Blocks", *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, Mar. 2016.

- S. Li, T. Nanjundaswamy, Y. Chen, and K. Rose, "Asymptotic Closed-loop Design for Transform Domain Temporal Prediction", *Proc. IEEE Intl. Conf. on Image Processing (ICIP)*, Sep. 2015. (**Top 10% paper award**)
- S. Li, O. Guleryuz, and S. Yea, "Reduced-rank Condensed Filter Dictionaries for Inter-picture Prediction", *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, April. 2015.
- S. Li, Y. Chen, J. Han, T. Nanjundaswamy, and K. Rose, "Rate-Distortion Optimization and Adaptation of Intra Prediction Filter Parameters", *Proc. IEEE Intl. Conf. on Image Processing (ICIP)*, Oct. 2014.
- J. Wen, B. Li, S. Li, Y. Lu and P. Tao, "Cross Segment Decoding of HEVC for Network Video Applications", *Proc. IEEE Intl. Packet Video Workshop (PVW)*, Dec 2013.
- J. Wen, S. Li, Y. Lu, M. Fang, X. Dong, H. Chang and P. Tao, "Cross Segment Decoding for Improved Quality of Experience for Video Applications", *IEEE Data Compression Conf. (DCC)*, Mar 2013.

Abstract

Towards Prediction Optimality in Video Compression and Networking

by

Shunyao Li

In modern video compression and communication systems, prediction is one of the key schemes to exploit spatial and temporal redundancies. However, current approaches are suboptimal as they do not fully exploit the spatial and temporal correlations within signals. This dissertation focuses on the optimal prediction algorithms that fully utilize the correlations, and the optimal design of predictors that accounts for the rich variety of video statistics as well as the instability due to quantization error propagation in the closed-loop video coding system. Complementary to predictive coding, we also expand the design framework to the general predictive coding system, focusing on the optimal transform design that spatially de-correlates the residual data, leading to better compactness and compression performance.

The contributions in this dissertation cover the topics of spatial (intra) prediction, temporal (inter) prediction, the layered prediction in scalable coding and transform design. The contributions have been proposed to or accepted in multiple video coding standardization efforts including the Moving Picture Experts Group (MPEG) and the Alliance for Open Media (AOM), and have provided significant improvements in the video compression performance.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 An Overview of Video Coding

Video applications (storage, streaming, real-time communication, etc.) have long been an essential part of the modern technology and have been developing rapidly over the past thirty years. High video quality requirements and limited bandwidth lead to the growth and standardization of video compression technology. The latest video coding standard, High Efficiency Video Coding (HEVC) [1], also known as H.265 and MPEG-H Part 2, was developed by the Joint Collaborative Team on Video Coding (JCT-VC) in 2013. Since then, it has been a benchmark for both the industry and the academia when new compression algorithms are proposed. Besides the effort of JCT-VC, another joint development project named the Alliance for Open Media (AOM) was launched in 2015 focusing on delivering the next-generation royalty-free video codecs. Their open-source video codec AV1 will be supported by major browser vendors, content providers and hardware manufacturers, and has become a major player in the video compression industry. Its predecessor, VP9 [2], was developed by Google and has been another benchmark while new algorithms are proposed to AV1.

The basic video compression framework includes four major components: prediction, transform, quantization and entropy coding. A simplified block diagram is shown in Fig. 1.1, where the four components are represented as P, T, Q, EC respectively. A video frame is first divided into blocks. For each block x, the encoder first predicts the block from available relevant information, and gets the prediction $\tilde{x}$. Then it computes the prediction error (a.k.a. residual) $e = x - \tilde{x}$, and convert it into the transform domain E. To make sure the bitstream size matches the bandwidth condition, it uses a quantizer to compress the values in E to a prescribed set of values, introducing a certain amount of distortion (a.k.a. quantization error or reconstruction error). The trade-off between rate (bitstream size) and distortion (quality) is carefully balanced to either minimize the bitrate maintaining the quality, or minimize the distortion given a target bitrate. To minimize the bits needed for the quantized values, an entropy coder is used to optimize the codebook, which is used by both encoder and decoder to convert between the bits and quantized values. Those quantized values are thus converted to bits and written to the bitstream.

At the decoder side, it receives the bitstream and decodes it using the same codebook, then dequantizes it to reconstruct the transform domain residual $\hat{E}$. An inverse transform is applied on $\hat{E}$ to convert it back to the pixel domain $\hat{e}$. The reconstructed residual $\hat{e}$ is usually different from the real residual e due to the information loss at the quantizer. In the exact same way as encoder, the decoder would compute a prediction $\tilde{x}$ based on available reconstructed information. The reconstruction of the block $\hat{x}$ is then generated by adding together the reconstructed residual $\hat{e}$ and $\tilde{x}$. Since the blocks are coded sequentially, the reconstruction $\hat{x}$ is then used to predict future blocks, forming a predictive loop. In this dissertation, we consistently refer to prediction using notation $\tilde{}$, and reconstruction using $\hat{}$, which are commonly used in academic papers in this field.

Each of the four components plays an important role in the system. Prediction

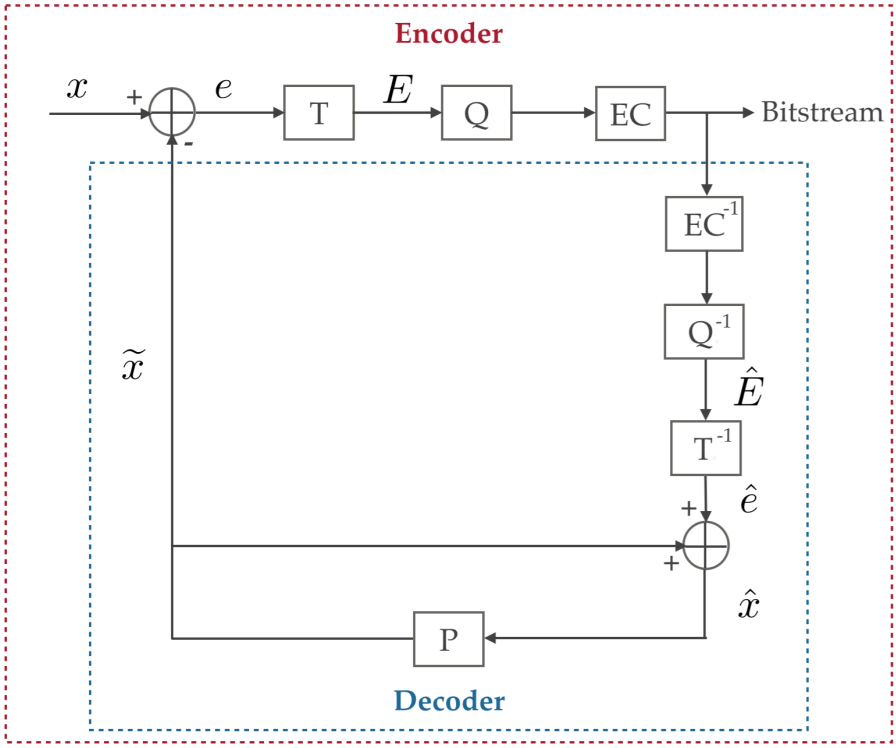


Figure 1.1: Video codec block diagram

exploits the correlation between pixels (both spatially between neighboring pixels and temporally between adjacent frames). Transform furthers removes the redundancy within the residual block, and redistributes the energy more compactly so that the quantizer can achieve a better rate-distortion (RD) trade-off. Quantizer decides the information to remove (usually high frequency signals), and optimizes the bit allocation based on the bitrate requirement. Entropy coding optimizes the codebook based on the prior probability information to minimize the average number of bits needed. Regardless of the codec development organization, almost all the modern video codecs follow the similar basic workflow.

To compromise the trade-off between rate and distortion, a metric named RDcost is introduced to measure both the deviation from the source data and the number of

bits needed to encode the residual as well as the coding decisions. Mathematically, it is defined as

$$\text{RD cost} = \text{rate} + \lambda \text{distortion} \tag{1.1}$$

where rate is measured by the number of bits, and distortion is measured as the mean square error between source and reconstruction. Usually, an encoder tries out all the possible coding decisions (block sizes, prediction modes, transform modes, quantizer parameters) and chooses the one that yields the minimum RD cost. The encoder decisions are sent to the decoder as extra overhead.

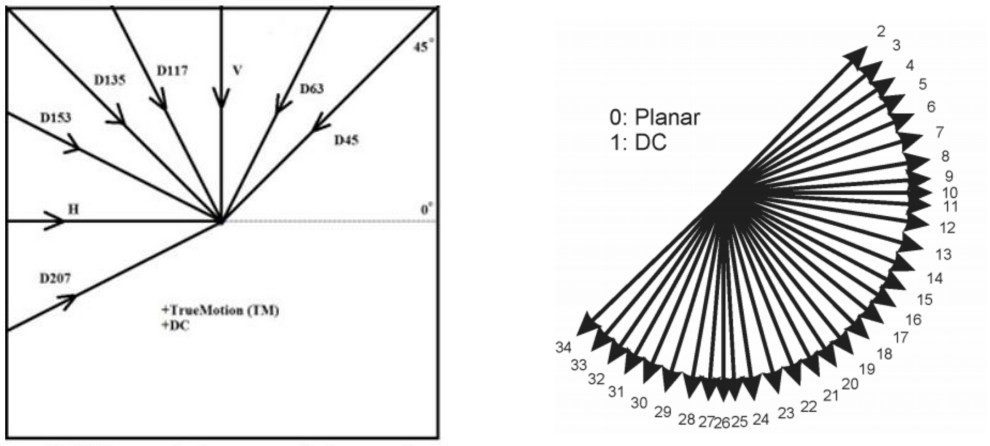
To better target at specific applications like video streaming or video conferencing, several extension have been adopted along with the video compression standards. Scalable video coding [3, 4] allows the videos to be encoded progressively at different settings (quality, resolution or frame rate), thus depending on the network condition, one or more sets of bitstream can be transmitted to the decoder for reconstruction at different levels. Other extensions include screen content coding (for video containing rendered graphics, text, or animation), multi-view extensions (for multiple camera views), range extensions (for HDR video or content production), etc.

In this dissertation, we will mainly focus on optimizing the prediction module in regular video coding system as well as the scalable video coding system. We also extend some of the design to transform coding in general predictive systems. Some of the contributions in this dissertation have been proposed to AOM and adopted by the AV1 codec. Some other work have gained attentions from major contributors in JCT-VC for the development of the next generation video codec, H.266.

1.2 Prediction in Video Coding

Prediction is widely used in video coding to remove redundancies by exploiting the correlation between pixels. For regular 2D video frames, there are two major types of prediction: intra prediction (a.k.a. spatial prediction) which uses the neighboring pixels information within the same frame, and inter prediction (a.k.a. temporal prediction) which uses the information in previous frames. Traditionally in intra prediction, the pixels on the block boundary are copied along a specific direction to generate the prediction for the block, wherein the directionality depends on the local texture. Each direction is considered as an intra mode. The encoder would search for the best intra mode for prediction, and send the mode index to the decoder. The number of modes varies in different codecs. AV1's predecessor, VP9, has 10 directional intra modes (Fig. 1.2a), while HEVC has 36 directional intra modes (Fig. 1.2b). [5] discussed about the intra prediction efficiency in HEVC and VP9. In inter prediction, the encoder predicts a block from a similar reference block in previously reconstructed frames. The reference block is described by the index of the reference frame and a motion vector between the coordinates of the two blocks. Usually the codec maintains a list of reference frame candidates based on the distance to the current frame and the reconstruction quality. In scalable video coding [3, 4], the reconstruction of the same frame at different quality levels are added to the candidate lists as well.

In general predictive coding systems, the prediction is used as reference for coding future blocks or frames, forming a closed prediction loop system. This prediction loop structure is widely used in all kinds of signal compression system (e.g.: audio, video, etc.). The design framework proposed in this dissertation widely applies to any predictive coding system, thus is of significantly broad interests.



(a) Intra prediction modes in VP9 (b) Intra prediction modes in HEVC

Figure 1.2: The intra prediction modes in HEVC and VP9
(From 1051-8215/$31.00 © 2012 IEEE and 1545-0279/$31.00 © 2015 IEEE)

1.3 Challenges and Limitations

The challenges of prediction mainly come from two parts. First, the underlying coupled spatial and temporal correlation is complicated and hard to describe in a simple prediction model. This leads to a line of research in designing better prediction models (e.g.: joint spatial-temporal prediction [6, 7, 8], combined intra prediction [9], 3D sub-band coding [10, 11], multi-tap filtering [12, 13, 14], etc.). Second, the correlation varies considerably in natural video sequences and it cannot be well described using a single model. Usually, multiple modes are designed for the encoder to choose from. The encoder may choose one mode that works the best for the block, frame or sequence (depending on the flexibility level), and signal the mode index in the bitstream with extra overhead. Therefore, another line of research is about optimizing the mode design to accurately fit the statistics.

The approaches often used in the mode design generally fall into two categories: online adaptation and offline training. Online adaptation updates the model on-the-fly at both encoder and decoder as the codec proceeds through the frames [15, 16].

However, many applications require the decoding to be performed in real time on low-cost devices, thus the decoders can not afford to support complicated online adaptation algorithms. Offline training, on the other hand, can be extensive and comprehensive. The models are designed offline, based on the extracted statistics from a training set, and later are applied on the actual real world data. The training algorithms are becoming more and more accessible and effective recently thanks to the rapidly growing of machine learning techniques [17, 18, 19]. However, as we will discuss later in Chapter 3, a severe design instability problem occurs when we optimize for a closed-loop system, where the reconstructed data will be used to encode future data. In lossy compression, due to the quantization error, the reconstructed data is not the same as the original data. So any changes in reconstruction after applying the designed predictor will propagate to the future through the prediction loops. As the changes accumulate, the statistics become mismatched with the ones we designed for, which lead to design instability. In the low bitrate scenario, this issue becomes more severe as more quantization error propagates through the loop.

1.4 Dissertation Organization

The rest of this dissertation is organized as follows. In Chapter 2, we extend the idea of Transform Domain Temporal Prediction (TDTP) [20] where we decorrelate the spatial correlation in the transform domain and do an optimal one-to-one temporal prediction for transform coefficients. We introduce EBTDTP and the joint design of EBTDTP with interpolation filters to account for its coupled interference with each other.

While training the joint predictors offline, we noticed a prevalent and catastrophic instability problem that applies to all the offline design for closed-loop systems (like video coding), which would lead to growing deviation in statistic through the prediction

loop. We address this instability problem in Chapter 3, and tackle it using the two-loop Asymptotic Closed-loop (ACL) design approach. Further, we incorporate it with the K-modes clustering approach to optimize the mode design. With this design system, we achieve significant compression gain over the baseline codec using the joint predictors.

In Chapter 4, we focus on the intra prediction and propose a 2D non-separable Markov model, where each pixel is predicted using a four-tap filter from its neighboring pixels to cover all the possible prediction directions. It is the result of a collaboration with Yue Chen, and has previously appeared in the [21]. It is partly reproduced here with the permission of Yue Chen. Compared to [21], this chapter focuses more on the individual contribution of rate-distortion optimization and adaptation of the filter design. Since the filter design also suffers from the same instability problem, we further incorporate it with the ACL approach for stable intra predictor design.

In Chapter 5, we improve on the enhancement layer prediction in scalable video coding, by generalizing the estimation-theoretic (ET) framework [22, 23, 24] to support the various new coding tools (e.g.: quadtree structured partitioning, hybrid transform and the RDOQ adjustment). The generalized framework is compatible with all the existing features in the latest scalable video codec, SHVC, with no additional overhead and negligible additional complexity.

In addition to all the advances proposed for the prediction module, we also extend the two-loop ACL approach to other components in the predictive coding system, such as transform design, proposing an ACL-based KLT design for temporal prediction residual in Chapter 6. We show that the ACL-designed transforms outperform the non ACL-designed transforms consistently, proving that the ACL design scheme is not only useful in predictor design, but also for other components in the video compression system.

Chapter 7 concludes the dissertation and suggests directions for future research.

Chapter 2

Transform Domain Temporal Prediction

Current temporal prediction relies on motion-compensated pixel-to-pixel copying techniques, which is suboptimal since it ignores the underlying spatial correlation between neighbor pixels. Transform Domain Temporal Prediction (TDTP) was proposed in our lab to decouple the spatial and temporal correlation by doing an optimal one-to-one prediction in the transform domain, where little spatial correlation remains. It further recognizes the variation in the true temporal correlation across frequencies, which is usually unidentifiable in the pixel domain dominated by low frequencies. In this chapter[1], we focus on the optimal design of TDTP which: *i)* fully exploits spatial correlations both inside and outside block boundary, *ii)* fully accounts for the coupled interference with sub-pixel interpolation filters, *iii)* circumvents the challenge of catastrophic design instability due to quantization error propagation through the prediction loop, and *iv)* employs effective mode design to cover a variety of statistics. Experimental results validate the

[1]This chapter is adapted from 978-1-4799-9988-0/16/$31.00 © 2016 IEEE and 978-1-5090-4117-6/17/$31.00 © 2017 IEEE.

effectiveness of the designed TDTP system, and can achieve an average of 4.2% reduction in bitrate over the state-of-the-art HEVC codec.

2.1 Introduction

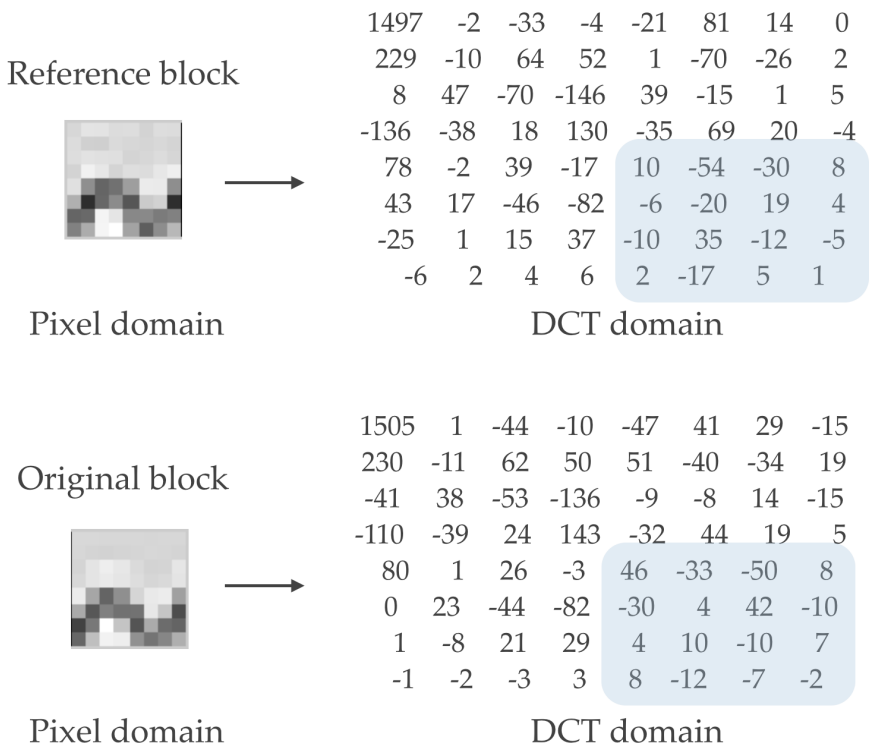
Modern video coding standards, such as HEVC, exploit the inherent temporal dependencies in a video sequence via inter prediction (or temporal prediction). Instead of directly encoding the raw pixel values for each block, the encoder predicts them from a similar reference block in previously reconstructed frames through pixel domain block matching (a.k.a. motion compensation). The prediction error is then transformed, typically by the discrete cosine transform (DCT), and the transform coefficients are quantized and coded.

Over the past decade, a considerable amount of research has been focused on the optimality of motion compensated prediction. One research direction [25, 26, 27, 28, 29, 30] has focused on effectively describing the motion that objects follow. [28] significantly improved the motion precision to quarter-pixel level by introducing sub-pixel motion compensation; [29, 30] extends the motion representation from 2D translational motion vector to a 3D affine transformation. Another line of the research [31, 32, 33] focuses on finer block partition by introducing quadtree, variable block sizes and segmentation for rigid objects.

Despite all the efforts on improving the motion accuracy and granularity, all of the work described are based on the one-to-one pixel matching criteria in searching for the most similar reference block. However, this one-to-one pixel matching fails to account for the underlying spatial correlation between pixels. Ideally, each pixel needs to be matched with some combination of its neighboring pixels according to its specific spatial correlation. This is an extremely hard problem for two reasons: *i)* Given the rich variety

of the spatial correlations in a natural video sequence, it is almost impossible to capture them precisely; and *ii)* the spatial correlations and temporal correlations are mingled together, resulting in a complicated 3D problem. Prior works on this include: [12, 13, 14] introduced multi-tap filtering in inter prediction to capture complicated transitions (e.g.: lighting changes); [10, 11] solves the complicated 3D problem using 3D subband coding; [6, 7] proposed joint spatial-temporal prediction, but under a much more simplified assumption. However, these approaches either suffer from high encoder complexity or over-simplify the problem.

A more effective approach to model complex spatio-temporal correlation is via DCT-domain temporal prediction, where spatial decorrelation is largely achieved first and allows for optimality of subsequent one-to-one transform coefficient prediction. The temporal correlation ρ for each transform coefficient varies across frequencies, which is usually overlooked in the traditional pixel domain motion compensation. For example, in Fig. 2.1(a), the reference block we find in the pixel domain using the traditional approach looks very similar to the original block. In the DCT domain, this is also true for lower frequency components. However, the similarity breaks down for higher frequency coefficients where more details and noise are. As a result, the temporal correlation ρ is close to 1 at lower frequencies and generally decreasing as frequencies become higher, as shown in Fig. 2.1(b). This variation in correlation across frequencies is masked in the pixel domain by the dominant low frequencies, and the resulting $\rho \approx 1$ led to the prevalence of block matching and copying techniques in current coders. Thus, the advantages of transform domain temporal prediction (TDTP) can be viewed from two perspectives: *i)* an effective paradigm to disentangle spatial and temporal correlations allowing for optimal prediction, and *ii)* a means to make explicit, and hence properly account for, the variation in temporal correlation across frequency, which is otherwise hidden in the pixel domain.



(a) Reference block and original block in pixel and DCT domain

0.999	0.998	0.997	0.970	0.944	0.930	0.842	0.808
0.996	0.978	0.979	0.963	0.957	0.884	0.900	0.797
0.983	0.984	0.975	0.944	0.978	0.931	0.857	0.794
0.967	0.980	0.977	0.965	0.958	0.920	0.930	0.768
0.960	0.950	0.962	0.964	0.942	0.889	0.904	0.756
0.927	0.938	0.934	0.922	0.919	0.882	0.831	0.748
0.898	0.881	0.919	0.906	0.869	0.815	0.700	0.512
0.835	0.760	0.826	0.769	0.717	0.640	0.470	0.339

(b) Transform prediction coefficients for 8x8 DCT coefficients for *Mobile* sequence at QP=22

Figure 2.1: An illustration of difference in correlations between pixel domain and DCT domain

The significant potential of TDTP was recognized in an earlier paper from our group [20] where a TDTP approach was proposed in conjunction with full-pixel motion, with coefficients trained from original video sequences, which yielded substantial coding gains. In this chapter, we significantly extend the TDTP approach to solve the challenges of

the interference of sub-pixel interpolation filters with TDTP. As discussed, the temporal correlation decays as the frequency gets higher. To account for the decaying correlation in TDTP, the high frequency components are attenuated more than the low frequency components. This has a similar effect as the sub-pixel interpolation filter, which is typically a low pass filter [1]. Therefore, the sub-pixel interpolation filters are interfering with TDTP and the gains are largely diluted when we move to the sub-pixel motion compensation.

In this chapter, we will introduce several techniques to solve the challenges above. First, we will introduce extended blocks based TDTP (EBTDTP) to fully disentangle the spatial and temporal correlation both inside the block as well as the neighboring pixels outside the block boundary used by the interpolation filter. Second, we jointly design the EBTDTP and the sub-pixel interpolation filters to account for the interference to optimize the overall prediction error. Third, we further investigate different types of sub-pixel interpolation filters including both separable and non-separable filters with EBTDTP.

Later in Chapter 3, we also address an instability issue in closed-loop system training, we employ an iterative open-loop design technique, leveraging inspiration from a prior work on vector quantizer design named asymptotic closed-loop (ACL) approach. Here we use a two-loop ACL approach to jointly design the EBTDTP and interpolator to circumvent the instability problem, and further combine it with a K-modes iterative design approach to design multiple modes to cover a variety of statistics.

We first validated the initial ideas in a constrained HEVC framework where only one block size was enabled and multiple coding tools (e.g. multi-hypothesis motion compensation, SAO) were disabled. The preliminary results of EBTDTP and the joint optimization of EBTDTP and interpolation filters for the constrained HEVC framework can be found in [34] and [35]; preliminary results of the ACL design for TDTP can be

found in [36]. Here, we generalize the overall design approach to support the full HEVC with all the coding tools enabled, and show the comprehensive experimental results of the proposed framework.

2.2 Background

Without loss of generality, we assume that the motion compensated reference block is in the immediate previous frame. The conventional motion-compensated prediction assumes blocks along a motion trajectory form a first-order auto-regression (AR) process

$$A_n = A_{n-1} + Z_n \tag{2.1}$$

where the A_n and A_{n-1} are the original blocks along the motion trajectory in frame n and $n-1$, and Z_n is the innovation. For decoder to mimic the same procedure as encoder, video codecs are closed-loop systems where each frame is predicted from the reconstructed version of reference frame. We use $\hat{}$ to denote reconstruction, and $\tilde{}$ to denote prediction. The AR process becomes

$$A_n = \hat{A}_{n-1} + R_n \tag{2.2}$$

where $\hat{A}_{n-1}$ is the reconstructed reference blocks in frame $n-1$ and R_n is the residual block. The motion compensated prediction $\tilde{A}_n = \hat{A}_{n-1}$.

Similarly, in TDTP, we assume the DCT coefficients of blocks along a motion trajectory form a first-order auto-regression (AR) process per frequency. We denote by x_n a DCT coefficient at a particular frequency of the original block in frame n, and by $\hat{x}_{n-1}$ the corresponding DCT coefficient of its reconstructed reference block in frame $(n-1)$,

then the AR process is given as,

$$x_n = \rho \hat{x}_{n-1} + r_n \tag{2.3}$$

where the ρ is the temporal correlation at a certain frequency (also referred to as prediction coefficient), and the r_n is the residual at the corresponding DCT frequency. The TDTP prediction at the DCT frequency is

$$\tilde{x}_n = \rho \hat{x}_{n-1} \tag{2.4}$$

Note that the conventional pixel domain block matching and copying is equivalent to employing $\rho = 1$ at all frequencies. We estimate ρ to minimize the mean square prediction error,

$$J = E((x_n - \rho \hat{x}_{n-1})^2). \tag{2.5}$$

The optimal prediction coefficient ρ is

$$\rho = \frac{E(x_n \hat{x}_{n-1})}{E(\hat{x}_{n-1}^2)}, \tag{2.6}$$

which forms the basic TDTP paradigm proposed in [20]. ρ captures the temporal dependency of DCT coefficients at a given frequency. In this model, ρ is equivalent to the temporal correlation between the DCT coefficient in the original block x_n and the corresponding DCT coefficients in the reconstructed reference block $\hat{x}_{n-1}$. One example of how ρ varies across frequencies for *mobile* sequence is shown in Fig. 2.1(b). ρ is trained offline and sent to or stored at the decoder if TDTP is enabled.

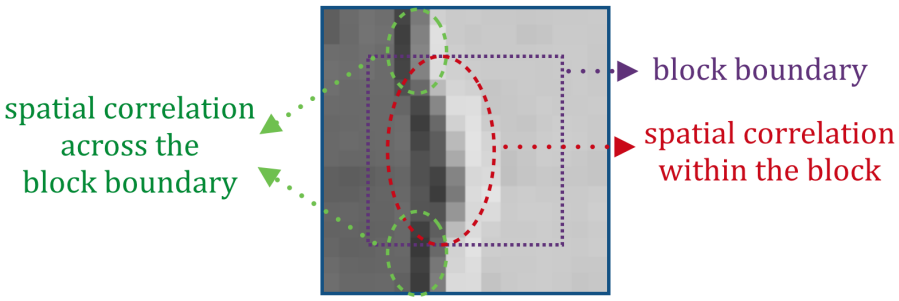


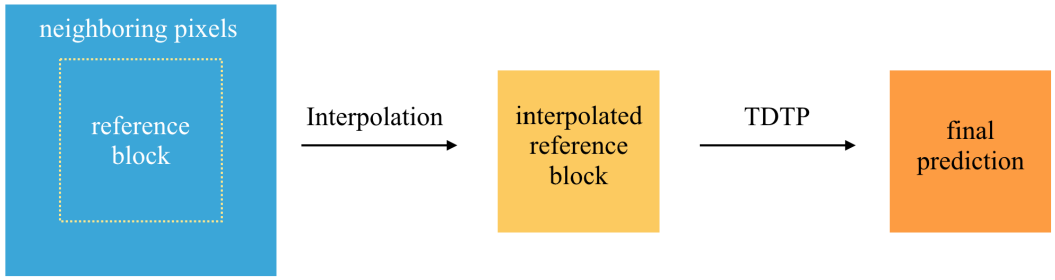
Figure 2.2: An example of spatial correlation within a block and across its boundary

2.3 Optimization for TDTP and Interpolation Filters

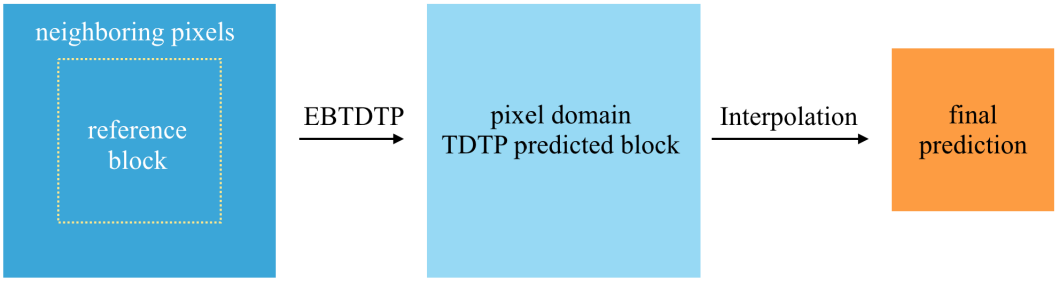
As discussed in Sec I, the sub-pixel interpolation filters interfere with TDTP since both pass low frequencies and attenuate high frequencies. In this section, we will focus on the joint design of interpolation filters and TDTP. We first propose the EBTDTP to fully disentangle the spatial and temporal correlation within the block as well as in its neighboring pixels. Then we introduce an iterative approach to jointly design the EBTDTP and interpolation filters. All the design is off-line and the designed EBTDTP parameters and interpolation filters (collectively described as "prediction parameters") will be stored at both the encoder and decoder or sent to the decoder depending on the applications.

2.3.1 Extended Block based TDTP (EBTDTP)

To design the TDTP accounting for the interference of sub-pixel interpolation filters, the first step is to include all the information used in the interpolation filters. While the traditional TDTP approach only uses information inside the block, the interpolation filter also includes neighboring pixels outside the block boundary. Naturally, the spatial correlation is never strictly limited within a block (as shown in Fig. 2.2), thus



(a) TDTP and interpolation in traditional inter prediction



(b) TDTP and interpolation in EBTDTP

Figure 2.3: An illustration to compare the tradition TDTP framework and the EBTDTP framework

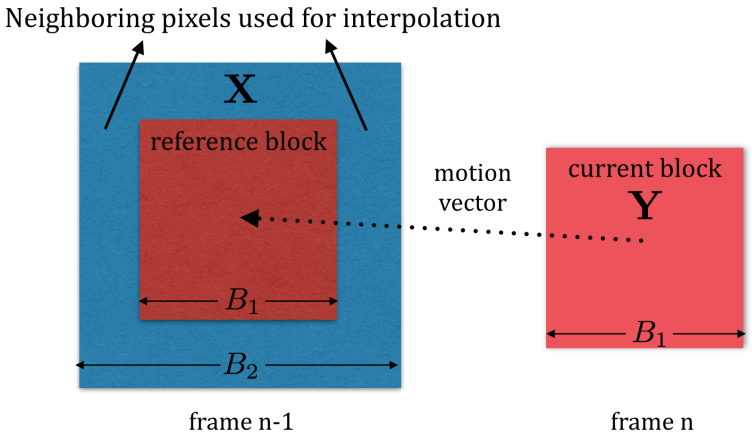


Figure 2.4: Extended reference block with neighboring pixels

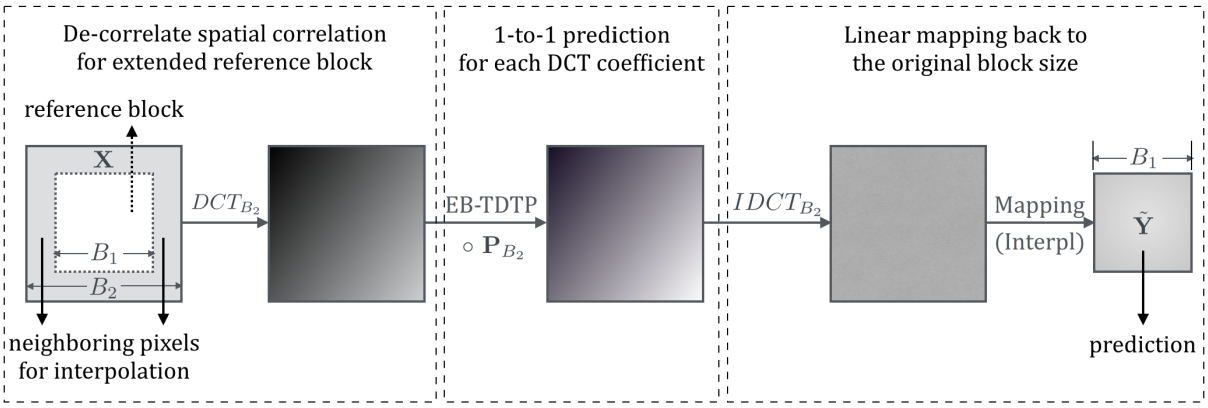


Figure 2.5: Block diagram of the proposed TDTP framework employing extended blocks

it is suboptimal if we limit TDTP within the block. From a different perspective, the interpolation filter projects the block as well as its neighboring pixels to a *subspace*, as shown in Fig. 2.3(a). Transforming this interpolated block into frequency domain, can only achieve spatial decorrelation of information in this *subspace*, which is suboptimal.

To fully disentangle the spatial and temporal correlation both inside and outside the block, we propose the extended block based TDTP (EBTDTP), as shown in Fig. 2.3(b), where we switch the order of TDTP and interpolation filter. We take an extended block centered around the reference block to cover all the neighboring pixels needed in the interpolation, and apply TDTP on it. Thus we reach spatial decorrelation of information in the whole space, and perform an optimal one-to-one temporal prediction for each transform coefficients. The interpolation filter is applied afterwards to map it to the original size.

The TDTP coefficient for each frequency is designed to minimize the final prediction error. To optimize the TDTP coefficients, we denote $\mathbf{X}$ as the extended block of size $B_2 \times B_2$ and $\mathbf{Y}$ as the current block of size $B_1 \times B_1$ ($B_2 > B_1$). Thus $\hat{\mathbf{Y}}$ is the final

prediction. The prediction error is defined as the mean square error (MSE),

$$J = \left\| \mathbf{Y} - \tilde{\mathbf{Y}} \right\|^2. \tag{2.7}$$

The vertical and horizontal interpolation filters in the matrix form are denoted as $\mathbf{F}_1$ and $\mathbf{F}_2$. Specifically, if the b-tap 1D vertical and horizontal interpolation filters are denoted as $\mathbf{f}_1$ and $\mathbf{f}_2$ (column vectors), then,

$$\mathbf{F}_1 = \begin{bmatrix} 0 & \mathbf{f}_1^T & 0 & \cdots & 0 \\ 0 & 0 & \mathbf{f}_1^T & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \mathbf{f}_1^T \end{bmatrix} \tag{2.8}$$

$$\mathbf{F}_2 = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ \mathbf{f}_2 & 0 & \cdots & 0 \\ 0 & \mathbf{f}_2 & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & \mathbf{f}_2 \\ & & & \end{bmatrix} \tag{2.9}$$

are at dimension $B_1 \times B_2$ and $B_2 \times B_1$, respectively. Therefore, the interpolated reference block is $\mathbf{F}_1\mathbf{X}\mathbf{F}_2$. The matrix operator of vertical 1D-DCT of size b is denoted as $\mathbf{D}_b$. We also define the operator $\circ$ as element-by-element multiplication. The traditional temporal prediction is the same as the interpolated reference block, i.e.,

$$\tilde{\mathbf{Y}} = \mathbf{F}_1\mathbf{X}\mathbf{F}_2. \tag{2.10}$$

The TDTP in the prior work [20] was formulated as,

$$\tilde{\mathbf{Y}} = \mathbf{D}'_{B_1}((\mathbf{D}_{B_1}\mathbf{F}_1\mathbf{X}\mathbf{F}_2\mathbf{D}'_{B_1}) \circ \mathbf{P}_{B_1})\mathbf{D}_{B_1}, \tag{2.11}$$

where $\mathbf{P}_b$ is a $b \times b$ matrix with elements as the temporal prediction coefficients, ρ, corresponding to each frequency.

The block diagram of EBTDTP is shown in Fig. 2.5. First the extended block is converted to DCT domain to achieve spatial decorrelation, then the prediction coefficients $\mathbf{P}_{B_2}$ are applied to the DCT coefficients using (2.3), and finally the prediction is converted back to pixel domain for sub-pixel interpolation. Therefore, the EBTDTP can be formulated as,

$$\tilde{\mathbf{Y}} = \mathbf{F}_1\mathbf{D}'_{B_2}((\mathbf{D}_{B_2}\mathbf{X}\mathbf{D}'_{B_2}) \circ \mathbf{P}_{B_2})\mathbf{D}_{B_2}\mathbf{F}_2. \tag{2.12}$$

To minimize the prediction error J in (2.7), we propose to jointly design the prediction coefficients $\mathbf{P}_{B_2}$ and the interpolation filters $\mathbf{F}_1, \mathbf{F}_2$ together. Note that although traditionally in video codecs, the interpolation is performed via separable filters [37, 28, 38, 39, 40], these filters sometimes cannot perfectly capture the spatial correlation. Alternatively, non-separable filters [41, 42] can be more flexible, but cover smaller spatial area if we want to maintain the same complexity as separable filters. We use 2D 4x4 non-separable filters, denoted as $\mathbf{F}$, to keep the same number of multiplications as using two 1D 8-tap filters (the current HEVC interpolation filters), and propose joint optimization approaches for both of them. In the experiment, the encoder may choose one based on the statistics of the video sequence, with a sequence-level flag. We use $\mathbf{F}_I$ to denote the interpolation filter set in general, which in the following sections can be $\{\mathbf{F}_1, \mathbf{F}_2\}$ for two 1D separable filters or $\mathbf{F}$ for 2D non-separable filters. Next we will show how to jointly optimize the prediction coefficients $\mathbf{P}_{B_2}$ and the interpolation filters $\mathbf{F}_I$

together.

2.3.2 Joint Optimization of EBTDTP and Separable Filters

Our overall objective is to design $\{\mathbf{P}_{B_2}, \mathbf{F}_1, \mathbf{F}_2\}$ to minimize the mean squared prediction error (MSE) J in (2.7). Instead of solving this non-linear multi-variate optimization problem directly, we propose an iterative approach of optimizing one of $\{\mathbf{P}_{B_2}, \mathbf{F}_1, \mathbf{F}_2\}$, while fixing the other two. To simplify the cost expression, let us set

$$\mathbf{H}_1 = \mathbf{F}_1\mathbf{D}'_{B_2},\ \mathbf{H}_2 = \mathbf{D}_{B_2}\mathbf{F}_2, \tag{2.13}$$

$$\mathbf{X}_T = \mathbf{D}_{B_2}\mathbf{X}\mathbf{D}'_{B_2}, \tag{2.14}$$

so the cost becomes,

$$J = \|\mathbf{Y} - \mathbf{H}_1(\mathbf{X}_T \circ \mathbf{P}_{B_2})\mathbf{H}_2\|^2 . \tag{2.15}$$

Optimizing $\mathbf{P}_{B_2}$ given $\mathbf{F}_1, \mathbf{F}_2$ fixed

The cost J in (2.15) is proportional to

$$\begin{aligned} J \ \propto\ & \sum_{m=1}^{B_1}\sum_{n=1}^{B_1}\Big[\mathbf{Y}(m,n) - \\ & \sum_{i=1}^{B_2}\sum_{j=1}^{B_2}\mathbf{P}_{B_2}(i,j)\mathbf{X}_T(i,j)\mathbf{H}_1(m,i)\mathbf{H}_2(j,n)\Big]^2 \end{aligned} \tag{2.16}$$

This is equivalent to the least square estimation problem of minimizing $||\mathbf{A}\mathbf{p}_{B_2} - \mathbf{b}||^2$, where $\mathbf{p}_{B_2}$ is the vector form (of size $B_2^2 \times 1$) of $\mathbf{P}_{B_2}$, $\mathbf{A}$ and $\mathbf{b}$ (of size of $B_1^2 \times B_2^2$ and

$B_1^2 \times 1$) are quantities derived from the training data as,

$$\mathbf{A}(k,l) = \mathbf{X}_T(i,j)\mathbf{H}_1(m,i)\mathbf{H}_2(j,n), \tag{2.17}$$

$$\mathbf{b}(k) = \mathbf{Y}(m,n), \tag{2.18}$$

where, $k = mB_1 + n$ $(m,n = 0 \dots B_1 - 1)$, and $l = iB_2 + j$ $(i,j = 0 \dots B_2 - 1)$. The optimal solution for the prediction coefficients is given as,

$$\mathbf{p}_{B_2} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}, \tag{2.19}$$

$$\mathbf{P}_{B_2}(i,j) = \rho_{i,j} = \mathbf{p}_{B_2}(l). \tag{2.20}$$

Optimizing $\mathbf{F}_1$ or $\mathbf{F}_2$ given $\mathbf{P}_{B_2}$ fixed

From (2.13) we know, we can optimize $\mathbf{F}_1, \mathbf{F}_2$ by optimizing $\mathbf{H}_1, \mathbf{H}_2$. Since $\mathbf{H}_1$ and $\mathbf{H}_2$ are symmetric in (2.15), their optimization approaches are very similar.

First we assume $\mathbf{H}_1$ and $\mathbf{P}_{B_2}$ are fixed, then $\mathbf{H}_2$ only depends on $\mathbf{f}_2$, which reduces to the overall problem to linear optimization. That is, we convert J to $\|\mathbf{A}\mathbf{f}_2 - \mathbf{b}\|^2$, where,

$$\mathbf{A}(v,i) = \sum_{k=0}^{B_2-1}\sum_{l=0}^{B_2-1}\mathbf{D}_{B_2}(l, i+n+1)\mathbf{H}_1(m,k)\mathbf{X}_T(k,l)\mathbf{P}_{B_2}(k,l), \tag{2.21}$$

$$\mathbf{b}(v) = \mathbf{Y}(m,n), \tag{2.22}$$

$$v = mB_1 + n \ (m,n = 0 \dots B_1 - 1), i = 0 \dots b - 1. \tag{2.23}$$

The optimal solution for $\mathbf{f}_2$, given $\mathbf{f}_1$ and $\mathbf{P}_{B_2}$, is

$$\mathbf{f}_2 = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}, \tag{2.24}$$

and $\mathbf{H}_2$ can be derived from (2.9) and (2.13). Similarly, we can use the same formula as (2.24) to optimize $\mathbf{f}_1$ (or $\mathbf{F}_1$) while fixing $\mathbf{F}_2$ and $\mathbf{P}_{B_2}$, where (2.21) and (2.22) would change to,

$$\mathbf{A}(v,i) = \sum_{k=0}^{B_2-1}\sum_{l=0}^{B_2-1}\mathbf{D}_{B_2}(k, i+m+1)\mathbf{H}_2(l,n)\mathbf{X}_T(k,l)\mathbf{P}_{B_2}(k,l), \tag{2.25}$$

$$\mathbf{b}(v) = \mathbf{Y}(m,n). \tag{2.26}$$

We iteratively optimize $\mathbf{F}_1$, $\mathbf{F}_2$ and $\mathbf{P}_{B_2}$, in each step the prediction error J decreases until it converges.

2.3.3 Joint Optimization of EBTDTP and Non-separable Filters

In the previous sub-section, we presented the joint optimization for prediction coefficients and separable filters. As mentioned, we also investigate the performance of non-separable filters as an alternative for more complex local statistics. The prediction $\tilde{\mathbf{Y}}$ from (2.12) becomes

$$\tilde{\mathbf{Y}} = (\mathbf{D}'_{B_2}((\mathbf{D}_{B_2}\mathbf{X}\mathbf{D}'_{B_2}) \circ \mathbf{P}_{B_2})\mathbf{D}_{B_2}) * \mathbf{F} \tag{2.27}$$

$$= (\mathbf{D}'_{B_2}(\mathbf{X}_T \circ \mathbf{P}_{B_2})\mathbf{D}_{B_2}) * \mathbf{F} \tag{2.28}$$

where $*$ denotes the 2D convolution, which returns $\tilde{\mathbf{Y}}$ as the valid $B_1 \times B_1$ region at the center. The cost function in (2.7) becomes

$$J = \left\| \mathbf{Y} - (\mathbf{D}'_{B_2}(\mathbf{X}_T \circ \mathbf{P}_{B_2})\mathbf{D}_{B_2}) * \mathbf{F} \right\|^2. \tag{2.29}$$

We set $\mathbf{G} = \mathbf{D}'_{B_2}(\mathbf{X}_T \circ \mathbf{P}_{B_2})\mathbf{D}_{B_2}$, and p as the size of 2D $p \times p$ non-separable filter, then

$$\tilde{\mathbf{Y}} = \mathbf{G} * \mathbf{F}, \tag{2.30}$$

$$\tilde{\mathbf{Y}}(m,n) = \sum_{i=-p/2}^{p/2-1} \sum_{j=-p/2}^{p/2-1} \mathbf{F}(i+\frac{p}{2}, j+\frac{p}{2})\mathbf{G}(m+i+1, n+j+1), \tag{2.31}$$

$$\mathbf{G}(s,t) = \sum_{k=0}^{B_2-1} \sum_{l=0}^{B_2-1} \mathbf{D}_{B_2}(k,s)\mathbf{X}_T(k,l)\mathbf{P}_{B_2}(k,l)\mathbf{D}_{B_2}(l,t). \tag{2.32}$$

$$(m,n = 0...B_1 - 1,\ s,t = 0...B_2 - 1)$$

We again use an iterative approach to jointly optimize $\mathbf{F}$ and $\mathbf{P}_{B_2}$. Given $\mathbf{P}_{B_2}$, the optimal $\mathbf{F}$ would be the Wiener filter. Given $\mathbf{F}$, to estimate $\mathbf{P}_{B_2}$, we also convert it to a least square estimation problem of minimizing $\|\mathbf{A}\mathbf{p}_{B_2} - \mathbf{b}\|^2$, where $\mathbf{p}_{B_2}$ is the vector form (of size $B_2^2 \times 1$) of $\mathbf{P}_{B_2}$, with $\mathbf{A}$ and $\mathbf{b}$ as,

$$\begin{aligned} \mathbf{A}(v,u) &= \sum_{i=1}^{p}\sum_{j=1}^{p} \mathbf{F}(i,j)\mathbf{D}_{B_2}(k, m-\frac{p}{2}+i) \\ &\qquad \mathbf{X}_T(k,l)\mathbf{D}_{B_2}(l, n-\frac{p}{2}+j) \qquad (2.33) \\ \mathbf{b}(v) &= \mathbf{Y}(m,n) \qquad (2.34) \\ v &= mB_1 + n\ (m,n = 0...B_1 - 1) \qquad (2.35) \\ u &= kB_2 + l\ (k,l = 0...B_2 - 1) \qquad (2.36) \end{aligned}$$

The optimal solution for prediction coefficients $\mathbf{P}_{B_2}$ given 2D non-separable interpolation filter $\mathbf{F}$ is

$$\mathbf{p}_{B_2} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}, \tag{2.37}$$

$$\mathbf{P}_{B_2}(k,l) = \rho_{k,l} = \mathbf{p}_{B_2}(u). \tag{2.38}$$

We iteratively optimize $\mathbf{F}$ and $\mathbf{P}_{B_2}$, until the prediction error J converges.

2.4 TDTP Adaptation to HEVC Encoder Decisions

We generalize the TDTP framework to fully support HEVC, which has adopted various coding features to capture different local statistics in video sequences. Based on the rate-distortion trade-off, the encoder divides the video frames into different block sizes to capture the texture and temporal correlation. It chooses to interpolate the reference blocks at certain sub-pixel locations, which affects the temporal correlation in the frequency domain (from the discussion in Sec. I). The quality (QP) of the reference blocks also largely influences the correlation between the reference blocks and original blocks. Therefore, to capture the various temporal correlation model determined by the encoder decisions, we train different TDTP models in the $\{\mathbf{P}_{B_2}, \mathbf{F}_I\}$ set for portions of data that use the corresponding combinations of block sizes, sub-pixel locations and the reference block QP. Since those coding features are available at both encoder and decoder, it does not incur extra overhead or complexity.

HEVC follows the quad-tree partition structure then further decides the PU and TU sizes. We support TDTP models for blocks of size 4x4, 8x8, 16x16 and 32x32. TDTP for blocks of size 64x64 is not supported because of its high complexity and its rare occurrence in the practical encoding. To achieve the best spatial decorrelation via TDTP, we perform

TDTP on the largest square size possible for each PU (i.e., on the whole PU block if it is square or sub-divide it into multiple square pieces if it is not square). To account for the quarter-pel precision sub-pixel interpolation filter in HEVC, we have different TDTP models for all the 16 sub-pixel locations. Instead of training for each individual QP value (0-51) in HEVC, we quantize the QP range to a few groups (e.g.: 20-25, 26-30, 31-35, 36-40) and train TDTP models for each QP group. To better capture the variation in statistics, we also introduce 8 frame-level TDTP modes for the encoder to choose from, and will design those 8 modes properly in the next chapter. In summary, we have 8 frame-level TDTP modes, each of which contains 256 ($4 \times 16 \times 4$) models accounting for the various local statistics. All the prediction parameters are trained offline and stored in both encoder and decoder.

The overall TDTP framework for temporal prediction can be summarized as follows. The encoder follows the standard quad-tree partition and motion search to get the reference block (and its neighboring pixels). Depending on the PU size, the QP and sub-pixel location of the reference block, the encoder applies the corresponding prediction parameters and interpolation filter (as described in Fig. 2.5) to get the prediction block. Depending on the type of applications, the encoder may choose to use the 1D 8-tap separable or the 2D 4x4 non-separable filters with a sequence-level flag. It may also choose to use one of the 8 modes for each frame for the best RD performance, with a frame-level flag.

2.5 Conclusion

In this chapter, we introduced a comprehensive framework to perform temporal prediction in transform domain, where spatial correlation is removed and we can perform an optimal one-to-one temporal correlation on transform coefficients. We proposed the

EBTDTP to better exploit spatial correlations, jointly designed the EBTDTP with interpolations filters to avoid the coupled interference. We generalized the TDTP framework to fully support HEVC, and adapted the TDTP model to encoder decisions to better capture the local statistics.

In the next chapter, we will focus on the effective design for a set of TDTP models to cover the rich variety of statistics via offline training. Experimental results of the TDTP framework with the full design scheme will be presented at the end of the next chapter.

Chapter 3

Asymptotic Closed-loop Design for Predictive Coding System

In this chapter[1], we address a catastrophic yet prevalent instability problem in the offline design for closed-loop systems. We use the joint predictor model in Chapter 2 as an example and propose a two-loop Asymptotic Closed-loop (ACL) approach to address the instability problem. It is further combined with the K-modes clustering approach to optimize the mode design. Consistent coding gains of the TDTP model using this design framework are presented. We will also compare the performance of the joint predictor model trained with and without the ACL framework.

3.1 The Instability Problem in the Closed-loop System Design

Video coding system is a typical closed-loop system as the reconstruction of a frame will be used as reference for future frames. Usually in the offline training in a closed-loop

[1]This chapter is adapted from 978-1-4799-8339-1/15/$31.00 © 2015 IEEE.

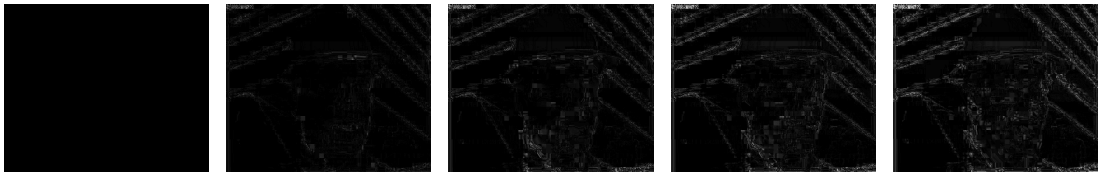
system, the reconstruction, a predictor is designed based on a series of reconstructed frames. However, when the designed predictor is applied in a closed-loop system, it changes the reconstruction, therefore the statistics become incompatible with those it is designed for. Such deviation in statistics potentially growing in magnitude as the coder advances through the sequence, as the quality of actual prediction impacts the quality of reconstruction and thereby the next frame's prediction, and so on. This problem is critical in low-bitrate encoding where the residual is not well encoded and the reconstruction is more reliant on the prediction.

To better formulate the problem, we take the joint predictor model in Chapter 2 as example. We denote a sequence of DCT coefficients at a certain frequency for blocks along the motion trajectory as, $x_1, x_2, \ldots, x_N$. The first frame is intra coded so the reconstruction of the DCT coefficients in the first frame $\hat{x}_1$ does not depend on other frames. As discussed earlier, (2.2) is also true in the DCT domain, $x_n = \tilde{x}_n + r_n$. In traditional motion compensation, $\tilde{x}_n = \hat{x}_{n-1}$, thus the reconstruction is

$$\hat{x}_n = \hat{x}_{n-1} + \hat{r}_n \tag{3.1}$$

where $\hat{r}_n$ is the reconstruction of residual r_n.

In the joint predictor framework, the prediction $\tilde{x}_n$ is defined as (2.12) or (2.28) (depending on the type of interpolation filters). We use $\{\mathbf{P}_{B_2}, \mathbf{F}_I\}(\hat{x}_{n-1})$ to denote the joint prediction for frame n, where the $\{\mathbf{P}_{B_2}, \mathbf{F}_I\}$ are the predictors designed based on the statistics of $\{\hat{x}_{n-1}, x_n\}$. On using this prediction coefficient in the coder, first reconstructed coefficient, $\hat{x}_1$ is unaltered as it is intra coded. For frame 2, we have a different reconstruction $\hat{x}'_2 = \{\mathbf{P}_{B_2}, \mathbf{F}_I\}(\hat{x}_1) + \hat{r}'_2$. This is then used to generate frame 3,



(a) Frame 1 (b) Frame 3 (c) Frame 5 (d) Frame 7 (e) Frame 9

Figure 3.1: The deviation between frames used in training and actual frames when applying the predictor in closed loop

$\hat{x}'_3 = \{\mathbf{P}_{B_2}, \mathbf{F}_I\}(\hat{x}'_2) + \hat{r}'_3$, so we have

$$\hat{x}'_n = \{\mathbf{P}_{B_2}, \mathbf{F}_I\}(\hat{x}'_{n-1}) + \hat{r}'_n \tag{3.2}$$

However, $\{\mathbf{P}_{B_2}, \mathbf{F}_I\}$ were designed for $\{\hat{x}_{n-1}, x_n\}$, which is different from the real statistics $\{\hat{x}'_{n-1}, x_n\}$ here. In Fig. 3.1, we show the difference between the frames used in training $\hat{x}_n$ and the real statistics $\hat{x}'_n$ when applying the predictor in closed loop at $n = 1, 3, 5, 7, 9$. The deviation builds up as we proceed to future frames. In low-bitrate encoding, this deviation is especially catastrophic, because the residual is not well encoded and the reconstruction is more dependent of the prediction.

This instability in offline design for closed-loop systems is very common and it also applies to the design of other types of predictors or codec components. In the intra predictor design in Chapter 4, the predictor is trained based on the reconstruction of the boundary pixels, which is dependent on the predictor itself, thus the change in reconstruction propagates spatially. In Chapter 6, we design KLTs for different types of residual statistics. Similarly, the residual depends on the reconstruction, which in return depends on the designed KLT. In the rest of this chapter, we will show a general way to solve this instability problem and apply it to the design of different codec components.

3.2 A Two-loop Asymptotic Closed-loop Approach

Inspired by the early asymptotic closed-loop (ACL) work in [36], we propose a two-loop ACL design which accounts for the encoder decisions to solve this instability problem effectively. The basic idea of ACL is to employ open-loop prediction to avoid the instability problem, while updating the prediction parameters in each iteration. Once the parameters converge, it becomes equivalent to the closed-loop operation. Here again, we use the joint predictor as an example to explain the two-loop ACL approach.

We use double subscripts, e.g., $x_{n,t}$ to indicate variables from frame n and iteration t. Given a set of reconstructed coefficients along a motion trajectory for a frequency at iteration $t-1$, $\hat{x}_{1,t-1}, \hat{x}_{2,t-1}, \ldots, \hat{x}_{N,t-1}$, we estimate the prediction parameters for iteration t as $\{\mathbf{P}_{B_2,t}, \mathbf{F}_{I,t}\}$, which is based on the statistics of $\{\hat{x}_{n-1,t-1}, x_n\}$. This prediction parameter set is then employed in open-loop to predict coefficients in frame n of iteration t, $\tilde{x}_{n,t} = \{\mathbf{P}_{B_2,t}, \mathbf{F}_{I,t}\}(\hat{x}_{n-1,t-1})$. So instead of (3.1) and (3.2) where all the samples are of the same iteration, the reconstruction in this open-loop scheme is given as,

$$\hat{x}_{n,t} = \{\mathbf{P}_{B_2,t}, \mathbf{F}_{I,t}\}(\hat{x}_{n-1,t-1}) + \hat{r}_{n,t} \tag{3.3}$$

where $r_{n,t}$ is the residual of frame n and iteration t, $r_{n,t} = x_n - \tilde{x}_{n,t}$, and $\hat{r}_{n,t}$ is its reconstruction. Different from the closed-loop system, the change in reconstruction in the iteration t does not propagate to the future frames, since each frame is referenced from the previous iteration $t-1$, as shown in Fig. 3.2. Therefore, there is no instability problem in this open-loop system.

Next, we will show that this series of open-loop processes will converge to the target closed-loop system asymptotically. Consider the statistics from the first iteration $\{\hat{x}_{n-1,1}, x_n\}$, we design $\{\mathbf{P}_{B_2,2}, \mathbf{F}_{I,2}\}$ and run the second iteration. Since $\{\mathbf{P}_{B_2,2}, \mathbf{F}_{I,2}\}$ is

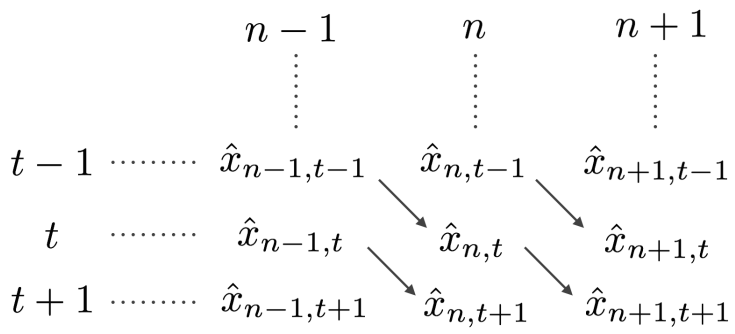


Figure 3.2: Asymptotic Closed-Loop (ACL) training approach

specifically designed to optimize the prediction error for $\{\hat{x}_{n-1,1}, x_n\}$, the prediction $\tilde{x}_{n,2}$ is guaranteed to improve. Better prediction often leads to better reconstruction, thus $\hat{x}_{n,2}$ is generally better than $\hat{x}_{n,1}$. In the third iteration, each frame is referencing from a better reconstruction, which generally leads to better prediction. So the $\tilde{x}_{n,3}$ is generally better than $\tilde{x}_{n,2}$, and following the same logic, $\hat{x}_{n,3}$ is generally better than $\hat{x}_{n,2}$. The reconstruction error is generally decreasing and would approach convergence. On convergence, the reconstruction error remains the same, i.e., $\hat{x}_{n-1,t-1} = \hat{x}_{n-1,t}$, which makes it equivalent to the closed-loop system, i.e.,

$$\hat{x}_{n,t} = \{\mathbf{P}_{B_2,t}, \mathbf{F}_{I,t}\}(\hat{x}_{n-1,t}) + \hat{r}_{n,t} \tag{3.4}$$

and the prediction coefficients converge as well, i.e., $\{\mathbf{P}_{B_2,t}, \mathbf{F}_{I,t}\} = \{\mathbf{P}_{B_2,t-1}, \mathbf{F}_{I,t-1}\}$.

The basic ACL idea was proposed in [36] to optimize vector quantizer design in differential pulse-code modulation (DPCM) system. The modern video codec, however, is much more complicated. First, as mentioned in Section 1, the encoder would choose the best coding decisions (block sizes, motion vectors, prediction modes, QP, etc.) to optimize the RD cost. The encoder decisions are dependent on the prediction parameters, while the prediction parameters are also dependent on the encoder decisions. Second, the joint predictors are designed to minimize prediction error, which mismatches with the

ultimate metric, RD cost. Here, we propose a two-loop design scheme. In the inner loop, we estimate prediction parameters $\{\mathbf{P}_{B_2}, \mathbf{F}_I\}$ via ACL while fixing the encoder decisions. In the outer loop, we update the encoder decisions while fixing the prediction parameters designed from the inner loops.

The basic assumption for convergence that better prediction and better reconstruction are mutually supportive is not always guaranteed. For real world sequences we observe that the prediction error cost decreases initially and then hits a limit cycle. Thus we simply stop the inner loop iterations when this cost stops decreasing. Moreover, the mismatch of the optimization criteria in the two loops also does not ensure full convergence in the outer loop. In practical implementation, we notice that after 5-8 iterations in the outer loop, the performance stops improving significantly and hits a limit cycle. Thus in our experiments, we stop the outer loop after 8 iterations and pick the predictors with the minimum RD cost.

3.3 Mode Design with the Two-loop ACL

It is well known that rich video contents contain various motion patterns (scaling, rotation, translation, etc.), leading to a variety of temporal correlation pattern in transform domain, which is impossible to be described in a single joint predictor model. To accommodate the variety in statistics of natural video sequences, we introduce multiple modes with different prediction parameters for encoder to choose from. In our experiment, the encoder will choose one mode per frame and write the mode decision in the bitstream with a negligible overhead.

This specific mode design problem can be viewed as a hard clustering problem. Among the typical clustering algorithms, K-means is known to be the most efficient in terms of execution time and good for large dataset [43]. A variation of K-means, named K-modes,

extends the approach to non-numerical data points. We can define the distance as the prediction error using the prediction parameters in a certain mode j, according to (2.15) and (2.29), based on the type of the interpolation filters, we have

$$J_j = \|\mathbf{Y} - \mathbf{H}_{1,j}(\mathbf{X}_T \circ \mathbf{P}_{B_2,j})\mathbf{H}_{2,j}\|^2 . \tag{3.5}$$

or

$$J_j = \left\|\mathbf{Y} - (\mathbf{D}'_{B_2}(\mathbf{X}_T \circ \mathbf{P}_{B_2,j})\mathbf{D}_{B_2}) * \mathbf{F}_j\right\|^2 . \tag{3.6}$$

Following the two-loop ACL approach, where we update encoder decision in the outer loop to minimize the RD cost and update prediction parameters in the inner loop to minimize the prediction error, we update a set of K modes of prediction parameters in the inner loop. To initialize the K modes, we randomly select some sequences and train one mode for each sequence, and use those per-sequence-trained modes as the initial modes. In each ACL iteration, it runs the two-step iterations:

- Re-design the prediction parameters for each mode based on the approaches in Sec. III.
- Re-cluster all the block pairs in the training set to the best mode that yields the minimum prediction error.

At each step, the prediction error decreases monotonically through the process until convergence. We incorporate this K-modes training step to the innermost core of the two-loop ACL training. After ACL converges, we have the optimal K modes designed for the closed loop process given the fixed encoder decisions. Then we update the encoder decisions in the outer loop. The whole training algorithm is shown in Algorithm 1.

Algorithm 1: Joint design algorithm for EBTDTP and interpolation filters

Input : A set of training sequences, the number of TDTP modes K
Output: The prediction coefficients set $\{\mathbf{P}_{B_2}, \mathbf{F}_I\}$ of size K

Initialize $\{\mathbf{P}_{B_2}, \mathbf{F}_I\}$;
Run the codec in closed-loop, store the encoder decision d, mode assignment a and reconstructed video file rec;
while $iter < max\_iter$ **do**
  while *MSE decreases* **do**
    Run the codec in open-loop using the same encoder decision d, but with rec as the motion compensation reference;
    Extract the reference and original blocks;
    while *MSE decreases* **do**
      Optimize $\mathbf{P}_{B_2}$ and $\mathbf{F}_I$ for each cluster;
      Assign the block pairs into the best mode to minimize MSE;
      Update a, update MSE;
    end
    Run the codec in open-loop again, using the same d, rec, and the optimized a, $\{\mathbf{P}_{B_2}\}$ and $\{\mathbf{F}_I\}$; Update rec as the newly reconstructed video file; Update MSE;
  end
  Run the codec in closed-loop, update the encoder decision d and reconstructed video file rec, store the RD-cost into array $\{cost\}$. $iter = iter + 1$;
end
Find the minimum RD cost in $\{cost\}$, set the corresponding $\{\mathbf{P}_{B_2}\}$ and $\{\mathbf{F}_I\}$ as the final trained predictors;

3.4 Experimental Results of the TDTP framework designed via ACL approach

3.4.1 Results for Off-line Encoding Applications

We first examine the full potential of the TDTP framework by designing a specific set of coefficients for each sequence using the off-line training method described above. This is targeting video storage applications where encoding is performed off-line (e.g. Youtube, Netflix, cloud video storage, etc.) so that designing for each individual sequence is possible. The proposed framework is implemented in HM 14.0, under the lowdelay P configuration with all default features enabled. Each sequence is tested at various bitrates with QP ranging from 22 to 37. We compare the coding gain of different approaches over the baseline HEVC in Table 3.1. The average bitrate reduction over HEVC for TDTP, EBTDTP, jointly optimized EBTDTP with separable and non-separable filters are 3.18%, 3.76%, 4.50%, 6.96%. As we can see, some sequences benefit more from the separable filters while others benefit more from the non-separable filters. For video storage applications, since the encoding is only performed once and the bitstreams are stored, we can introduce a separable/non-separable filter switch per sequence at the encoder for the best RD performance (EBTDTP + Interpl) with an average coding gain of 7.14%.

To show the importance of ACL, we also run the training algorithm without ACL in the same settings. The RD curves of sequence *Stefan* are shown in Fig. 3.3 with performance comparison to employing prediction coefficients trained without the ACL technique. As discussed earlier, the instability problem is significantly critical in low-bitrate encoding where the reconstruction relies more on the prediction. Therefore, training without ACL suffers more at low bitrate cases, resulting in worse performance than the

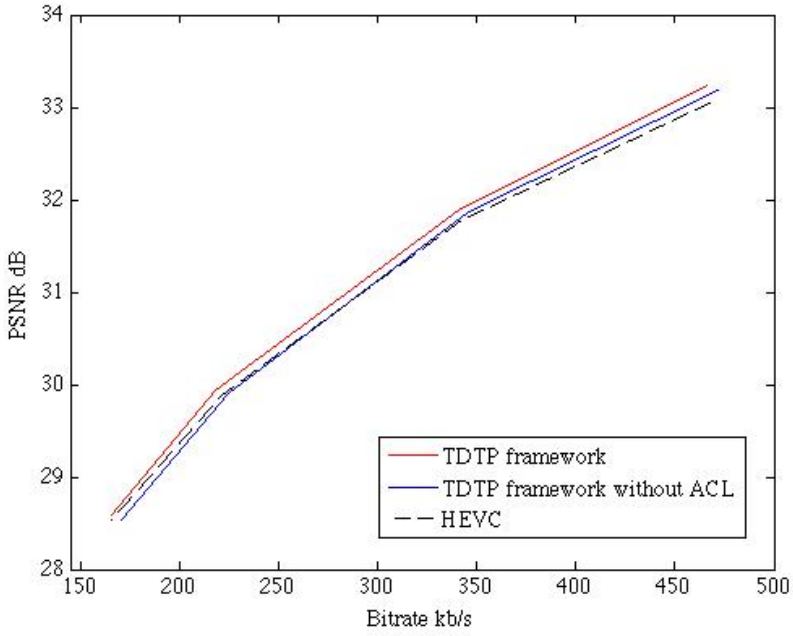


Figure 3.3: Coding performance comparison for sequence *Stefan* at *CIF* resolution

standard HEVC.

3.4.2 Results for Live Communication Applications

For the other type of applications where the encoding time is limited (e.g.: live video, real-time communication, etc.), per-sequence training is impractical. Here we provide a choice of fixed 8 sets of prediction parameters for the encoder to choose at each frame, with a negligible overhead of 3 bits per frame. In the preliminary experiments from our earlier work [36, 34, 35], we simply chose the 8 most distinct sets of prediction parameters from the training set (referred to as *the naive approach*), while in this work we optimize those modes by combining the K-modes training approach with the ACL framework (referred to as *the K-modes approach*). We noticed that the non-separable filters generally outperform the separable filters, so introducing a separable/non-separable filter switch at the encoder would help little but double the complexity. Therefore, for this type of

Sequences	TDTP	EBTDTP	EBTDTP +sep	EBTDTP +nonsep	EBTDTP +interpl
Coastguard (CIF)	9.47	9.45	9.90	10.71	**10.71**
Bridge-far (CIF)	4.83	5.75	7.66	5.25	**7.66**
Mobile (CIF)	2.55	3.53	3.83	7.90	**7.90**
Highway (CIF)	2.46	3.31	4.29	8.64	**8.64**
Stefan (CIF)	2.41	3.69	4.20	6.00	**6.00**
BlowingBubbles (240p)	0.33	0.09	0.10	0.76	**0.76**
BQMall (480p)	1.57	1.41	2.76	5.47	**5.47**
PartyScene (480p)	1.06	1.43	4.38	3.92	**4.38**
Keiba (480p)	2.52	2.16	2.16	5.35	**5.35**
Parkrun_ter (720p)	4.46	4.90	4.97	5.85	**5.85**
Mobcal_ter (720p)	2.41	3.68	9.37	18.23	**18.23**
Shields_ter (720p)	6.57	8.70	6.21	11.59	**11.59**
ParkScene (1080p)	0.65	0.74	0.78	1.44	**1.44**
BQTerrace (1080p)	5.78	7.19	7.70	13.29	**13.29**
Pedestrain_area (1080p)	2.80	2.96	2.75	4.66	**4.66**
Tennis (1080p)	0.94	1.12	0.97	2.35	**2.35**
Average	**3.18**	**3.76**	**4.50**	**6.96**	**7.14**

Table 3.1: Comparison of reduction in bitrate over HEVC for training set

encoding time sensitive applications, we only use the EBTDTP+nonsep approach.

To make sure the statistics of training set can represent the statistics of test set, we set the training set as a collection of short clips (referred to as the *Training Clip*) of different sequences, and set the test set as a collection of another short clips (referred to as the *Testing Clip*). We examine the efficacy of K-modes training by comparing the gain in the testing clips by using modes generated in *the naive approach* and in *the K-modes approach*, and present them in Table 3.2. By using *the K-modes approach*, the gain of jointly designed EBTDTP and non-separable filters increases from 2.91% to 4.28%.

3.4.3 Complexity Analysis and Future Work

Although we have jointly designed the EBTDTP and interpolation filters, tackled the instability problem and optimized the mode design, we mainly focus on optimizing

Sequences	Naive approach	K-modes approach
Football (CIF)	4.81	6.41
News (CIF)	0.82	1.47
Silent (CIF)	0.05	0.19
Bridge-close (CIF)	0.35	1.06
BasketballDrive (1080p)	3.68	4.99
Kimono (1080p)	4.15	4.68
Cactus (1080p)	1.27	2.81
Station (1080p)	0.17	0.38
Sunflower (1080p)	9.96	13.86
Tractor (1080p)	0.56	3.12
Ducks take off (1080p)	6.19	8.08
Average	**2.91**	**4.28**

Table 3.2: Reduction in bitrate over HEVC for test set

the prediction error rather than the rate-distortion cost (which explains why the joint approach is not guaranteed to be better than the others). How to approximate the global optimum in terms of rate-distortion cost would be an interesting topic for future research.

Compared to the standard HEVC, the TDTP framework has an additional pair of DCT and inverse DCT, and the extra multiplication during the scaling of transform coefficients. Under our preliminary implementation in HM 14.0, adding those extra operations doubles the encoding and decoding time. Note that fast implementation and hardware acceleration for DCT have been well-studied, leaving a lot of room for speed optimization. If we have K sets of prediction parameters for the encoder to choose from for each frame, the encoding time would increase by K times. Helping encoder make fast mode decisions adaptively would be another part of the future work.

3.5 Conclusion

In this chapter, we introduced a two-loop ACL approach to circumvent the challenge of catastrophic design instability due to quantization error propagation through the pre-

diction loop. It is further combined with the K-modes clustering approach for effective mode design. We show the compression gain of using the TDTP model designed using the proposed training framework for different applications, with an average of 4.3% reduction in bitrate over the state-of-the-art HEVC codec.

Chapter 4

Recursive Extrapolation Filter based Spatial Prediction

The conventional "pixel copying" intra prediction used in current video standards is sub-optimal because it ignores the varying non-separable spatial correlation. In this chapter[1], a recursive extrapolation filter based intra prediction is proposed, which captures the decaying non-separable correlation based on a 2D non-separable Markov model. The proposed system employs four-tap recursive extrapolation filters that can predict from all standard directions. Those filter coefficients are designed to adapt to relevant local information such block sizes and target bitrate. We design the filter to account for the overall rate-distortion cost in conjunction with the codec decisions, and combine it with the ACL design approach to avoid the instability problem in the offline design for closed-loop systems. Experimental evidence is provided for substantial coding gains over conventional intra prediction. The proposed approach has been adopted by the latest AV1 codec.

[1]This chapter is adapted from 978-1-4799-5751-4/14/$31.00 © 2014 IEEE.

4.1 Introduction

Intra prediction is critical for image and video coders to exploit spatial correlations within a image/frame. In current block based video coders [1, 2], reconstructed boundary pixels (or their linear combination) are copied along a specific direction to generate prediction for the current block, wherein the directionality depends on local texture. This technique assumes a separable Markov model with correlation coefficient of 1 along the selected direction and 0 along the perpendicular direction. This simplistic approach is sub-optimal because of the following two reasons. First, the spatial correlation in natural videos is not perfectly separable. Second, the correlation usually varies within a blocks, for example, the correlation with boundary pixels usually decay with distance.

Many approaches have been proposed to overcome these limitations. Predicting every pixel as a linear combination of all the boundary pixels was proposed in [44], and predicting blocks using weighted average of multiple decoded blocks was proposed in [45]. However, both these approaches suffer from considerable increase in computational complexity. Recursive extrapolation based on a separable Markov model was proposed in [46], however, this approach neglects the possibility that correlation is not perfectly separable. An early version of this work was published in [47], where a non-separable Markov model based recursive extrapolation approach with three-tap filters was proposed. However, challenges arise in the following aspects. First, the three-tap filters cannot capture prediction modes with top-right and bottom-left directions. Second, to improve the overall performance, the intra predictor needs to be designed to optimize the ultimate RD cost. Last, the design of the intra predictor also suffers from the instability problem similar to Chapter 3, as when we apply the designed predictor, the change in reconstruction would propagate spatially and lead to mismatched statistics.

In this chapter, we propose a stable, RD optimized, adaptive four-tap recursive filter

based intra prediction. The four-tap filter can cover all the intra prediction directions, with the fourth tap points at either the top-right pixel or the bottom-left pixel, depending on the directions. A two-phase filter optimization is proposed, with optimization on prediction error in the first phase, and optimization on RD cost in the second phase. The gradient descent approach is used in both phases with different optimization criteria. The filter design is also incorporated into the ACL framework to avoid the instability problem due to quantization error propagation. As the encoder decisions might imply some certain pattern of the statistics, we make the filters adaptive to local information of target bitrate and block size. We demonstrate the efficacy of the approach by implementing it within the VP9 framework. Evaluation results provide evidence of substantial coding gains over conventional intra coding.

4.2 Recursive Extrapolation Filtering

The basic building block of this proposal is the non-separable Markov model based recursive extrapolation filter to tackle the underutilization of available boundary information in the conventional "pixel copying" based intra prediction. In [47], the image signals were modeled by a three-tap Markov process with evolution recursion given as:

$$X = c_v V + c_h H + c_d D + \epsilon \tag{4.1}$$

This three tap structure has a critical limitation of not being able to represent directions arising from top-right and bottom-left. Hence, to cover all directions we add a fourth tap of, top-right pixel with left-to-right prediction scan order, or, bottom-left pixel with top-to-bottom prediction scan order.This modification requires previously reconstructed boundary pixels to be available at top-right of the current block, or at bottom-left of the

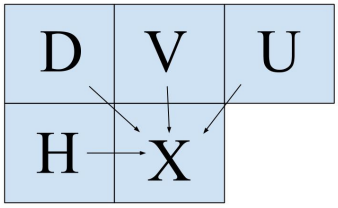
current block. While the availability of the top-right reconstructed boundary pixels is almost always ensured, newly introduced coding tools employing smaller prediction blocks (or prediction units, PU) within large coding blocks (or coding units, CU), also provide bottom-left reconstructed boundary pixels for some blocks. In case these additional boundary pixels are unavailable, the existing boundary pixel is copied to the unavailable pixel positions. The extended Markov process with four taps for directions originating from top-right is given as:

$$X = c_v V + c_h H + c_d D + c_u U + \epsilon \tag{4.2}$$

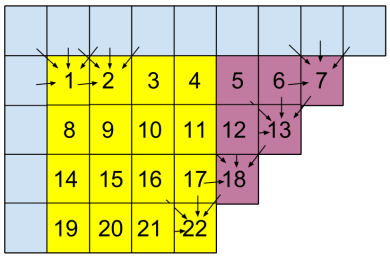
where V, H, D, U are neighboring pixels of X, as illustrated in Fig. 4.1(a). The four coefficients, c_v, c_h, c_d, and c_u, together capture the texture directionality. For medium to high bitrates, the reference pixels can be approximated by their reconstructions, to obtain the optimal predictor for X as

$$\tilde{X} = c_v \hat{V} + c_h \hat{H} + c_d \hat{D} + c_u \hat{U}. \tag{4.3}$$

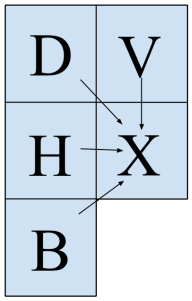
The recursive prediction order is enumerated from 1 to 22 in Fig. 4.1(b) and (d) for a 4×4 block. For prediction directions originating from the top-right, as illustrated in Fig. 4.1(a) and (b), the fourth tap locates to the top-right of the pixel; for prediction directions originating from the bottom-left, as illustrated in Fig. 4.1(c) and (d), the fourth tap locates to the bottom-left of the pixel, which is considered as the transposed version of the Markov process. A temporary triangular region is estimated to provide auxiliary reference to the right or bottom boundary pixels of the block. Without loss of generality, the discussion hereafter focuses mainly on the four-tap filters for directions originating from the top-right.



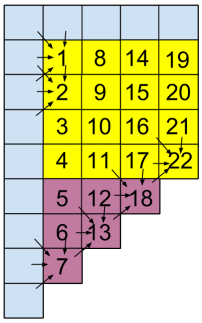
(a) 2-D Markov Process
for top-right directions



(b) Prediction order
for top-right directions



(c) 2-D Markov Process
for bottom-left directions



(d) Prediction order
for bottom-left directions

Figure 4.1: Four-tap recursive extrapolation filter

4.3 Filter Design

In this section we propose an offline filter design framework towards the ultimate goal of minimizing the rate-distortion cost in the actual compression for test sequences. We break it down into two phases, prediction error optimization and rate-distortion cost optimization. Gradient descent is applied at each phase, with different criteria and target function for the gradient calculation. This two-phase method is then combined into the ACL framework along with the K-modes clustering approach, to account for the flexibility in switching among modes and the quantization error propagation through the prediction loop.

4.3.1 Phase 1: optimization on prediction error

The overall prediction error to be minimized for the initial filter design is,

$$J = \sum_{k} J_k = \sum_{k} \sum_{\substack{\forall \text{ blocks} \\ \text{in Mode } k}} \sum_{i,j} (\tilde{x}_{i,j} - x_{i,j})^2. \tag{4.4}$$

To start the design, all the blocks are first classified into the original "pixel copying" modes based on the prediction error, then filters for each of these block subsets are initialized with the optimal linear predictor coefficients calculated as below, using the open-loop statistics of the 2-D non-separable Markov model of (4.2),

$$\begin{bmatrix} c_v \\ c_h \\ c_d \\ c_u \end{bmatrix} = \begin{bmatrix} R_{VV} R_{VH} R_{VD} R_{VU} \\ R_{VH} R_{HH} R_{HD} R_{HU} \\ R_{VD} R_{HD} R_{DD} R_{DU} \\ R_{VU} R_{HU} R_{DU} R_{UU} \end{bmatrix}^{-1} \begin{bmatrix} R_{XV} \\ R_{XH} \\ R_{XD} \\ R_{XU} \end{bmatrix}, \tag{4.5}$$

where R_{XV} denotes the cross correlation between pixel X and its upper pixel V, and so forth. Since the cross correlation is calculated using the original values rather than the predicted values in the actual recursive operation, a gradient descent approach is then employed to properly re-estimate filter coefficients for each cluster, with the gradient analytically calculated by taking a partial derivative of the per-mode cost J_k with respect to each of its coefficients, similar to [48]. For example, partial derivative with respect to $c_{v,k}$ is,

$$\frac{\partial J_k}{\partial c_{v,k}} = \sum_{\substack{\forall \text{ blocks} \\ \text{in Mode } k}} \sum_{i,j} 2(\tilde{x}_{i,j} - x_{i,j}) \frac{\partial \tilde{x}_{i,j}}{\partial c_{v,k}} \tag{4.6}$$

where $\frac{\partial \tilde{x}_{i,j}}{\partial c_{v,k}}$, derived using (4.3), has the following recursive relationship,

$$\begin{aligned} \frac{\partial \tilde{x}_{i,j}}{\partial c_{v,k}} = \tilde{x}_{i-1,j} + c_{v,k} \frac{\partial \tilde{x}_{i-1,j}}{\partial c_{v,k}} + c_{h,k} \frac{\partial \tilde{x}_{i,j-1}}{\partial c_{v,k}} \\ + c_{d,k} \frac{\partial \tilde{x}_{i-1,j-1}}{\partial c_{v,k}} + c_{u,k} \frac{\partial \tilde{x}_{i-1,j+1}}{\partial c_{v,k}}. \end{aligned} \tag{4.7}$$

Other partial derivatives can be derived similarly to calculate the required gradient.

4.3.2 Phase 2: optimization on RD cost

Designing filter coefficients to minimize prediction error is mismatched with the ultimate RD cost that coders optimize. In other words, the resulting prediction error reduction will not necessarily or fully translate into RD performance improvement. Therefore, we propose recursive extrapolation filters that are designed by direct optimization of the RD cost.

In the RD optimization phase, we use the gradient descent approach again with a new optimization criteria, RD cost. The RD cost [49, 50] is defined based on the Lagrange cost function as

$$RDcost = D + \lambda R \tag{4.8}$$

where the λ is the Lagrange multiplier which can be expressed as a function of quantization parameter (QP)

$$\lambda = 0.85 \times 2^{(QP-12)/3} \tag{4.9}$$

QP is determined by the encoder settings or chosen adaptively according to the target bitrate requirement. Here we treat the codec as a black box with the filter coefficients as input, and the RD cost as the output. We then can use a similar gradient descent approach where the RD cost is reduced at each step until we reach a local optima. The partial derivatives required for the gradient are calculated empirically, e.g., partial derivative with respect to the vertical coefficient of the kth mode is calculated as,

$$\frac{\partial RDcost(c_{v,k})}{\partial c_{v,k}} = \frac{RDcost(c_{v,k} + \Delta) - RDcost(c_{v,k} - \Delta)}{2\Delta}. \tag{4.10}$$

4.3.3 ACL based filter mode design

Similar to the traditional intra prediction which has multiple intra modes predicting from various directions, we also introduce multiple modes for the extrapolation filter based intra prediction (referred to as "filter-intra modes"), with different filter coefficients. In our experiments in the VP9 codec, to provide the encoder with more flexibility, we retain the 10 original intra prediction modes in addition to 10 designed filter-intra modes. From (4.3), we know that under special conditions of the filter coefficients, the four-tap filter model reduces to the original "pixel-copying" intra modes (e.g.: when $c_v = 1, c_h = c_d = c_u = 0$ is equivalent to the vertical intra prediction mode). Therefore, we initialize the 10 filter-intra modes by setting the filter coefficients as ones in those special cases corresponding to the original intra modes.

After initialization, we use a K-modes clustering approach by employing the following two steps iteratively: redesigning the filter coefficients using the two-phase optimization

approach and reclustering the blocks into the modes that yield the minimum RD cost defined in (4.8). To reach a convergence, it should be guaranteed that the RD cost is reduced at each step. Note that there is slight chance that the prediction error optimized coefficients might increase the RD cost due to the mismatch in the optimization criteria. Therefore, to make sure the RD cost is reduced monotonically, after phase one (prediction error optimization), we test the new coefficients and retrace back if the new ones lead to a worse RD cost.

We have presented the approach to design the RD optimal extrapolation filters given the original blocks and its top and left reconstructed boundaries. However, similar to Chapter 3, an instability problem also occurs in this intra prediction scenario, where the intra predictors depend on the reconstructed boundaries, and the reconstructed boundaries depend on the predictor itself. The reconstructed boundaries change as the new intra predictors are applied, so does the statistics. If the whole frame is an intra frame, this change in statistics would propagate through the whole frame. Therefore, to avoid this instability in the filter design, we incorporate the ACL and mode design framework into the filter design.

In the outer loop, we run the codec in closed-loop (i.e., the reconstructed boundaries are used to predict new blocks) at constant bitrate and store the encoder decisions (including block sizes, skip flags, etc.). While fixing the encoder decisions in the inner loop, we run the codec in open-loop (i.e., the reconstructed boundaries in the previous iteration are used to predict new blocks), and collect blocks of different sizes and modes and their corresponding boundaries. Those extracted blocks and their boundaries are then used to design the RD optimal filter coefficients using the K-modes approach. The new filter coefficients are applied to generate a new reconstruction with a smaller RD cost. Since the codec is running at the constant bitrate mode, a smaller RD cost is equivalent to smaller distortion in the reconstruction. This new and better reconstruction is then used

as reference in the new iteration of open-loop prediction, until the open-loop operations converge to the closed-loop operation. On convergence, we have the optimized filter coefficients given the fixed encoder decisions, when we go back to the outer loop to update the encoder decisions based on the RD cost.

We also make the filter coefficients adaptive to different encoder decisions (block sizes and target bitrate): blocks of different size may have different decaying patterns in spatial correlations and target bitrate determines the Lagrange multiplier λ in the rate-distortion tradeoff (4.8), which should be accounted for in the filter design. We design different sets of filter coefficients for block sizes of 4×4, 8×8, 16×16 and 32×32, and choose the filter set based on the encoder decision on block sizes. We also normalize the target bitrate that the encoder performs at to the spatial resolution of the sequence, and design filter coefficients for different normalized target bitrates. All the possible normalized target bitrates are divided into a few regions, and the filter coefficients for each region are designed by having the encoder perform at the centroid bitrate of the given region. As normalized target bitrate information is not available to the decoder, we add that as an additional side information per sequence.

We denote the filter coefficients set as

$$\mathbf{c}_{m,n} = \{c_{v,m,n}, c_{h,m,n}, c_{d,m,n}, c_{u,m,n}\} \quad (m = 1...4, n = 1...10) \tag{4.11}$$

where m corresponds to the index of block size and n corresponds to the index of filter-intra mode. The overall design algorithm is shown in Algorithm 2.

Algorithm 2: ACL based filter-intra mode design

Input : A set of training sequences, the number of intra modes K, the normalized target bitrate R_n

Output: The optimized intra filter coefficients set $\{\mathbf{c}_{m,n}\}$

Initialize the filter set $\{\mathbf{c}_{m,n}\}$;
Calculate the target bitrate for the training sequences resolution,
$R = R_n \times width \times height$;
Run the codec in closed-loop at the target bitrate R, store the encoder decision d, mode assignment a, and reconstructed video file rec;
while $iter < max\_iter$ **do**
  while *RDcost decreases* **do**
    Run the codec in open-loop using the same encoder decision d, but with rec as the motion compensation reference;
    Extract the original pixel values of blocks of different sizes and modes, and their reconstructed boundaries;
    while *RDcost decreases* **do**
      Redesign the filter coefficients;
      $\{\mathbf{c}_{m,n,old}\}$=$\{\mathbf{c}_{m,n}\}$;
      Run the codec in open-loop, using the same d, a, rec, and the $\{\mathbf{c}_{m,n,old}\}$, get the RD cost $RDcost_{old}$;
      For each mode and block size, optimize $\{\mathbf{c}_{m,n}\}$ in terms of prediction error;
      Run the codec in open-loop, using the same d, a, rec, and the new $\{\mathbf{c}_{m,n}\}$, get the RD cost $RDcost_{new}$;
      if $RDcost_{new} > RDcost_{old}$ **then**
        $\{\mathbf{c}_{m,n}\} = \{\mathbf{c}_{m,n,old}\}$
      end
      For each mode and block size, optimize $\{\mathbf{c}_{m,n}\}$ in terms of RD cost;
      Recluster the blocks;
      Run the codec in open-loop, using the same d, rec, and the new $\{\mathbf{c}_{m,n}\}$, update mode assignment a to minimize $RDcost$;
    end
    Run the codec in open-loop again, using the same d, rec, and new $\{\mathbf{c}_{m,n}\}$;
    Update rec as the newly reconstructed video file; Update $RDcost$;
  end
  Run the codec in closed-loop, update the encoder decision d and reconstructed video file rec, store the $RDcost$ into array $\{RDcost\}$. $iter = iter + 1$;
end
Find the minimum $RDcost$ in $\{RDcost\}$, set the corresponding $\{\mathbf{c}_{m,n}\}$ as the final filter-intra modes;

Table 4.1: Bitrate reduction using the 4-tap intra extrapolation filter designed with and without ACL (low resolution)

Low-res	Filterintra	Filterintra+ACL
Tempete (CIF)	3.61	3.88
Flower (CIF)	1.82	1.97
Mobile (CIF)	1.97	1.81
Coastguard (CIF)	2.54	2.59
Bus (CIF)	3.33	3.29
BasketballPass (240p)	3.42	3.86
BlowingBubbles (240p)	2.22	2.36
BQSquare (240p)	1.11	0.99
Average (Low-res)	**2.50**	**2.59**

Table 4.2: Bitrate reduction using the 4-tap intra extrapolation filter designed with and without ACL (middle resolution)

Mid-res	Filterintra	Filterintra+ACL
BQMall (480p)	3.24	3.43
Keiba (480p)	4.29	4.11
Flowervase (480p)	3.14	3.24
PartyScene (480p)	1.69	1.62
Racehorse (480p)	2.31	2.34
FourPeople (720p)	4.48	4.61
Johnny (720p)	3.39	3.89
KristenAndSara (720p)	3.50	3.61
Average (Mid-res)	**3.25**	**3.36**

Table 4.3: Bitrate reduction using the 4-tap intra extrapolation filter designed with and without ACL (high resolution)

High-res	Filterintra	Filterintra+ACL
BQTerrace (1080p)	1.24	1.51
BasketBallDrive (1080p)	3.85	4.39
Cactus (1080p)	3.76	3.98
Kimono (1080p)	3.88	3.88
Tennis (1080p)	2.03	2.31
Pedestrain_Area (1080p)	4.03	4.42
Sunflower (1080p)	4.28	4.43
Tractor (1080p)	3.47	3.77
Average (High-res)	**3.32**	**3.59**

4.4 Experimental Results

We implement the proposed technique within the VP9 framework to demonstrate its efficacy. 10 filter-intra modes are added in addition to the 10 original intra modes. For each original intra mode, the corresponding filter-intra mode being active is indicated with one additional bit, which is encoded using the arithmetic coding framework of VP9, and the probability table required is estimated using the training data. Note that in all our experiments training data is excluded from the test data. Various clips from the derf dataset were encoded in intra-only settings.

We compare the following three coders in the experiments:

- The standard reference VP9 codec
- VP9 codec with the proposed RD optimized filter-intra modes designed with no ACL approach
- VP9 codec with the proposed RD optimized filter-intra modes designed with the ACL approach

The bitrate reduction over the standard reference VP9 codec for sequences at low, middle and high resolutions are shown in Table 4.1, Table 4.2 and Table 4.3 respectively. Note that in the intra prediction scenario, the quantization error propagation is only limited within a frame, thus the instability problem is not as severe as it is in the temporal prediction case. However, the larger the frame resolution is, the more propagation there is, the more severe the instability problem is, and the more helpful the ACL frame can be. As shown in the following tables, the additional gain of using ACL is about 0.09%, 0.11% and 0.27% for low, mid and high resolution videos. Overall, the average compression gains of sequences at low, middle and high resolution are 2.59%, 3.36% and 3.59%.

4.5 Conclusion

The chapter proposes the four-tap recursive extrapolation filter based intra prediction, which captures the non-separable decaying spatial correlation within a block. Compared to prior work, the proposed technique effectively covers all the prediction directions, and adapts the filter coefficients to various local statistics, and aligns the filter design to the ultimate RD cost of the encoder. An ACL based design framework is used to avoid the instability problem due to quantization error propagation. Experimental results on the VP9 codec show consistent gains on sequences at all resolutions.

Chapter 5

Generalized Estimation-theoretic Framework for Scalable Video Coding

Scalable video coding suffers from the under-utilization of base layer information, where usually only the reconstruction in the base layer is used for enhancement layer prediction. Prior work from our lab proposed an optimal estimation-theoretic (ET) approach for quality scalable coding, wherein the estimates are obtained by utilizing all the available information from base layer quantization interval and enhancement layer distribution for transform coefficients. While this approach was proposed for fixed block size encoding, modern codecs employ variable block size quadtree structured partitioning, which results in different partitions at base layer and enhancement layer based on the rate-distortion trade-off, thus makes the base layer information not directly usable in the enhancement layer. Other new tools such as hybrid transform and the rate-distortion optimized quantizer (RDOQ) also have an impact on the information available for optimal estimation.

In this chapter[1], we generalize the ET framework for quality scalable video coding to account for the quadtree structured partitioning, hybrid transform and the RDOQ adjustment. Experimental evidence is provided for consistent coding gains over standard SHVC.

5.1 Introduction

Modern video applications (streaming, broadcasting, etc.) operate on RTP/IP [51] networks for real-time services, which is characterized by a broad range of connection qualities and receiving devices. To adapt to the differences in the end-user devices' capabilities and network conditions, scalable video coding (SVC) was proposed and adopted as extensions to video coding standards of H.264 [4] and HEVC (SHVC) [3]. SVC allows the video sequences to be encoded "progressively", i.e., a video sequence encoded at one quality can be enhanced to a higher quality by adding a refinement bitstream, successively any number of times. In this hierarchical structure, even if the top refinement bitstreams are lost due to temporary constraints in the network, the rest would still be a valid decodable bitstream. Specifically, bitstream at the lowest quality is referred to as the base layer (BL), while bitstreams at higher qualities are referred to as the enhancement layers (EL). In addition to quality scalability, there is also spatial and temporal scalability where the resolution and frame rate varies between layers.

A critical challenge that limits the practical use for SVC is how to exploit the BL information effectively in the ELs, especially in the EL prediction. In the SVC standards [4, 3], the BL reconstruction can be used as an additional reference frame for EL motion compensation, and the BL motion vectors can be used to predict EL motion vectors. In [52], a pyramid approach is proposed where the interpolated BL residual is used to

[1]This chapter is adapted from 978-1-5090-2175-8/17/$31.00 © 2017 IEEE.

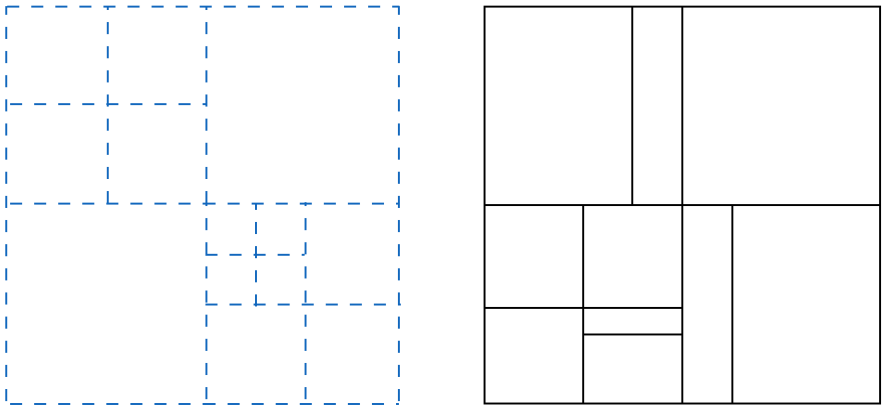


Figure 5.1: An example of different partitions at base and enhancement layer.

predict the EL residual. In [53, 54], a subband coding approach is proposed where the different resolutions of subband data are obtained from different layers. A linear combination of EL and BL with three additional weighting types is introduced in [55]. A rate-distortion (RD) optimized selection between the above approaches is proposed in [56, 57]. While all the prior approaches try to exploit the BL reconstruction for EL prediction, none of them utilize the BL quantizer information in the transform domain, which gives the exact region that the original value lies in. Prior work from our lab [22] exploits this quantization interval information and combines it with the transform coefficients distribution information in EL from an estimation-theoretic (ET) viewpoint, which provides the theoretically optimal EL prediction. A follow-up work on this [23, 24] significantly extended it to the spatially scalable video coding with considerable coding gains.

However, as more advanced tools are being developed for video coding, the ET approach has not been updated to account for them. One critical new tool to be supported is the quadtree block partition [1], which provides significant flexibility and hence commonly used in modern video coders. With this tool, the BL and ELs are generally partitioned

differently due to their different rate-distortion trade-offs, as shown in Fig. 5.1. This mismatch in partitions obviously carries over to the transform domain, and thus the BL interval information cannot be directly combined with the EL distribution information as proposed in ET prediction. The hybrid transform adopted by HEVC [1] and proposed for the next generation video codec JVET [58] greatly expand the family of transform kernels and leads to more variations in the distribution of transform coefficients, which needs to be accounted for properly. The rate-distortion optimized quantizer (RDOQ) [1], where the quantized index is adjusted to achieve better rate-distortion performance, results in erroneous interval information, i.e., the quantization interval does not always contain the true value of the coefficients, which also needs to be taken into account.

In this chapter, we generalize the ET framework to account for the advanced tools of quadtree partitioning, hybrid transform, and RDOQ for quality scalable video coding. To account for partitioning mismatch, we first use EL prediction parameters to generate transform coefficients distribution at BL transform unit size, then combine this with BL quantization interval information as in the standard ET approach, and finally transform this to generate final ET prediction in the EL prediction unit size. To account for the various types of transform, we train the distribution parameters for DCT, ADST and transform skip (TS) at different target bitrates separately, and apply them in tandem for hybrid transform. We adjust the quantizer interval information accounting for the RDOQ to avoid inaccurate interval information. The proposed approach is implemented in SHVC and compatible with all the existing features with no additional overhead and negligible additional complexity. Consistent gains across video sequences at different resolutions are presented to prove the efficacy of the approach.

5.2 Background: ET Prediction

The ET approach for EL prediction is formulated as an estimation problem of the current sample given all the available information. Without loss of generality, we assume there are only two layers, which are coded in a quality scalable encoder. For each sample in the EL, there are two sources of information available: EL reconstruction of prior samples, and the parameters (reconstruction, prediction, compressed residual, quantization parameters, etc.) associated with the BL coder for the same sample.

In a single-layer coder where only one information source is available, the prediction, $\tilde{x}$, can be derived via motion compensation or intra prediction. The residual, $x - \tilde{x}$, is then transformed, quantized, and sent to the decoder. It has been shown in prior work [59, 60, 61] that the DCT coefficients of the residual, ϵ, can be approximated by a Laplacian distribution centered at zero, $\frac{\lambda}{2}exp(-\lambda|\epsilon|)$. Therefore, the DCT coefficients, x^T, of the actual pixel value, x, would follow the same Laplacian distribution centered at the prediction in the transform domain $\tilde{x}^T$, i.e.,

$$f(x^T \mid \tilde{x}^T) = \frac{\lambda}{2}exp(-\lambda|x^T - \tilde{x}^T|). \tag{5.1}$$

The modern quantizer is hence designed as an uniform dead-zone quantizer based on the distribution and the quantization parameters (QP).

In a two-layer SNR scalable coder where two sources of information are available from BL and EL, the BL reconstruction is usually used as an additional reference and combined with EL prediction, either linearly as in the standard [3], or via other suboptimal approaches [52, 53, 54]. However there is more information available from the BL prediction and QP. Given quantized residual index i^b and QP, we know the exact interval

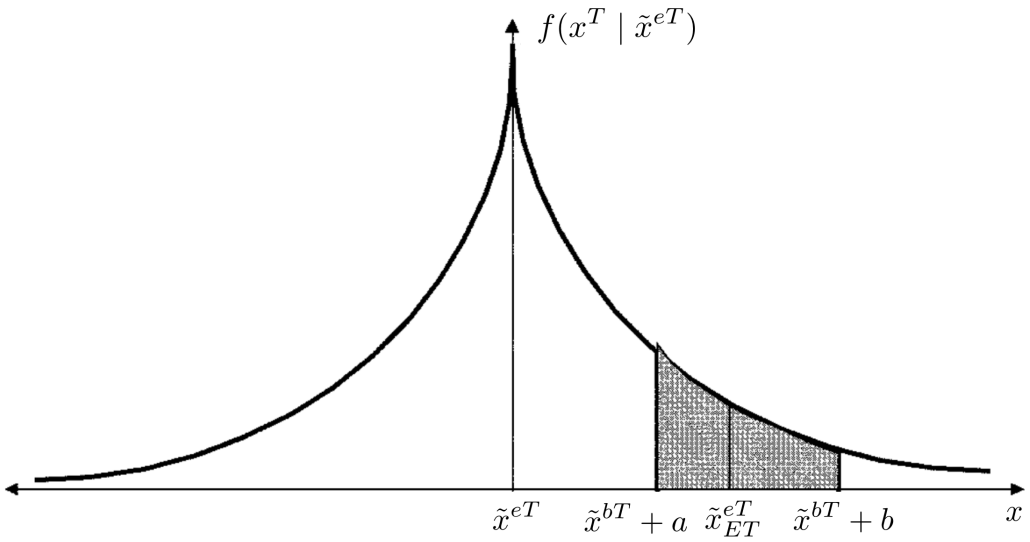


Figure 5.2: The distribution for transform coefficients (the centroid of the shaded region is its optimal ET prediction)

(a, b) associated with i^b. If $\tilde{x}^{bT}$ is the BL prediction in the transform domain, we have

$$\epsilon^b = x^T - \tilde{x}^{bT} \in (a, b), \tag{5.2}$$

$$x^T \in (\tilde{x}^{bT} + a, \tilde{x}^{bT} + b). \tag{5.3}$$

Similar to (5.1), the EL prediction, $\tilde{x}^{eT}$, provides the distribution information, $f(x^T \mid \tilde{x}^{eT}) = \lambda/2\ exp(-\lambda|x^T - \tilde{x}^{eT}|)$, and (5.3) provides the interval information from BL that indicates the region the original value would fall in. Together we have a truncated Laplacian distribution, as shown in Fig. 5.2, the centroid of which would be the best estimation for x^T(also referred to as ET prediction in the rest of the chapter),

$$\begin{aligned}\tilde{x}^{eT}_{ET} &= E(x^T \mid x^T \in (\tilde{x}^{bT} + a, \tilde{x}^{bT} + b), \tilde{x}^{eT}) \\ &= \frac{\int_{\tilde{x}^{bT}+a}^{\tilde{x}^{bT}+b} x^T f(x^T \mid \tilde{x}^{eT}) d(x^T)}{\int_{\tilde{x}^{bT}+a}^{\tilde{x}^{bT}+b} f(x^T \mid \tilde{x}^{eT}) d(x^T)}.\end{aligned} \tag{5.4}$$

5.3 ET prediction with partitioning mismatch between layers

In modern video codecs such as HEVC [1], each video frame is divided into 64×64 blocks (referred to as CTU), then each of them can be further splitted recursively into different sizes of coding units (CU) in a quadtree structure. At each leaf node of the quadtree, a CU can be further partitioned (rectangularly) into different prediction unit (PU), each associated with a motion vector or intra prediction mode. After the prediction, each CU is further split recursively in a similar quadtree method to different sizes of transform units (TU). In general, finer partition leads to better coding quality (less distortion) but at a higher bitrate. Depending on the target bitrate (or target quality) requirement, the encoder makes the partition decision based on the rate-distortion cost, which is a Lagrangian formula defined for the rate-distortion trade-off. ELs and BL are coded at different qualities, thus usually have different partition decisions across layers (as an example shown in Fig. 5.1).

As described in Section 5.2, in the ET approach, the BL interval information lies in the transform domain thus is determined by the BL TU sizes. However, if the BL TU is not aligned with EL PU, this interval information cannot be directly used with the transform domain distributions for EL prediction. Although, as a naive approach, we can make them compatible by performing a linear transform (from one size to another) either on the interval information or on the distribution information, it is neither effective or practical. Performing a linear transform on the interval information means finding the overall support region of a linear objective function, $y = \mathbf{c}^T\mathbf{x}$, with $x_i \in (a_i, b_i]$. This overall region would be, $y \in \bigcup R_i$, where $R_i = (c_i a_i, c_i b_i]$ if $c_i \geq 0$, and $R_i = [c_i b_i, c_i a_i)$ if $c_i < 0$. This would result in a much larger interval for y, and sometimes lead to meaningless information of $(-\infty, \infty)$ whenever any of the variables in $\mathbf{x}$ has no interval

information available (e.g., due to RDOQ as explained later). Performing the linear transform on the distribution involves a set of convolution operations, which are too complicated to be practical.

Instead, we propose an elegant and optimal solution, where we exploit the linearity property of expectations. From Section 5.2, we know the ET prediction for the EL, as in (5.4), is the centroid (a.k.a. the expectation) of the distribution for each transform coefficient, which by linearity property follows,

$$E(\mathbf{Y}) = E(\mathbf{A}^T\mathbf{X}\mathbf{A}) = \mathbf{A}^T E(\mathbf{X})\mathbf{A}. \tag{5.5}$$

Therefore, instead of performing the linear transform on the interval information or the distribution information, we directly work on the expectations, which retains the optimality of ET prediction even after any linear operation. To preserve flexibility for further TU partitioning after the prediction, we transform the ET prediction back to the pixel domain. Hence for our framework, $\mathbf{X}$ represents the transform domain coefficients in BL TU size, $\mathbf{Y}$ represents its corresponding pixels, and $\mathbf{A}$ corresponds to the inverse transform kernel. Specifically, we first transform the EL prediction in the same size as $\mathbf{X}$ to obtain $\tilde{x}^{eT}$, then calculate the optimal ET prediction $E(\mathbf{X})$ via (5.4), and finally calculate $E(\mathbf{Y})$ using (5.5). Transforming the EL prediction (in a PU) in the size of the BL TU involves blocks merging, extending and cropping. Depending on the mismatch of partitions, we consider the following three different cases:

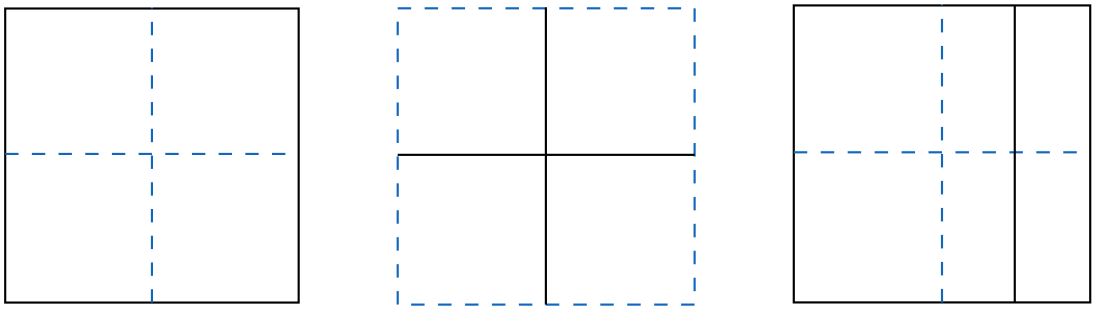
- *Case 1: The BL TUs are inside the EL PU (see Fig. 5.3(a)).* The EL PU is divided into small blocks that are aligned with BL TU. For each small block, the EL prediction is converted into transform domain to compute the optimal ET prediction using (5.4). Then we get the corresponding pixel domain ET prediction for each small block using (5.5), and finally merge them together to get the optimal

ET predicted EL PU.

- *Case 2: Part of the BL TU is outside the EL PUs (see Fig. 5.3(b)).* We could simply merge the EL predictions in different PUs, but this would introduce delay since the prediction of all the required PUs might not be available at the same instant. Instead, for each EL PU, we extend the EL prediction to the BL TU size using the same motion vector, and transform it to get the optimal ET prediction using (5.4). Then we get the corresponding pixel domain ET prediction using (5.5), and copy the corresponding prediction to the EL PU region. Also in this case, if it is intra predicted in the EL, we skip ET prediction due to lack of boundary information for extending beyond EL PU region.

- *Case 3: An EL PU covers multiple BL TUs, some of which are fully inside the EL PU and some extend outside (see Fig. 5.3(c)).* We transform such an EL PU using the same size as BL TUs via both division and extension. For the BL TUs fully inside an EL PU, we divide the EL PU into the same sizes as the BL TUs as we do in case 1; for those partly outside an EL PU, we extend the prediction to the size of BL TU as we do in case 2. The optimal ET prediction in pixel domain for all the divisions and extensions are merged together as the overall prediction.

The full block diagram of the EL prediction framework with ET scheme in scalable video coding with quadtree structured partitioning is shown in Fig. 5.4. In the traditional EL prediction without the ET scheme (the red block), the only information exploited from the BL is the motion vector and the reconstruction, while with the ET scheme we are also using the partition and quantization interval information from the BL. For each effective EL prediction via motion compensation or intra direction, we enhance it using the ET approach, and compare it with the BL reconstruction and use the best one as prediction. Note that although in principle the BL reconstruction is contained within



(a) Case 1 (b) Case 2 (c) Case 3

Figure 5.3: Three cases of the partition mismatch between the EL PU (black line) and BL TU (blue dotted line)

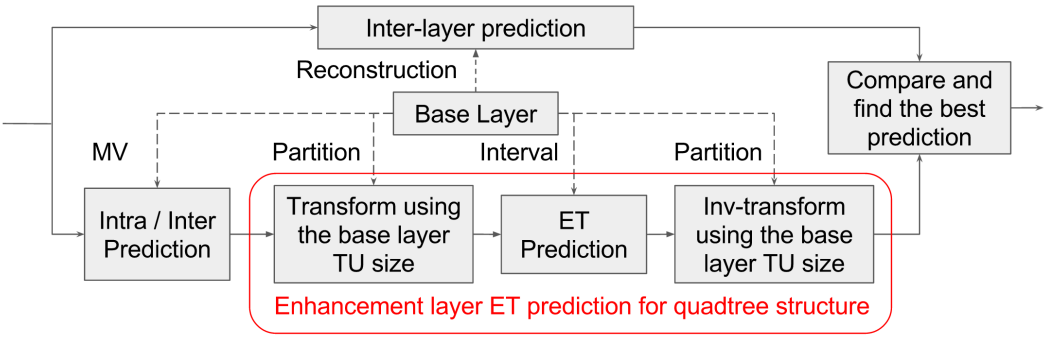


Figure 5.4: The EL framework block diagram in SHVC with ET prediction

the interval information, we noticed that directly referencing from base layer sometimes yields better rate-distortion performance due to savings in side information. The residual is then transformed and quantized using the most optimal TU quadtree structure. One of the future research directions will be to further exploit and account for the BL partition information while optimizing the CU/TU partition in EL.

It has been shown that DCT is not always the best separable transform to approximate KLT. Hence, modern video coders, such as HEVC, employ hybrid transform (DCT, ADST) for better decorrelation under certain conditions, and transform skip (TS), where quantization is done directly in pixel domain. Though the Laplacian distribution assumption for transform coefficients is usually only valid for DCT, we extend it to ADST and TS, and train the λ for the three different transform types, following the maximum-likelihood estimation, with N number of samples as,

$$\lambda = \frac{N}{\sum_{i=0}^{N-1} |x_i^T - \tilde{x}_i^{eT}|}. \tag{5.6}$$

Since statistics of prediction, $\tilde{x}^{eT}$, also depend on QP and transform block size, we train separate λ for different range of QPs and for each block size. We then employ these λ adaptively according to the QP and block size chosen by the encoder.

To improve the overall performance, rate-distortion optimized quantizer (RDOQ) was introduced in recent video coding standards. For each residual transform coefficient, in addition to its correct quantizer magnitude L, the encoder also considers two additional magnitudes $L-1$ and 0, and chooses the one with the lowest RD cost. Similarly, the encoder also has the option to eliminate a whole coefficient group (which is usually 4×4) if it is cost effective. The skip mode (where the residual of the whole block is set to 0) is also used quite often when the bitrate budget is low. All of these techniques contribute significantly in improving the RD performance, but they also result in inaccurate interval

information if derived solely from the quantization index. [23] dealt with a simpler variation of RDOQ in H.264 by disabling the ET prediction for a certain corner case. But in SHVC, we need a more robust approach to address the problem. Let's denote the quantizer interval associated with index i^b as $I_{i^b} = (a_{i^b}, b_{i^b}]$. Since we employ regular quantizers, the intervals of neighboring indices are consecutive, i.e., $a_{i+1} = b_i$. We propose a more robust rule to account for RDOQ by expanding the BL interval information in the following way:

- If $i^b = 0$, set the interval as $I_{-1} \bigcup I_0 \bigcup I_1 = (a_{-1}, b_1]$
- If $i^b > 0$, set the interval as $I_{i^b+1} \bigcup I_{i^b} = (a_{i^b}, b_{i^b+1}]$
- If $i^b < 0$, set the interval as $I_{i^b} \bigcup I_{i^b-1} = (a_{i^b-1}, b_{i^b}]$
- If $i^b = 0$ for all the transform coefficients in the block, then this block is very likely to be coded in skip mode, where no interval information is available, i.e., the interval is $(-\infty, \infty)$, and thus the ET prediction $\tilde{x}^{eT}_{ET}$ is the same as the EL prediction $\tilde{x}^{eT}$.

5.4 Experimental Results

To evaluate the performance, the proposed ET framework is implemented in SHM 8.0, and is compared to standard SHVC with two-layer quality scalability. Eleven test sequences were tested in the lowdelay P configuration, each with four bitrate points: BL QPs (25, 30, 35, 40) combined with an EL QP offset of -3 (which results in a BL bitrate of about half of the total bitrate). Similar to [23], we use a look-up table to store the centroid offset within the interval, which significantly reduces the complexity of the ET framework.

Table 5.1: Prediction gains for blocks with valid ET prediction in EL

	Prediction Gain
BQMall (480p)	2.67 dB
BasketballDrill (480p)	1.69 dB
Keiba (480p)	3.00 dB
FourPeople (720p)	2.07 dB
Johnny (720p)	0.75 dB
Vidyo1 (720p)	1.29 dB
Cactus (1080p)	2.77 dB
BasketballDrive (1080p)	2.89 dB
BQTerrace (1080p)	1.78 dB
Kimono (1080p)	5.05 dB
ParkScene (1080p)	3.20 dB
Average	**2.47 dB**

Table 5.2: Overall bitrate reduction of the ET framework

	Standard ET-SHVC	Constrained ET-SHVC
BQMall (480p)	3.43%	3.99%
BasketballDrill (480p)	4.21%	2.43%
Keiba (480p)	2.13%	0.29%
FourPeople (720p)	3.88%	4.83%
Johnny (720p)	2.92%	3.64%
Vidyo1 (720p)	3.23%	4.43%
Cactus (1080p)	4.41%	3.92%
BasketballDrive (1080p)	2.84%	1.41%
BQTerrace (1080p)	2.45%	4.13%
Kimono (1080p)	3.12%	1.12%
ParkScene (1080p)	3.94%	4.78%
Average	**3.32%**	**3.18%**

We conducted three sets of experiments to show the effectiveness of the proposed ET framework. In our first experiment, we evaluate the prediction gain purely from the blocks that have valid ET prediction in EL (i.e., these blocks have valid base layer interval information available). As shown in Table 5.1, the ET prediction framework provides an average 2.47dB gain in prediction (equivalent to 45% reduction in prediction error). However, in practice, only 3% to 15% of the blocks (depending on the bitrate) have a valid interval information from base layer, which largely dilutes the overall gain. In our second experiment, we compare the overall RD performance of the SHVC with ET framework and the standard SHVC, and get an average of 3.3% reduction in bdrate [62], as shown in the "Standard ET-SHVC" column of Table 5.2. This dilution also suggests a future research direction of jointly optimizing BL and EL, where interval information is introduced in BL so as to benefit the ET prediction in EL.

To show that we have effectively tackled the challenges due to the quadtree structured partitioning, hybrid transform, and RDOQ, we conducted a third experiment where ET framework is applied on a constrained SHVC where none of the above tools are enabled. In this third experiment, all the block sizes are forced to be 8×8, DCT is used as the only transform and RDOQ is disabled. The performance of the original ET framework in this limited version of SHVC over the baseline is shown in the "Constrained ET-SHVC" column of Table 5.2, with an average of 3.18% reduction in bdrate. We achieve a similar and consistent gain in our proposed framework with all the tools enabled, which proves its effectiveness in practice.

5.5 Conclusion

This chapter generalizes the ET framework to manage the mismatch in quadtree structured partitioning at different layers, by exploiting the linearity property of estima-

tions to convert information between different partitions. The parameter of the transform coefficient distribution is separately trained for different types of transform, block sizes and QPs. And a more robust way of exploiting the BL quantization interval information is proposed to avoid erroneous information due to the RDOQ adjustment. Experimental results demonstrate the effectiveness of the proposed technique with consistent gains over standard SHVC. Future research directions include the joint optimization of BL and EL, and further exploitation of BL partition information.

Chapter 6

Asymptotic Closed-loop Based Transform Design

Data-dependent transforms can achieve better energy compaction than regular transforms (like DCT, ADST) if designed properly. In this chapter, we extend the ACL idea to the transform module in the predictive coding system, designing a set of separable KLT to cover the various temporal residual block statistics. The designed transforms are tested in the VP9 codec, where the encoder chooses one from the transform set per inter coded block. We show that the transform set designed with ACL consistently outperforms those designed without ACL.

6.1 Introduction

Transform coding is widely adopted in image and video compression. It decorrelates the spatial correlation and redistributes the energy within the residual blocks, so that the transformed signals have little redundancy and better energy compaction. The Karhunen-Loeve transform (KLT) has been proven to be the optimal transform. But its dependence

on data as well as the high complexity makes it impractical for many applications. The discrete cosine transform (DCT) has long been a popular substitute for KLT due to its fast implementation and the good energy compaction property.

However, DCT is not always a good approximation of KLT (unless the signal is a first-order Markov process [63]). In natural video sequences, the residual usually depends on the prediction. For example, the residual of an intra block might have stronger correlation along the direction of the prediction mode; the residual of an inter block might have stronger correlation at block boundaries or object boundaries based on its prediction model.

There have been lots of research on designing transforms that better describe the residual statistics. Han et. al. proposed the Asymmetric Discrete Sine Transform (ADST) in [64] to describe intra residual blocks where only partial boundary information is available. Directional DCTs for inter and intra are proposed respectively in [65] and [66] to cover directional correlations more than horizontal and vertical directions. Compared to KLT, separable KLT is much faster with great energy compaction on the rows and columns. A mode-dependent separable KLT approximation for intra residual blocks is proposed in [67]. [68] learnt the separable KLT by applying SVD on the row and column directions, while [69] and [70] learnt it by approximating the inverse covariance matrix.

More advanced transform structures were proposed to approximate non-separable filters with less complexity. Row-column transform (RCT) [71] defines 2D non-separable transforms with the aid of a set of 1D linear transforms and a basis ordering permutation, while Layered-givens transforms (LGT) [72] approximates them using layers of permutations and rotations. A more generic and algebraic approach called Sparse Orthogonal transform (SOT) was proposed in [73], which enforces sparsity on transform coefficients and reduces to KLT on Gaussian signals.

Despite the comprehensive research on the transform models, there is a catastrophic

instability problem (as described in Chapter 3) in the training of the transform models. The transforms are trained based on the statistics of the residual blocks, which is changed after applying the designed transforms. The transforms determine the residual signals to be quantized and reconstructed, and in a closed-loop system like video coding, the reconstruction would in return be used to predict future frames. Thus any changes from applying the designed transforms would propagate to future frames, leading to mismatched statistics with the statistics we designed for. In this chapter, we will use the two-loop asymptotic closed-loop design approach which we earlier proposed for predictor design to design transforms that correctly represent the residual statistics.

Compared to intra residual data, inter residual has less structured pattern and is more prone to the instability problem. Therefore, we chose to design a set of separable KLTs for different classes of residual data using the two-loop ACL based mode design approach. It also provides a generic transform design framework free of the instability problem, which can be used to train any arbitrary transforms.

6.2 Background

6.2.1 KLT

If we denote $\mathbf{x}$ as a signal vector, and $\mathbf{y}$ is the signal vector in the transform domain. KLT is defined as

$$\mathbf{y} = \mathbf{T}^T\mathbf{x} \tag{6.1}$$

with the KLT transform matrix $\mathbf{T}$ defined by

$$\mathbf{K_x} = \mathbf{T}\Lambda\mathbf{T}^T \tag{6.2}$$

where the $\mathbf{K_x}$ is the covariance matrix of $\mathbf{x}$, and Λ is a diagonal matrix. In other words, each column of the KLT matrix is the eigenvector of the covariance matrix. The covariance matrix of the transformed signal $\mathbf{y}$ is the diagonal matrix Λ, thus any element in $\mathbf{y}$ has zero correlation with any other elements. Therefore, KLT can fully de-correlate $\mathbf{x}$.

It can be shown that the transform coding gain is related to energy compaction, and can be measured by the ratio between the arithmetic mean and the geometric mean of the variances of all the elements in the transformed vector [74].

$$G_T = \frac{D_Q}{D_T} = \frac{1/k \sum_i \sigma_{y_i}^2}{(\prod_i \sigma_{y_i}^2)^{1/k}} \tag{6.3}$$

where D_T and D_Q are the distortion in reconstruction with and without the transform, and G_T is referred to as the transform coding gain given a fixed average bitrate.

As both KLT and DCT are orthogonal transforms, they do not change the arithmetic mean of the variances. Therefore, the smaller the geometric mean is, the larger the coding gain is. KLT is also proved to be the optimal transform for minimizing the geometric mean. In the experiments, we also use this metric to measure the energy compaction of the designed transforms.

6.2.2 Separable KLT

For a 2D signal $\mathbf{X}$, if we can assume all the columns have the same covariance matrix $\mathbf{K}_c$, and all the rows have the same covariance matrix $\mathbf{K}_r$. We can write the KLT in a separable form [75]

$$\mathbf{Y} = \mathbf{T}_c^T \mathbf{X} \mathbf{T}_r \tag{6.4}$$

where $\mathbf{Y}$ is the 2D transform coefficients, and $\mathbf{T}_c$, $\mathbf{T}_r$ are the KLT matrix corresponding to $\mathbf{K}_c$, $\mathbf{K}_r$ respectively, i.e.,

$$\mathbf{K}_c = \mathbf{T}_c \Lambda_c \mathbf{T}_c^T \tag{6.5}$$

$$\mathbf{K}_r = \mathbf{T}_r \Lambda_r \mathbf{T}_r^T \tag{6.6}$$

It can be shown that,

$$\mathbf{K}_\mathbf{x} = \mathbf{K}_r \otimes \mathbf{K}_c \tag{6.7}$$

$$= (\mathbf{T}_r \otimes \mathbf{T}_c)(\Lambda_r \otimes \Lambda_c)(\mathbf{T}_r \otimes \mathbf{T}_c)^T \tag{6.8}$$

Therefore, the KLT for $\mathbf{x}$ can be written in the form of the two KLTs for the row vectors and column vectors, $\mathbf{T} = \mathbf{T}_r \otimes \mathbf{T}_c$

6.2.3 Transform tools in AV1

In the latest development of AV1 [76], a set of 16 transforms are introduced for inter and intra residual blocks. For each block, the codec can choose to use one of up to 16 different transforms as follows:

$$\{\text{DCT, ADST, FlipADST, IDTX}\}_{\text{horizontal}} \times \{\text{DCT, ADST, FlipADST, IDTX}\}_{\text{vertical}} \tag{6.9}$$

where FlipADST is the flipped version the ADST, and IDTX is an identity transform (same as transform skip (cite)) which is good for screen content video sequences. A reduced set of transforms is used for 16×16, 32×32, 64×64 as some of the transforms act similarly for larger blocks.

In this chapter, we start from the default transforms in AV1, divide the blocks into

several classes, and optimize the separable KLT for each class. Clustering techniques are used to optimize the mode assignment and mode design. ACL techniques are used to ensure the design is stable for the closed-loop system.

6.3 ACL Based Transform Design

Algorithm 3: ACL based transform design algorithm

Input : A set of training sequences, the number of transforms K
Output: A set of separable KLT kernels of size K

Initialize the separable KLTs using the default transform set Run the codec in closed-loop, store the encoder decision d, mode assignment a and reconstructed video file rec;
while $iter < max\_iter$ **do**
  while *RDcost decreases* **do**
    Run the codec in open-loop using the same encoder decision d, but with rec as the motion compensation reference;
    Extract the residual blocks;
    while *RDcost decreases* **do**
      Design the separable KLT for each cluster;
      Assign the residual blocks into the best mode to minimize $RDcost$;
      Update a, update $RDcost$;
    end
    Run the codec in open-loop again, using the same d, rec, and the optimized a and transform sets; Update rec as the newly reconstructed video file; Update $RDcost$;
  end
  Run the codec in closed-loop, update the encoder decision d and reconstructed video file rec, store the $RDcost$ into array $\{cost\}$. $iter = iter + 1$;
end
Find the minimum $RDcost$ in $\{cost\}$, set the corresponding transform sets as the final trained transforms;

Although the separable KLTs can achieve better energy compaction than the default transform sets, they are designed based on the residual statistics, which depend on the reconstruction of the reference frames in a closed-loop system. As the designed separable

KLTs are applied to the residual blocks, we may get a different reconstruction based on how the new transform coefficients fall in the quantizer intervals. This change in reconstruction is passed on to future frames through the prediction loops, leading to different statistics in residual blocks. In a word, the instability problem we discussed in Chapter 3 not only applies to prediction module, but also to other components as well, such as the transform module. Therefore, we use a similar ACL framework in the transform design.

Given a fixed bitrate, the designed transforms can reduced the reconstruction distortion by G_T times, with the transform coding gain G_T defined in (6.3). While running the codec in open loop, the reconstruction is guaranteed to be improved, and with a better reconstruction as reference, the reconstruction of the next iteration is generally improving too. The distortion of reconstruction decreases until convergence, when the open loop is equivalent to closed loop.

Similar to the framework in Chapter 3, the overall training algorithm is shown in Algorithm 3. Since the codec is running at constant bitrate, minimizing RD cost is equivalent to minimizing the overall distortion in reconstruction. With this example, we show that this ACL framework can be generally used in optimizing various modules in the offline training for video coding.

6.4 Experimental Results

We tested the designed transforms in the VP9 codec, where the set of 16 transforms $\{\text{DCT, ADST, FlipADST, IDTX}\}_{\text{horizontal}} \times \{\text{DCT, ADST, FlipADST, IDTX}\}_{\text{vertical}}$ is one of the experimental features. Instead of 16 transforms, we only focus on the 9 of them in this experiment, $\{\text{DCT, ADST, FlipADST}\}_{\text{horizontal}} \times \{\text{DCT, ADST, FlipADST}\}_{\text{vertical}}$, since the identity transform (IDTX) is mostly intended for screen content sequences,

Table 6.1: The transform coding gain (dB) for sequence *paris* at different bitrate using different transform set

Target bitrate	Default	non-ACL designed separable KLTs	ACL designed separable KLTs
100	5.60	5.44	5.77
200	3.60	3.80	3.84
300	3.20	3.43	3.71
500	2.93	3.08	3.31
700	2.79	2.93	2.97

Table 6.2: Bitrate reduction over reference VP9 using the transform sets for inter residual blocks

Sequences	Default	non-ACL designed separable KLTs	ACL designed separable KLTs
Akiyo	3.80	4.37	6.33
City	1.21	1.80	3.00
Coastguard	3.90	5.47	9.68
Container	0.58	0.89	2.12
Crew	2.24	1.56	4.48
Harbour	2.37	3.42	4.75
News	1.09	5.92	9.60
Paris	2.03	2.21	3.51
Silent	3.32	4.32	5.53
Stefan	2.25	3.37	4.11
Waterfall	0.13	0.63	1.37
Mobile	1.71	2.14	3.16
Tempete	0.64	2.48	3.41
Flower	3.74	4.64	5.95
Husky	2.92	3.47	3.79
Hall moniitor	7.40	11.01	12.64
Bridge-close	5.22	5.18	5.87
Football	-3.10	-3.93	-2.77
Average	**2.31**	**3.28**	**4.8**

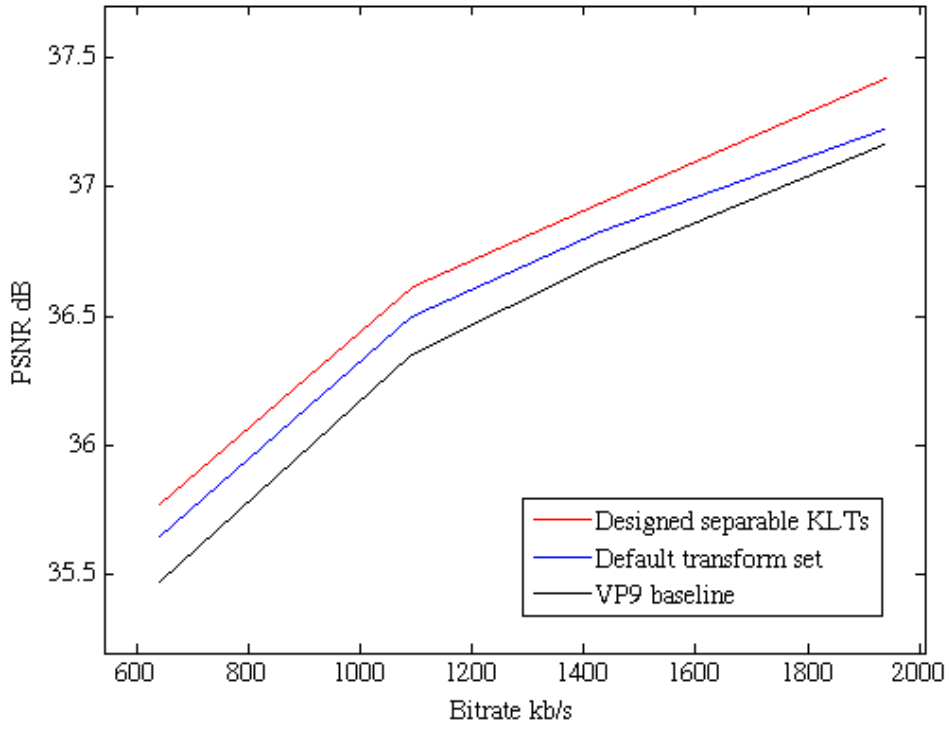


Figure 6.1: Coding performance comparison for sequence *Hall monitor* at *CIF* resolution

which are not in the test sequences here. To simply the experiments, the transform block size is restricted to 8×8. The encoder can choose one of the 9 transforms for each inter residual block, and indicate the choice in the bitstream as extra overhead.

We first evaluate the transform coding gain that measured by the ratio between the arithmetic mean and the geometric mean of the variances of the transform coefficients. In Table 6.1, we show the transform coding gain at different bitrate using the default transform set, the separable KLTs designed without the ACL framework and the separable KTLs designed with the ACL framework. We can see that for transform set designed without ACL, the transform gain is better than the default transform set at middle and high bitrate, yet gets worse at low bitrate. As mentioned in Chapter 3, the instability problem becomes especially severe at low bitrate encoding case, since there is more quantization error propagating through the prediction loop, thus leads to worse performance in the non-ACL design. For transform set designed with ACL, the transform gain is consistently better than the default transform set and the non-ACL designed transform set at all bitrates.

Compared to the baseline results (where only DCT is used for inter residuals), we present the bitrate reduction of using the default 9 transforms, the separable KLTs designed without ACL, the separable KLTs designed with ACL in Table 6.2. On average, we can get an extra 0.97% in bitrate reduction by using separable KLTs over the default transforms, and an extra 2.49% by designing separable KLTs with the ACL approach. The RD curve of sequence *Hall monitor* is shown in Fig. 6.1, showing the performance of baseline VP9 and VP9 using the default transforms and the designed transforms. Very rarely, the overhead to encode the transform mode outweighs the gain of using multiple transforms (e.g.: sequence *Football*). This problem can be solved by finding the optimal number of modes, which would be one of the future research directions.

6.5 Conclusion

The instability problem widely exists in the predictive coding system. In this chapter, we extended the ACL idea to the transform module and designed a set of separable KLTs to cover the various temporal residual block statistics. We compared the designed transforms with the default transforms in VP9, and got an extra 2.5% in bitrate reduction. We also compared the ACL-designed transforms with non-ACL designed transforms. While the non-ACL designed transforms would lead to worse performance at low bitrate case, the ACL-designed transforms outperform the default transforms at all bitrates.

Chapter 7

Conclusions and Future Work

In this dissertation, we addressed the sub-optimality of the traditional prediction approach in video coding, and proposed effective yet manageable models for temporal, spatial and scalable video coding prediction. We also proposed a two-loop ACL framework for offline training to avoid the design instability in closed-loop systems. This two-loop ACL framework was furthered proved to be useful in not only predictor design, but also in other video compression modules such as transform design. In most of the approaches we proposed in this dissertation, the ACL framework is used along with clustering approaches for effective mode design.

In Chapter 2, we proposed the extended block transform domain temporal prediction, to fully decouple the spatial and temporal correlation for an optimal temporal prediction in the transform domain. The proposed approach also allows us to better exploit the temporal correlation in the high frequency components, which is usually neglected in the traditional pixel domain temporal prediction. We further recognized that the proposed predictors interferes with the sub-pixel interpolation filters in the high frequency components, and as a result, we proposed a joint design approach to optimize the predictors and interpolators together. To make the predictors better adapt to local statistics, we

trained different predictors for different combinations of encoder decisions (block sizes, QP, etc.). The proposed approach was implemented in HEVC, and showed significant coding gain over standard HEVC.

In Chapter 3, we explained the design instability in the closed-loop systems, and proposed an iterative open-loop design technique that asymptotically optimizes the system for closed-loop operations, called an asymptotic-closed loop (ACL) approach. To further stabilize it for video coding system, we extend it to a two-loop ACL framework where the encoder decisions are optimized in terms of rate-distortion cost in the outer loop, and predictors are optimized in terms of prediction error in the inner loop. Clustering approach like K-modes approach is incorporated into the innermost loop to optimize the mode design. We compared the performance gain using TDTP predictors designed with and without ACL approach, and showed that in low bitrate cases the instability problem is so severe that designing without ACL approach would result in worse performance than the baseline.

In Chapter 4, we designed a recursive extrapolation filter based intra prediction approach that effectively captures the non-separable spatial correlation and its variation within blocks without incurring much extra complexity. This predictor is optimized via a two-phase approach where it is first optimized to minimize prediction error, and then optimized to minimize the rate-distortion cost. Gradient descent is applied in both phases with gradients calculated either analytically or empirically. The ACL framework is also applied to avoid the design instability problem, although the quantization error propagation in this scenario is only limited within a frame. Consistent gains over standard VP9 on sequences at various resolutions are presented, and the proposed intra predictor was adopted by the latest video codec AV1.

In Chapter 5, we generalized the estimation-theoretic (ET) approach in scalable video coding enhancement prediction to account for modern coding tools, such as quadtree par-

tition, hybrid transform and rate-distortion optimized quantizer (RDOQ). The original ET approach relies on the alignment of the block partition in enhancement layers and base layer, so that the base layer quantizer interval information can be aligned with the enhancement layer distribution information to generate a better prediction. We addressed this mis-alignment problems in the quadtree structure by exploiting the linearity property of expectations. The RDOQ, which would generate erroneous interval information, and the hybrid transform, which leads to more variation in the distribution of transform coefficients, are also taken in account properly. The generalized approach fully supports the latest scalable video coding codec, SHVC, and leads to substantial coding gains.

In Chapter 6, we applied the ACL framework in designing transforms, another important component in video compression other than the prediction module, and proved that this framework is not only crucial in the predictor design, but also in the design of other modules in the predictive coding system. Using the ACL framework, we designed a set of separable KLTs for temporal residual blocks of different statistics, and incorporate them into the VP9 codec for the encoder to choose one per block. The performance gain over the standard VP9 using transform set designed with ACL consistently outperforms those designed without ACL.

The work in this dissertation also leads to some interesting directions for future research. In the inner loop of the ACL framework where the optimization is mostly based on mean square error, it is not guaranteed that the reconstruction error would decrease monotonically since the quantizer works intrinsically based on rate-distortion theory. Despite the workaround of fixing the bitrate, improving the ACL framework so that it is RD optimal would be an interesting topic to research on. In the mode design, we pre-determine the number of modes empirically, K, and then use the K-modes clustering approach to optimize them. However, depending on the models, statistics, and the RD

trade-off, the optimal number of modes can vary tremendously. Optimizing the number of modes would be another research topic of very broad interests.

Bibliography

[1] G. J. Sullivan, J. 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* **22** (2012), no. 12 1649–1668.

[2] D. Mukherjee, J. Han, J. Bankoski, R. Bultje, A. Grange, J. Koleszar, P. Wilkins, and Y. Xu, *A technical overview of vp9the latest open-source video codec*, *SMPTE Motion Imaging Journal* **124** (2015), no. 1 44–54.

[3] 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* **26** (2016), no. 1 20–34.

[4] 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* **17** (2007), no. 9 1103–1120.

[5] M. P. Sharabayko, O. G. Ponomarev, and R. I. Chernyak, *Intra compression efficiency in vp9 and hevc*, *Applied Mathematical Sciences* **7** (Jul, 2013) 6803–6824.

[6] Y. Chen, J. Han, T. Nanjundaswamy, and K. Rose, *A joint spatio-temporal filtering approach to efficient prediction in video compression*, in *IEEE Picture Coding Symposium (PCS)*, pp. 81–84, 2013.

[7] Y. Chen, D. Mukherjee, J. Han, and K. Rose, *Joint inter-intra prediction based on mode-variant and edge-directed weighting approaches in video coding*, in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7372–7376, 2014.

[8] J. Xin, K. N. Ngan, and G. Zhu, *Combined inter-intra prediction for high definition video coding*, in *IEEE Picture Coding Symposium (PCS)*, 2007.

[9] A. Gabriellini, D. Flynn, M. Mrak, and T. Davies, *Combined intra-prediction for high-efficiency video coding*, *IEEE Journal of selected topics in Signal Processing* **5** (2011), no. 7 1282–1289.

[10] S.-J. Choi and J. W. Woods, *Motion-compensated 3-d subband coding of video*, *IEEE Transactions on Image Processing* **8** (Feb, 1999) 155–167.

[11] J.-R. Ohm, *Three-dimensional subband coding with motion compensation*, *IEEE Transactions on Image Processing* **3** (Sep, 1994) 559–571.

[12] J. Kim and J. W. Woods, *Spatio-temporal adaptive 3-D kalman filter for video*, *IEEE Transactions on Image Processing* **6** (1997), no. 3 414–424.

[13] T. Wedi, *Adaptive interpolation filter for motion and aliasing compensated prediction*, in *Electronic Imaging*, pp. 415–422, International Society for Optics and Photonics, 2002.

[14] S. Li, O. G. Guleryuz, and S. Yea, *Reduced-rank condensed filter dictionaries for inter-picture prediction*, in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1428–1432, 2015.

[15] W. Jiang and A. Ortega, *Forward/backward adaptive context selection with applications to motion vector field encoding*, in *IEEE International Conference on Image Processing (ICIP)*, vol. 2, pp. 168–171, 1997.

[16] J. Han, V. Melkote, and K. Rose, *Transform-domain temporal prediction in video coding with spatially adaptive spectral correlations*, in *IEEE International Workshop on Multimedia Signal Processing (MMSP)*, pp. 1–6, 2011.

[17] O. Maimon and L. Rokach, *Data Mining and Knowledge Discovery Handbook*. Series in Solid-State Sciences. Springer US, 2010.

[18] S. Jaballah, K. Rouis, and J. B. Tahar, *Clustering-based fast intra prediction mode algorithm for hevc*, in *IEEE Signal Processing Conference (EUSIPCO), 2015 23rd European*, pp. 1850–1854, 2015.

[19] W. Zhu, W. Ding, Y. Shi, Y. Sun, and B. Yin, *Adaptive intra modes reduction by clustering for h. 264/avc*, in *IEEE International Conference on Image Processing (ICIP)*, pp. 1665–1668, 2011.

[20] J. Han, V. Melkote, and K. Rose, *Transform-domain temporal prediction in video coding: exploiting correlation variation across coefficients*, in *IEEE International Conference on Image Processing (ICIP)*, pp. 953–956, 2010.

[21] Y. Chen, *Towards Joint Optimality of Spatial, Temporal Prediction and Transform Coding of Video Signals*. University of California, Santa Barbara, Santa Barbara, Calif.], 2016.

[22] K. Rose and S. L. Regunathan, *Toward optimality in scalable predictive coding*, *IEEE Transactions on Image Processing* **10** (2001), no. 7 965–976.

[23] J. Han, V. Melkote, and K. Rose, *An estimation-theoretic framework for spatially scalable video coding*, *IEEE Transactions on Image Processing* **23** (2014), no. 8 3684–3697.

[24] J. Han, V. Melkote, and K. Rose, *An estimation-theoretic approach to spatially scalable video coding*, in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 817–820, 2012.

[25] L. K. Liu and E. Feig, *A block-based gradient descent search algorithm for block motion estimation in video coding*, *IEEE Transactions on Circuits and Systems for Video Technology* **6** (1996), no. 4 419–422.

[26] A. M. Tourapis, O. C. Au, and M. L. Liou, *Predictive motion vector field adaptive search technique (pmvfast)-enhancing block based motion estimation*, in *Proceedings of SPIE*, vol. 4310, pp. 883–892, 2001.

[27] J. Konrad and E. Dubois, *Bayesian estimation of motion vector fields*, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **14** (1992), no. 9 910–927.

[28] T. Wedi and H. G. Musmann, *Motion-and aliasing-compensated prediction for hybrid video coding*, *IEEE Transactions on Circuits and Systems for Video Technology* **13** (2003), no. 7 577–586.

[29] T. Wiegand, E. Steinbach, and B. Girod, *Affine multipicture motion-compensated prediction*, *IEEE Transactions on Circuits and Systems for Video Technology* **15** (2005), no. 2 197–209.

[30] R. C. Kordasiewicz, M. D. Gallant, and S. Shirani, *Affine motion prediction based on translational motion vectors*, *IEEE Transactions on Circuits and Systems for Video Technology* **17** (2007), no. 10 1388–1394.

[31] M. Chan, Y. Yu, and A. Constantinides, *Variable size block matching motion compensation with applications to video coding*, *IEE Proceedings I (Communications, Speech and Vision)* **137** (1990), no. 4 205–212.

[32] J. Lee, *Optimal quadtree for variable block size motion estimation*, in *IEEE International Conference on Image Processing (ICIP)*, vol. 3, pp. 480–483, 1995.

[33] D. Wang, C. Labit, and J. Ronsin, *Segmentation-based motion-compensated video coding using morphological filters*, *IEEE Transactions on Circuits and Systems for Video Technology* **7** (1997), no. 3 549–555.

[34] S. Li, T. Nanjundaswamy, and K. Rose, *Transform domain temporal prediction with extended blocks*, in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1476–1480, 2016.

[35] S. Li, T. Nanjundaswamy, and K. Rose, *Jointly optimized transform domain temporal prediction and sub-pixel interpolation*, .

[36] S. Li, T. Nanjundaswamy, Y. Chen, and K. Rose, *Asymptotic closed-loop design for transform domain temporal prediction*, in *IEEE International Conference on Image Processing (ICIP)*, pp. 4907–4911, 2015.

[37] S. Wittmann and T. Wedi, *Separable adaptive interpolation filter for video coding*, in *IEEE International Conference on Image Processing (ICIP)*, pp. 2500–2503, 2008.

[38] K. Ugur, J. Lainema, and M. Gabbouj, *Adaptive interpolation filter with flexible symmetry for coding high resolution high quality video*, in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 1, pp. I–1013, 2007.

[39] D. Rusanovskyy, K. Ugur, and M. Gabbouj, *Adaptive interpolation with flexible filter structures for video coding*, in *IEEE International Conference on Image Processing (ICIP)*, pp. 1025–1028, 2009.

[40] Y. Ye, G. Motta, and M. Karczewicz, *Enhanced adaptive interpolation filters for video coding*, in *IEEE Data Compression Conference (DCC)*, pp. 435–444, 2010.

[41] Y. Vatis and J. Ostermann, *Locally adaptive non-separable interpolation filter for H. 264/AVC*, in *IEEE International Conference on Image Processing (ICIP)*, pp. 33–36, 2006.

[42] Y. Vatis, B. Edler, D. T. Nguyen, and J. Ostermann, *Motion-and aliasing-compensated prediction using a two-dimensional non-separable adaptive wiener interpolation filter*, in *IEEE International Conference on Image Processing (ICIP)*, vol. 2, pp. II–894, 2005.

[43] K. S. Al-Sultana and M. M. Khan, *Computational experience on four algorithms for the hard clustering problem*, *Pattern recognition letters* **17** (1996), no. 3 295–308.

[44] L. Zhang, X. Zhao, S. Ma, Q. Wang, and W. Gao, *Novel intra prediction via position-dependent filtering*, *Journal of Visual Communication and Image Representation* **22** (Nov, 2011) 687–696.

[45] H. Kimata, M. Kitahara, and Y. Yashima, *Recursively weighting pixel domain intra prediction on H.264*, in *Visual Communications and Image Processing 2003*, pp. 2035–2042, International Society for Optics and Photonics, Nov, 2003.

[46] F. Kamisli, *Intra prediction based on statistical modeling of images*, in *IEEE Visual Communications and Image Processing (VCIP)*, pp. 1–6, 2012.

[47] Y. Chen, J. Han, and K. Rose, *A recursive extrapolation approach to intra prediction in video coding*, in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1734–1738, 2013.

[48] Y. Chen, J. Han, T. Nanjundaswamy, and K. Rose, *A joint spatio-temporal filtering approach to efficient prediction in video compression*, in *IEEE Picture Coding Symposium (PCS)*, Dec, 2013.

[49] H. Everett III, *Generalized lagrange multiplier method for solving problems of optimum allocation of resources*, *Operations research* **11** (1963), no. 3 399–417.

[50] T. Wiegand and B. Girod, *Lagrange multiplier selection in hybrid video coder control*, in *IEEE International Conference on Image Processing (ICIP)*, vol. 3, pp. 542–545, 2001.

[51] V. Jacobson, R. Frederick, S. Casner, and H. Schulzrinne, *RTP: A transport protocol for real-time applications*, *RFC 1889* (Jan. 1996).

[52] M. L. Comer, *A new approach to motion compensation in spatially scalable video coding*, in *Electronic Imaging*, pp. 60770L–60770L, International Society for Optics and Photonics, 2006.

[53] R. Zhang and M. L. Comer, *Subband motion compensation for spatially scalable video coding*, in *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, vol. 6508, p. 65082v, 2007.

[54] R. Xiong, J. Xu, and F. Wu, *In-scale motion compensation for spatially scalable video coding*, *IEEE Transactions on Circuits and Systems for Video Technology* **18** (2008), no. 2 145–158.

[55] X. Li, J. Chen, K. Rapaka, and M. Karczewicz, *Generalized inter-layer residual prediction for scalable extension of HEVC*, in *IEEE International Conference on Image Processing (ICIP)*, pp. 1559–1562, 2013.

[56] R. Zhang and M. L. Comer, *Efficient inter-layer motion compensation for spatially scalable video coding*, *IEEE Transactions on Circuits and Systems for Video Technology* **18** (2008), no. 10 1325–1334.

[57] T. K. Tan, K. K. Pang, and K. N. Ngan, *A frequency scalable coding scheme employing pyramid and subband techniques*, *IEEE Transactions on Circuits and Systems for Video Technology* **4** (1994), no. 2 203–207.

[58] V. Lorcy, P. P., B. T., Z. X., S. V., and K. M., *EE2: adaptive primary transform improvement*, *Doc. JVET-D0065 ITU-T SG 16 WP3, Chengdu, CN, 15-21 Oct 2016*.

[59] G. J. Sullivan, *Efficient scalar quantization of exponential and laplacian random variables*, *IEEE Transactions on Information Theory* **42** (1996), no. 5 1365–1374.

[60] J. W. Kang and C. S. Kim, *On DCT coefficient distribution in video coding using quad-tree structured partition*, in *Asia-Pacific Signal and Information Processing Association, Annual Summit and Conference (APSIPA)*, pp. 1–4, IEEE, 2014.

[61] E. Y. Lam and J. W. Goodman, *A mathematical analysis of the DCT coefficient distributions for images*, *IEEE Transactions on Image Processing* **9** (2000), no. 10 1661–1666.

[62] G. Bjontegaard, *Calcuation of average PSNR differences between RD-curves*, *Doc. VCEG-M33 ITU-T Q6/16, Austin, TX, USA, 2-4 April 2001*.

[63] R. Wang, *Introduction to Orthogonal Transforms: With Applications in Data Processing and Analysis*. Cambridge University Press, 2012.

[64] J. Han, A. Saxena, V. Melkote, and K. Rose, *Jointly optimized spatial prediction and block transform for video and image coding*, *IEEE Transactions on Image Processing* **21** (2012), no. 4 1874–1884.

[65] F. Kamisli and J. S. Lim, *1-d transforms for the motion compensation residual*, *IEEE Transactions on Image Processing* **20** (2011), no. 4 1036–1046.

[66] B. Zeng and J. Fu, *Directional discrete cosine transformsa new framework for image coding*, *IEEE Transactions on Circuits and Systems for Video Technology* **18** (2008), no. 3 305–313.

[67] C. Yeo, Y. H. Tan, Z. Li, and S. Rahardja, *Mode-dependent fast separable klt for block-based intra coding*, in *Circuits and Systems (ISCAS), 2011 IEEE International Symposium on*, pp. 621–624, IEEE, 2011.

[68] Y. Ye and M. Karczewicz, *Improved h. 264 intra coding based on bi-directional intra prediction, directional transform, and adaptive coefficient scanning*, in *IEEE International Conference on Image Processing (ICIP)*, pp. 2116–2119, 2008.

[69] H. E. Egilmez, Y. H. Chao, A. Ortega, B. Lee, and S. Yea, *Gbst: Separable transforms based on line graphs for predictive video coding*, in *IEEE International Conference on Image Processing (ICIP)*, pp. 2375–2379, 2016.

[70] E. Pavez, A. Ortega, and D. Mukherjee, *Learning separable transforms by inverse covariance estimation*, in *IEEE International Conference on Image Processing (ICIP)*, vol. 9, 2017.

[71] H. E. Egilmez, O. G. Guleryuz, J. Ehmann, and S. Yea, *Row-column transforms: Low-complexity approximation of optimal non-separable transforms*, in *IEEE International Conference on Image Processing (ICIP)*, pp. 2385–2389, 2016.

[72] H. E. Egilmez, O. G. Guleryuz, J. Ehmann, and S. Yea, *Layered-givens transforms blabla*, in *IEEE International Conference on Image Processing (ICIP)*, pp. 2385–2389, 2016.

[73] O. G. Sezer, O. G. Guleryuz, and Y. Altunbasak, *Approximation and compression with sparse orthonormal transforms*, *IEEE Transactions on Image Processing* **24** (2015), no. 8 2328–2343.

[74] Y. Q. Shi and H. Sun, *Image and Video Compression for Multimedia Engineering: Fundamentals, Algorithms, and Standards, Second Edition*. Image Processing Series. CRC Press, 2017.

[75] J. J. Gerbrands, *On the relationships between svd, klt and pca*, *Pattern recognition* **14** (1981), no. 1-6 375–381.

[76] S. Parker, Y. Chen, J. Han, Z. Liu, D. Mukherjee, H. Su, Y. Wang, J. Bankoski, and S. Li, *On transform coding tools under development for vp10*, in *Applications of Digital Image Processing XXXIX*, vol. 9971, p. 997119, International Society for Optics and Photonics, 2016.