Distortion Estimation and Graph-based Transform for Visual Communications

by

Dong Zhang

M.A.Sc., Simon Fraser University, 2010
B.A.Sc., Simon Fraser University, 2007

Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

in the
School of Engineering Science
Faculty of Applied Science

Approval

Name:	**Dong Zhang**
Degree:	**Doctor of Philosophy (Engineering)**
Title:	***Distortion Estimation and Graph-based Transform for Visual Communications***
Examining Committee:	**Chair:** Shahram Payandeh Professor

Jie Liang
Senior Supervisor
Professor

Faisal Beg
Supervisor
Professor

Shawn Stapleton
Supervisor
Professor Emeritus

Ivan Bajic
Internal Examiner
Associate Professor

Kenneth Rose
External Examiner
Professor
Department of Electrical Engineering
University of California Santa Barbara

Date Defended: April 29, 2016

Abstract

In this thesis, we study several visual communications problems, including joint source-channel coding for single view video transmission, transmission distortion estimation for multiview video coding, and depth video coding for multiview video applications.

The first contribution in this thesis is the design and implementation of an error-resilient video conferencing system. We first develop an algorithm to estimate the decoder-side distortion in the presence of packet loss. We then design a family of very short systematic forward error correction (FEC) codes to recover lost packets. Finally, FEC codes are dynamically optimized to minimize the distortion from packet loss. The proposed scheme is demonstrated on a real-time embedded video conferencing system.

A similar joint source channel coding framework can also be applied to multiview video coding applications such as free-viewpoint TV. Therefore an algorithm is needed for the encoder to estimate the distortion of the synthesized virtual view. We first derive a graphical model to analyze how random errors in the reference depth image affect the synthesized virtual view. We then consider the case where packet loss occurs in both the encoded texture and depth images during transmission, and develop a recursive algorithm to calculate the pixel level texture and depth probability distributions in the reference views. The recursive algorithm is then integrated with the graphical model method to estimate the distortion in the synthesized view.

The graph-based transform has been extensively used for depth image coding in multiview video applications. In this thesis, we aim to develop a single graph-based transform for a class of depth signals. We first propose a 2-D first-order autoregression (2-D AR1) model and a 2-D graph to analyze depth signals with deterministic discontinuities. We show that the inverse of the biased Laplacian matrix of the proposed 2-D graph is exactly the covariance matrix of the proposed 2-D AR1 model. Therefore the optimal transform are the eigenvectors of the proposed graph Laplacian. Next, we show that similar results hold when the locations of the discontinuities are randomly distributed within a confined region. The theory in this thesis can be used to design both pre-computed and signal-dependent transforms.

Keywords: Joint source channel coding, Multiview distortion estimation, Graph transform

Acknowledgements

I would like to express my sincere gratitude to my senior supervisor Professor Jie Liang who guided me through the fumbling years towards the final completion of my PhD study. His instincts and intuitions in the field of signal processing and multimedia communication have provided me with valuable guidance and suggestions. Being a dedicated academic, Professor Liang's obsession on the theoretical aspects of research has profoundly affected my working style. Not only have I benefited from his technical expertise, but also his philosophical influence has deeply entrenched in my subconscious the right values for being a scientific researcher, which are often obscured in the industrial environment that I grew accustomed to, where profitability and materialism prevail.

I would like to thank Professor Shawn Stapleton and Professor Faisal Beg for serving on my supervisory committee. They provided valuable suggestions for my thesis proposal. I like to thank Professor Ivan Bajic for being the internal examiner. I immensely benefited from his Information Theory course, which paved solid foundations for my current research work. I also like to thank Professor Shahram Payandeh for chairing my thesis defense.

I like to thank Professor Kenneth Rose from University of California Santa Barbara for being my external examiner. His publications have greatly inspired my research work. It is an honor to have him attending my thesis defense.

I am grateful to my former manager Inderpreet Singh at Broadcom for his support on the publication of my first journal paper. Having years of experience in the technology industry, Inderpreet took a keen interest in the theoretical aspects of my work and provided opportunities and flexibilities on numerous occasions. I also like to thank Professor Gene Cheung who provided valuable comments on the graph-based transform work which lead to one of my publications.

I like to thank my friends and lab mates, Xing Wang, Yu Gao, Chongyuan Bi and Xiao Luo, who are my Friday board game and workout buddies. Because of their effusive personalities, the final chapter of my school life is filled with bustling and eventful memories.

Finally, I would like to thank my families for their loving support, compromises and sacrifices. I am particularly thankful for the family's perpetual influence on my values which kept me resolute during the years while my counterparts are engaged in more financially beneficial endeavors.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In the past two decades, rapid advances in semiconductor technologies and the wide spread of Internet paved the foundations for many resource demanding applications that are now integral parts of our lives. As an example, video streaming applications are so ubiquitous nowadays that the very concept is rarely mentioned in conversations. Since the debut of YouTube in 2005, our society witnessed a growing demand for online videos. Many companies subsequently emerged in hope to catch this bandwagon of online video frenzy. Visioning the online TV program provisioning as a potential, Netflix became widely acknowledged as a business success. Due to competitive pricing and quality, a significant amount of market share which was once enjoyed by the monopoly of traditional cable networks is now capitulated to online TV program providers. The FaceTime feature found on Apple products conveniently brings the video conferencing experience to our daily life. With the traditional means of verbal communication complemented by the visual senses, the conversing parties are immersed in the illusion that the opposing speaker is physically present. A picture is worth a thousand words, the visual aid immensely enhanced our ways of communication and the distribution of information. We grow so dependent on online video based applications that the new generation may find life missing its intrigue if they are to time travel to the past. The recent success of online video based applications only offers a glimpse of more sophisticated derivatives that are to come in the future. Realizing the importance of this field, in this thesis, we study several visual communications problems, including joint source-channel coding for single view video transmission, transmission distortion estimation for multiview video coding, and depth video coding for multiview video applications.

1.1 Background and Motivation

The introduction of video compression standards such as H.120 in 1984 and H.261 in 1988 was originally intended for the efficient storage of video data. Later as algorithms and supporting hardwares matured, video coding became widely used for media distribution on optical discs. In the late 1990s, as Internet gradually became a necessity, the concept of using Internet as the distribution medium for video started to take shape. Video conferencing is one of many applications that rely on the Internet for video transmission. However, its quality can be affected by packet loss during transmission. Therefore forward error correction (FEC) codes are usually used to mitigate the impact of packet loss. In this thesis, we aim to develop a video conferencing system that adaptively protects video packets against loss. The application platform considered is a standalone Internet video phone or tablet, where all system functionalities are implemented on an embedded system. Given the special hardware architecture, the limited computational resource, and the real-time constraint of the system, we design a system that meets the following three requirements. First, the encoder should be able to accurately estimate the decoder-side quality of a frame in the presence of packet loss. Second, a high-performance FEC code with low encoding and decoding complexity is needed to recover lost packets. Finally, a fast optimization algorithm is required to find the optimal FEC rate of each frame based on the estimated decoder-side quality.

As single view video applications gradually mature, research interests are diverted to more challenging applications such as multiview video coding. Free viewpoint television (FTV) [63] is one of the many applications of multiview video coding. FTV technology is designed to elevate the viewing experience by allowing the user to freely select the viewing angle. However, in order to support this user paradigm, a scene has to be captured by an array of cameras. Consequently, data transmission becomes an obstacle due to the sheer amount of data required by the viewer. View synthesis emerges as one possible solution. The essence of view synthesis-based FTV is that only a limited number of captured multiview videos are transmitted, and the receiver applies view synthesis algorithm to construct the virtual views between the transmitted views. Therefore the transmission bandwidth can be drastically reduced.

There are two types of view synthesis algorithms: image-based-rendering (IBR) algorithms and depth-image-based-rendering (DIBR) algorithms. The IBR algorithms render a virtual view by exploiting the disparities between the captured views [60]. On the other hand, the DIBR algorithms use the geometric information in the additional depth maps to generate the virtual view with better quality than the IBR algorithms [65]. One commonly used DIBR tool is the MPEG View Synthesis Reference Software (MPEG-VSRS) [6], which includes two algorithms. The 1D parallel algorithm is designed for scenes captured by parallel cameras with only horizontal disparity, whereas the general algorithm does not have this restriction on the cameras, as long as the camera parameters are known.

Errors in the depth and texture images can adversely affect the synthesized view [45]. Therefore the success of FTV also depends on the reliable delivery of the encoded multiview videos. Transmission impairments such as packet loss have been extensively studied in the conventional single-view video delivery. Due to the predictive nature of the encoder, losing a part of a frame can cause visual artifacts to subsequent frames. Therefore, it is important to estimate the packet-loss-induced distortion of the decoded video from the encoder. The recursive optimal per-pixel estimate (ROPE) method [89, 73] is a well-known algorithm to estimate the first two moments of the decoded pixel during encoding, from which the distortion at the decoder side can be estimated by the encoder. Such distortion estimation algorithms are useful for various error-resilient designs. For example, the coding mode and quantization parameter can be optimally decided by minimizing the video distortion at the decoder [25, 92]. Joint source-channel coding applications can also utilize the distortion estimation algorithms to find the optimal channel code rate [87].

The second contribution of this thesis is that we develop a ROPE-like scheme to estimate the decoder-side distortion of the synthesized view from the encoder. Compared to distortion estimation in single-view video transmission, estimating the distortion in the synthesized view in 3D video transmission is a lot more challenging due to the complicated warping process. However, through our theoretical derivation, we show that precise estimation of the view synthesis distortion is indeed possible.

In some multiview or 3D video applications, to facilitate view synthesis at the decoder [65, 83], the depth map sequences of the scene also need to be encoded, in addition to the conventional texture sequences. Different from natural texture images, depth images are characterized by large smooth areas within objects and sharp discontinuities at object boundaries. The most intuitive approach for depth image compression is to use a block transform such as the discrete cosine transform (DCT) followed by lossless entropy coding. However, the DCT is derived from the first-order autoregression (AR1) model with correlation coefficient approaching 1. Although it is a good model for many natural images, it is not true for some scenarios, such as depth image blocks with sharp edges or discontinuities.

To address the limitation of the DCT, a number of adaptive transforms have been proposed for depth signals, and performance improvements over DCT have been achieved. However, most of them are signal-dependent, and suffer from either expensive description cost or high complexity of the large amount of eigen-decompositions involved. Motivated by these deficiencies, our objective is to design a low-cost transform for depth image coding that is optimal for a class of signals with distinct but similar discontinuity geometries. This is the final contribution made in this thesis.

In the following chapter, we separately review the background works related to the three areas of study and then present our contributions.

Chapter 2

Review and Contributions

2.1 Real-time error resilient coding for single view video transmission

The first goal in this thesis is to design a real-time embedded video conferencing system that utilizes estimated transmission distortion information to adaptively apply FEC protection on video packets. There are three key components required by the system design. The first component is a transmission distortion estimation algorithm which accurately estimates the decoder-side quality of a frame in the presence of packet loss. The second component is a low-delay FEC code for packet loss recovery. The last component is an optimization framework which adaptively allocates FEC code rates for each video frame based on the estimated distortion. Before introducing our system design in Chapter 3, we review the previously published works related these three areas of study.

2.1.1 Review of transmission distortion estimation

In many joint source-channel coding schemes, the objective function is to minimize the expected distortion of a video frame perceived at the decoder. This distortion is usually referred to as end-to-end distortion (EED), which depends on the quantization noise introduced by the source coding, the packet loss during transmission, the error concealment algorithm, and error propagation from previous frames. Therefore how to accurately estimate the distortion is critical to the optimal designs of many encoder components and parameters.

A theoretical analysis of the EED of the complete video transmission system is developed in [61] to study and understand the interactions and tradeoffs of different system parameters. In particular, it assumes that the distortion at the decoder can be decomposed into the summation of two terms, the source-coding distortion and the packet-loss-induced transmission distortion. A model for the transmission distortion is also proposed, which involves parameters based on coding features.

In [89], by treating the reconstructed pixels at the decoder as random variables, a recursive optimal per-pixel estimate (ROPE) of the expected distortion of a reconstructed pixel is derived. The distortion involves the first and second moments of the reconstructed pixel. The expressions are obtained for both intra-coded and inter-coded macroblocks (MBs). The method is shown to be fairly accurate and can be used in, for example, optimal mode selection. In [74], some approximations are proposed to reduce the complexity of ROPE when sub-pixel motion estimation is used. A ROPE-based motion compensation optimization scheme is also proposed in [72]. In [12], another algorithm is developed to efficiently calculate the second moment of a weighted sum of correlated random variables, which is required when ROPE is used with sub-pixel motion estimation. However, the complexities of these pixel-level methods are still quite high for real-time embedded systems.

In [25], both the original and reconstructed pixels are treated as random variables. As in [61], it assumes that the distortion at the decoder is the summation of the source-coding distortion and transmission distortion. The expression of the transmission distortion is derived for both intra-coded and inter-coded MBs. The average transmission distortion of the frame is thus found to be a linear combination of that of the previous frame and the average difference between the two frames.

In [92], the transmission distortion estimation method in [25] is used in the mode decision optimization when sending H.264/AVC bit stream over error-prone networks. In [35], the problem of EED estimation for the transmission of pre-encoded videos is studied. The reconstructed frame difference required in [25] is calculated in the transform domain, which is faster than computing in the pixel domain after complete decoding of the pre-encoded videos.

A frame-level transmission distortion model is developed in [26], where the transmission distortion caused by packet losses is treated as the impulse response of a linear time-invariant system. It is found that for MPEG-4 codec with intra and inter MBs, the transmission distortion can be approximated by an exponential model, whose decaying factor and initial distortion can be easily estimated.

An important factor that is not considered by [25, 92, 35, 26] is the unconstrained intra prediction (UIP) mode in P slices of H.264/AVC standard, where intra-coded MBs in P slices can use neighboring inter-coded MBs as reference for intra prediction. In most applications of H.264/AVC, UIP is enabled to get improved coding efficiency [18, 48]. Our system has very strict bit budget requirement, therefore UIP mode is enabled. In this case, if the neighboring MBs are affected by transmission error, the error will be propagated to the intra-coded MBs as well. Therefore, more mismatches will result if this is not considered in the EED estimation.

In [49], it is observed that when UIP mode is enabled, the exponential decaying model developed in [26] is no longer valid for H.264/AVC, where the transmission distortion could even increase, instead of always decreasing as in MPEG-4. It is also shown in [49] that by inserting some intra MBs with constrained intra prediction, *i.e.*, no pixels from other MBs are used for intra prediction, the transmission distortion of H.264/AVC codec will exhibit the exponential decaying behavior. However, using constrained intra prediction increases the bit rate, which is undesired in many applications.

Another frame-level recursive model is derived in [68] to estimate the transmission distortion without using motion vector (MV) and coding mode information of each MB. This allows the estimation of the relationship between the average channel distortion and the average intra rate and packet loss rate before actually encoding the frame. It also considers the impacts of sub-pixel motion estimation, UIP and deblocking filter. However, the accuracy of the model relies on selecting the right parameters. Therefore it is not clear whether the proposed framework is applicable to an arbitrary video sequence in practice.

In [55], a better approximation of the error-concealment cross-correlation term in the model of [68] is developed, which leads to improved accuracy. In [54], the model in [55] is generalized to support multiple reference frames. However, only constrained intra prediction is used in [55, 54].

In this thesis, we develop a fast distortion estimation method that meets the real-time constraint of our embedded system. Our method operates at MB level, and considers the impact of the UIP and all the intra prediction modes in H.264/AVC. We also develop various techniques to reduce the complexity without sacrificing too much accuracy.

Some preliminary results of the proposed distortion estimation method were reported in [86], where a simple approximation was used to represent the impacts of different intra prediction modes. In this thesis, we derive the closed-form expression for each mode, and demonstrate the performance of the method using more results.

2.1.2 Review of FEC-based erasure protection

FEC is an effective method to correct errors or erasures during data transmission. In this thesis, we are interested in FEC-based erasure protection since the system considered in this thesis transmits packets over Internet.

Some examples of erasure protection codes include Reed Solomon (RS) Code [37], LT code [39], and Raptor Code [59]. RS code is a systematic code with excellent error correction capability. It can have very short block length. However, since it is a non-binary code defined over finite fields, its complexity is relatively high, and there are constraints on the values of input and output symbol sizes, making it difficult to freely adjust the rate of the RS code [44, 10]. Therefore it is not suitable for resource-stringent real-time software and hardware systems.

Low density parity check (LDPC) code [23] was originally designed for error correction, and was later applied to mitigate packet loss. Simple XOR operations can be used for encoding and decoding. The optimization of the LDPC code structure has been studied extensively [52, 16]. The drawback of LDPC code is that very long code block length is needed to get good performance, which is not desired for low delay applications.

Fountain codes [44, 10] are a class of erasure codes where a large number of encoding symbols can be generated from a given source symbol block, and the source can be recovered from any subset of the encoding symbols with size slightly larger than the number of source symbols. Fountain codes are also called rateless codes since they do not exhibit a fixed code rate.

A simple example of fountain codes is the random linear fountain code, which, however, suffers from high encoding and decoding complexities. The LT code was the first practical example of fountain codes [39]. It has good performance as well as lower complexity, which scales as $K \ln K$, where K is the input size. This is achieved by using a sparse random encoding matrix with a robust Soliton degree distribution and a fast and iterative message-passing decoding algorithm, which can be considered as a restricted Gaussian elimination method.

To further reduce the complexity of LT codes, the Raptor code is developed in [59], where a LT code with very low average degree is used, leaving a small fraction of the LT code inputs unconnected to the output. These inputs are then protected by another simple code, such as irregular LDPC code. As a result, linear encoding and decoding complexity can be achieved.

Recently, the RaptorQ code is developed to further reduce the redundancy of Raptor code [1], but its decoding complexity is also increased [47]. Therefore it is suitable for data delivery where fast decoding is not a critical requirement.

The rateless codes described above are generally non-systematic. It is also possible to construct systematic rateless codes. One example is given in [59, 40] by first forcing a part of output as the input symbols, and then finding the corresponding state symbols that generate these outputs. However, its complexity is higher than the non-systematic code.

In [8], systematic code is obtained by multiplying the input with the inverse of the first part of the generator matrix. However, this approach destroys the degree distribution of the remaining output nodes. As a result, maximum likelihood decoding has to be used instead of the simple message passing method.

In [21], systematic outputs are obtained by choosing the first part of the generator matrix to be the identity matrix. In addition, each check node has the same degree.

Most applications of rateless codes use very long block sizes in order to get better performance. However, for real-time video conferencing, short block sizes are desired to reduce the delay. In this thesis, we design a family of rateless codes that run across different slices in a frame, by treating each MB slice as a source symbol. Therefore it has very short block size. For example, there are only 30 slices when the frame size is 720×480. To reduce the complexity, a systematic code similar to that in [21] is used. The difference from [21] is that our code has roughly the same degree for each input node, in order to provide the same protection to each slice.

2.1.3 Review of optimization of channel code rate

Given the estimated end-to-end distortion of each frame and the FEC scheme, the next task is to optimally allocate the FEC code rates to different frames in a group under the bit rate constraint, such that the total distortion can be minimized, where the FEC code rate is defined as K/N, with K and N being the number of source symbols and number of coded symbols, respectively.

In order to perform this optimization, a closed-form rate distortion model is needed to map a channel loss rate to a loss rate after FEC decoding given a specific code structure and code rate. Proietti [20] provides a method to derive a closed-form solution for the LDPC code. It was shown that the bit erasure probability of an individual code construction approaches the ensemble average of bit erasure probabilities of all possible codes constructed as code block length increases. It is also true that the ensemble average of all code constructions approaches cycle free performance as code block increases. Consequently the focus is on deriving a closed-form solution for the ensemble average to approximate individual code performance, and asymptotic analysis is used during the derivation process. It should be noted that the closed-form solution derived is based on the assumption that the code block length is long. The derived solution is recursive and thus also has complexity issues during calculation.

Since FEC codes with very short block sizes are used in our video conferencing system, the solution from asymptotic analysis can no longer be used as the actual expression. At very short block length, not many algorithms have been proposed on how the code structure should be designed. A simple but effective way to develop a rate distortion model for FEC code is by simulation. The constructed FEC codes are simulated using dummy data for different loss rates and different code rates. The results can be pre-calculated and then be used by the distortion optimization process in real time operations.

2.1.4 Our contributions

We have implemented the entire distortion estimation, FEC rate optimization, and FEC encoding/decoding on a real-time embedded video conferencing system. Here we briefly discuss the architecture of the system, which poses some constraints for the design of our system.

The video conferencing system consists of a dual-core main processor running at 1GHz with 512MB memory and a video co-processor running at 250MHz with 128MB memory. The main processor and co-processor operate independently with their own real time operating systems. A hardware interface and the corresponding interface drivers are responsible for data communications between the two processors. Such two-processor systems are also quite common in the industry.

The video co-processor has dedicated H.264/AVC hardware acceleration blocks shared by encoding and decoding. Any software control and intervention mechanisms on H.264/AVC hardware blocks need to be implemented on the video co-processor. In addition, the video coprocessor has a vector of arithmetic logic units (ALUs) that can perform paralleled arithmetic operations much faster than the scalar main processor. Therefore, our distortion estimator is implemented on the video co-processor. Thus the entire estimation algorithm needs to be accommodated by 250MHz processing power and fitted into a 128MB memory footprint, in addition to the regular video encoding and decoding. Note that during real time two-way video conferencing, video encoding and decoding are happening simultaneously. Consequently computation resources are shared between various hardware, software modules and operating system on the video co-processor.

On the other hand, the FEC encoding, FEC decoding and FEC rate allocation optimization are handled by the main processor. During video encoding, the compressed frames and distortion information are generated by the video co-processor and transferred to the main processor through the interface. The FEC rate optimization algorithm is performed using video frame distortion information. The FEC encoder is then instructed by the optimization process to encode each video frame using the optimal channel code rate. The packetization and transmission conforms to the RTP standard. We also designed a transport layer protocol extension which complies with the RFC3984 standard, with considerations of interoperability and backward compatibilities.

After implementing the proposed distortion estimation and adaptive FEC scheme, the entire system can support real-time two-way video conference with resolution up to 1024×576 pixels, 30 frames per second (fps) and 4 megabits per second (Mbps).

2.2 Transmission Distortion Estimation for Synthesized Virtual View

A joint source channel coding framework for multiview video applications such as FTV requires a transmission distortion estimation algorithm for the synthesized virtual views. Therefore, the second contribution made in this thesis is an accurate distortion estimation algorithm for synthesized virtual views when reference texture and depth images are affected by packet loss. Different from single view video, the synthesized virtual video is composed of geometrically warped images from the reference views. Consequently, the distortion in the virtual view contains contributions from the reference views and the estimation task becomes more challenging.

There are two sources for the distortions in the virtual view: distortion caused by source coding and distortion caused by channel errors. We separately review the existing distortion estimation algorithms which target these two types of errors.

2.2.1 Review of source-coding-caused synthesis distortion estimation

In [62], a generalized view interpolation scheme is studied, where the synthesized view is the sum of the filtered left and right warped texture images. The depth maps are assumed to be corrupted by noise signals with known distributions. A Fourier-domain analysis is used to find the optimal filters that minimize the distortion in the synthesized view. It is shown that under certain conditions the commonly used distance-based view merging method is near optimal.

In [32], the difference between the reference texture image and its horizontally shifted version is used to derive a global parameter to estimate the synthesized view distortion. The estimated synthesized view distortion is then employed to optimally select the skipping mode for coding the depth maps. Further improvements are made in [33], where the correlation between a reference frame and its shifted version is studied. A closed-form model utilizing the correlation coefficient is proposed. The distortion model proposed in [33] has been used in [50] for AVC/HEVC-compatible 3D video codecs.

A joint video/depth rate allocation scheme is proposed in [38]. A distortion estimation method based on motion warping analysis [56] is used to find the optimal quantization parameter for the texture and depth images that minimizes the view synthesis distortion. In [88, 17], the authors also employ the results from motion warping analysis to model the synthesis distortion to optimally code the depth images.

In [15], a closed-form cubic synthesis distortion model is used in the bit allocation problem. An extra dimension is added to the optimization process to choose the best reference view for synthesis. A closed-form synthesis distortion model is also developed in [77] for the bit allocation problem to address the complexity issue in search-based allocation algorithms [38].

In [19], the structural similarity (SSIM) metric [69] is integrated into the synthesis distortion to optimize the codec for better subjective quality.

In [22], the quantization distortion from lossy coding is assumed to be a zero mean white noise signal [77, 78], and the distortion in the synthesized view is decomposed as the sum of texture image coding distortion and depth image coding distortion. It is shown that the distortion in reference texture images directly contribute to the synthesis distortion. On the other hand, in order to find the distortion contribution from depth image errors, the warped texture images are first partitioned using gradient maps into spatial invariant and spatial variant regions. Next, the distortion contribution is found by performing spectral and spatial analyses on the spatial invariant regions and spatial variant regions respectively.

In [93], depth images are partitioned into two types of regions: color texture regions in the depth map corresponding to areas in the texture image with rich textures, and color smooth regions corresponding to smooth regions. Therefore, synthesis distortion can be found by summing the distortions from different regions. For each of the two types of regions, approximations are made such that synthesis distortion can be linearly related to the position errors caused by depth errors. Law of large numbers is used extensively in the model approximation and simplification.

In [67], analysis on motion warping [56] is utilized to relate distortions in the depth images to the synthesis distortion. To better fit the assumption imposed by the motion warping analysis, Quarter-tree sub-division (QTSD) is used to divide the depth image into regions with similar depth levels. The distortion for each region is individually estimated and summed up to give the distortion in the synthesized view.

In [90], distortion in the synthesized view is related to the correlation between a texture image and its translation. Additionally, rounding effect on the disparity error is also incorporated into the distortion model. The distortion estimation algorithm is then used for optimal coding of the depth images.

Simple distortion estimation techniques have been used in rate-distortion-optimized rate control algorithms. In [29], it is found that the synthesized view quality is approximately linear to the quantization parameter used to code the texture and depth images. In [57], the synthesized view distortion is treated as the sum of the distortion caused by texture coding and distortion caused by depth coding. A fast approximation is made such that the distortions caused by the texture and depth images are modeled as being inversely proportional to the respective coding rate for the texture and depth images.

2.2.2 Review of transmission-error-caused synthesis distortion estimation

The methods discussed above only consider the impact of source coding on the synthesized view, but not the impact of transmission error. In [13], a quadratic model is proposed to relate the disparity errors caused by packet loss in the depth maps to the distortion contribution in the synthesized view. In [43, 42, 41], the quadratic-model-based distortion estimation is used at the encoder for reference frame selection and quantization parameter optimization when coding the depth and texture images. In [41], the view synthesis at the decoder is designed such that reliability factors are introduced and related to the estimated synthesis distortion to give higher synthesis quality. The overall transmission distortion estimation framework used in them is a block-based recursive approach, which is similar to the one used in [87].

2.2.3 Our contributions

In this thesis, we first develop a probabilistic graphical model method to accurately calculate the synthesized view distortion when the reference depth maps contain random errors with known distributions. The graph can precisely represent the warping competition operation in the DIBR algorithms [65], where multiple depth pixels can be warped to the same destination location, and the pixel nearest to the camera is selected to be the winning pixel. The graph method can be used for both the general and 1D parallel algorithms in MPEG-VSRS. To the best of our knowledge, there has been no previous algorithm to capture the random interaction between pixels and the warping competition operation during the view synthesis process. The probabilistic model can serve as a foundation for designing fast approximations.

We then consider the setup where the texture and depth images are independently encoded by video codecs such as H.264/AVC. The coded texture and depth bitstreams are assumed to be affected by packet loss during transmission. In this case, to calculate the distortion in the synthesized view caused by the packet loss, we develop a recursive optimal distribution estimation (RODE) method to estimate from the encoder the entire probability distribution of each reconstructed reference texture or depth pixel at the decoder. The RODE method is more general than the ROPE algorithm [89, 73], which only estimates the first two moments of each pixel. Consequently RODE enables the calculation of other statistical properties. For example, if some models are developed to describe certain statistical quantities associated with a decoded pixel, the results from RODE can serve as the ground truth for the analysis. Again, to the best of our knowledge, a method to precisely calculate the pixel-level probability distribution due to packet loss has not been studied.

Finally, we integrate the RODE and the graphical model-based methods to estimate the distortion of the synthesized view when there are packet losses during the transmission of the reference texture and depth maps.

In single-view video transmission, pixel-based transmission distortion estimation algorithms like ROPE [89, 73] is shown to be optimal and more accurate than the block-based methods like the one used in [43, 42, 41]. Consequently, as a generalization of the ROPE, the proposed RODE method inherits its estimation accuracy and is expected to outperform the block-based methods. The quadratic model used in [13, 43, 42, 41] is not a direct and accurate representation of the actual distortion in the synthesized view. Furthermore, the warping competition operation in the DIBR algorithms is ignored in them. By comparison, our graphical model-based estimation algorithm considers the warping competition operation and is able to precisely calculate the synthesized view distortion. This will be confirmed by the experimental results.

2.3 Graph-based Transform for Depth Images

As mentioned earlier, some multiview video based applications rely on the accurate and efficient representation of depth images. In this thesis, our goal is to design a single transform that is optimal for a class of the depth image blocks with similar edges. Before presenting the theories behind our design in Chapter 5, we review a few signal dependent transform designs that are closely related to our work.

2.3.1 Review of adaptive transforms

In image and video coding, intra prediction is widely used to remove spatial redundancies among adjacent image blocks. However, the intra prediction process creates residuals that are not necessarily optimal for DCT. Consequently, some alternative transform designs have been explored for intra prediction residuals.

In [79], a separable directional DCT is introduced, where in the first step, 1D DCTs of various sizes are applied to the intra prediction residue in the prediction direction to avoid filtering across potential edges. In the second step, another 1D DCT is applied to the transform coefficients from the first step. Since variable-length DCTs are used in the first step, DC corrections are required to prevent the second DCT from generating unnecessary nonzero coefficients. In [71], the directional DCT is generalized to lapped transform [66].

In [75], various directional KLTs are used instead of the DCT to encode the residues of Intra predictions in video coding. However, the use of KLT leads to higher complexity than DCT. In [24], directional KLTs are derived for the Intra prediction residues in video coding. A first-order Gauss-Markov sequence is used to model the residual signals. Derivations reveal that the KLTs for the Intra prediction residues can be approximated by asymmetric discrete sine transform (ADST). The proposed directional transforms are then implemented for the horizontal and vertical Intra prediction modes in H.264/AVC codec, which results in considerable coding gain. A similar development can be found in [76]. By assuming a directional image correlation model, the KLT for the residual signals from various prediction angles can be approximated by separately applying DCT and DST.

Graph-based transform is another adaptive transform that has been applied in the compression of depth images, which can be considered as piecewise smooth [58, 14, 34, 31, 30]. The idea is to represent the depth image by a graph such that each pixel is a vertex and pixels within the same object boundary are connected in the graph. The optimal decorrelating transform can be obtained from the eigen-decomposition of the Laplacian matrix of the graph. To use the graph-based transform in image coding, the discontinuity information of each block should be transmitted to the decoder, and eigen-decomposition has to be employed for each block by both the encoder and the decoder. As a result, the graph-based transform suffers from expensive description cost and high algorithm complexity.

In [31], a multi-resolution graph-based transform is proposed for depth image coding. The edge locations are encoded in the original resolution while graph transforms are applied to the downsampled depth image. Two types of graph-based transforms are used. One uses disconnected graphs for blocks with obvious discontinuities and the other uses connected weighted graphs if the discontinuity is weak. To derive the optimal transform for the connected graph, first, a AR1 model corresponding to a line graph is proposed to represent a specific discontinuity, and the edge weights are found by inverting the covariance matrix. Next the optimal 1D analysis is assumed for 2-D signals such that a 4-connected graph can be constructed to transform image blocks with a weak discontinuity. It is found empirically that only a small portion of all graph transforms is frequently used. Therefore, these common transforms can be pre-computed at the encoder and decoder to reduce the algorithm complexity. During encoding, the best transform is selected based on a rate proxy function. The complexity can be further reduced by utilizing lifting transform to approximate the graph transform [11, 36]. Further improvements on intra coding are suggested in [30], based on the same multi-resolution coding framework.

There are also efforts on finding a single optimal transform for a collection of signals. In [51], three types of graph templates with various edge densities are introduced. Since graph Laplacian can be considered as the inverse of the covariance matrix, the edge weights that define the graph Laplacian are found by convex optimization given the actual signal covariance matrix. Similar to the graph template transform design, in [53] several typical graph-based transforms are introduced as offline templates. The graph templates are constructed by edges in two different directions. By assuming the resulting graph Laplacian matrix to be a Gaussian Markov Random Field (GMRF) signal model, the edge weight ratio that determines the transform can be approximated by collecting signal statistics. During coding, each image block is assigned to one of the predefined templates based on its principal gradient. The index for the selected transform template is then encoded as side information for reconstruction at the decoder side.

Besides the coding applications mentioned above, graph theory also serves as a powerful analysis utility. In [80] the graph-based analysis is used to prove the optimality of the separable 2-D DCT for smooth signals.

2.3.2 Our contributions

In this thesis, we first consider the deterministic case with a known discontinuity location in each row. We propose a 2-D first-order autoregression (2-D AR1) model and a 2-D graph for this type of signals. We then derive the closed-form expression of the inverse of a biased Laplacian matrix of the proposed 2-D graph. We also show that the inverse is exactly the covariance matrix of the proposed 2-D AR1 model. Therefore the optimal transform for the signal are the eigenvectors of the proposed graph Laplacian. Next, we show that similar results still hold in the random case, where the locations of the discontinuities in different rows are randomly distributed within a confined region, and we derive the corresponding optimal 2-D graph Laplacian.

The theory developed in this thesis enables pre-computing the graph-based transforms for different classes of random signals, thereby avoiding the cost for description and eigen-decomposition. It can also be used to obtain signal-dependent transforms with much lower complexity than existing methods, because each graph transform can be used in many similar blocks. This reduces the number of transforms that have to be computed dynamically, making the scheme suitable for real-time applications. Finally, depth image coding experiments demonstrate that our methods can achieve similar performance to the state-of-the-art method, but our complexity can be reduced by 86-267 times.

Graph is usually used together with the Gaussian Markov Random Field (GMRF) to represent images [80, 51, 53]. However, extracting physical meanings from GMRF can be difficult. Therefore sometimes it is more convenient to link a more concrete model such as the AR process to a graph for analysis. To the best of our knowledge, our theory is the first to relate a 2-D AR process to a graph.

The theoretical analyses in this thesis are more general and consistent than those in [31, 51, 53]. They can provide useful insights for the design of other graph-based transforms for more complicated signal models.

2.4 Publications

During the course of completing this thesis, we have produced the following publications.

Journal papers:

- D. Zhang, and J. Liang, Graph-based Transform for 2-D Piecewise Smooth Signals with Random Discontinuity Locations, in preparation for *IEEE Trans. Image Process.*
- D. Zhang, and J. Liang, View Synthesis Distortion Estimation with Graphical Model and Recursive Calculation of Probability Distribution, *IEEE Trans. on Circuits and Systems for Video Technology*, Vol. 25, No. 5, pp. 827-840, May 2015
- D. Zhang, J. Liang, and I. Singh, Fast Transmission Distortion Estimation and Adaptive Error Protection for H.264/AVC-based Embedded Video Conferencing Systems, *Signal Processing: Image Communication*, Volume 28, Issue 5, May 2013, pp. 417-429.

Conference papers:

- D. Zhang, and J. Liang, Graph-based Transform for 2D Piecewise Smooth Signals with Random Discontinuities, *Data Compression Conference (DCC)*, Utah, USA, Mar. 2016, to appear.
- D. Zhang, J. Liang, and X. Gong, Distortion Estimation for Two-Step View Synthesis, *Proc. Picture Coding Symposium (PCS)*, San Jose, CA, Dec. 2013, pp. 93-96.

- D. Zhang, J. Liang, and I. Singh, End-to-End Distortion Estimation for H.264 with Unconstrained Intra Prediction, *Proc. 2012 IEEE Inter. Conf. Image Processing (ICIP)*, Orlando, FL, USA, Oct. 2012, pp. 697-700.

Chapter 3

Fast Transmission Distortion Estimation and Adaptive Error Protection for H.264/AVC-based Embedded Video Conferencing Systems

3.1 Introduction

In this chapter, we propose a real-time embedded video conferencing system that utilizes estimated transmission distortion information to adaptively apply FEC protection on video packets. Given the special hardware architecture, the limited computational resource, and the real-time constraint of the system, this chapter proposes a system design that incorporates the following three key components. First, we design a transmission distortion estimation algorithm such that the encoder is able to accurately estimate the decoder-side quality of a frame in the presence of packet loss. Second, a high-performance FEC code with low encoding and decoding complexity is proposed to recover lost packets. Finally, a fast optimization algorithm is designed to find the optimal FEC rate of each frame. The entire integrated system can support real-time two-way video conference with resolution up to 1024×576 pixels, 30 frames per second (fps) and 4 megabits per second (Mbps).

The work presented in this chapter is a continuation and improvements on our previous work [81], which details the design of the preliminary version of the transmission distortion estimation algorithm and FEC code. The transmission distortion estimation algorithm presented in this chapter is similar to the one in [81]. However, in [81], the unconstrained Intra prediction is accounted for using simple averaging operations. In this thesis, we improve over the previous design by mathematically deriving the closed-form approximations for all the unconstrained Intra prediction modes. The FEC code design used in this thesis is the same as the one in [81]. To improve FEC code efficiency, an optimized rate allocation framework is added in this thesis, which is a new contribution to [81]. Another difference between the new and old designs is that the algorithms in [81] are implemented on an older generation of the hardware. Consequently, the supported resolution and bitrate are slightly lower than the specifications reported in this thesis.

This chapter is organized as follows. In Sec. 3.2 we present the distortion estimation algorithm. In Sec. 3.3, a practical short length FEC code is introduced. The FEC rate optimization method is described in Sec. 3.4. The performance of the system is reported in Sec. 3.5. Finally, Sec. 3.6 concludes the chapter. The closed-form approximations for various unconstrained intra prediction modes are presented in Appendix 3.A.

3.2 Fast MB-Level Recursive Distortion Estimation

3.2.1 Recursive Formulas for MB Distortion

We briefly summarize the transmission distortion estimation algorithm in [81]. The distortion D_f of a video frame f at the decoder as a function of the packet loss rate can be expressed as

$$D_f(P(f)) = (1 - P(f)) \sum_{m=1}^{M} D_{MB}^{R}(f,m) + P(f) \sum_{m=1}^{M} D_{MB}^{L}(f,m), \tag{3.1}$$

where $P(f)$ is the packet loss probability of the frame, M is the total number of MB's in the frame, $D_{MB}^{R}(f,m)$ is the distortion when the m-th MB is received, and $D_{MB}^{L}(f,m)$ is the distortion when it is lost.

Our goal is to find the recursive expressions of $D_{MB}^{R}(f,m)$ and $D_{MB}^{L}(f,m)$. We first look at $D_{MB}^{L}(f,m)$. It is derived in our earlier work [81] that $D_{MB}^{L}(f,m) = D_{MB}^{S}(f,m) + RFD(f, f-1, m) + D_{MB}^{C}(f-1,m)$, where $D_{MB}^{S}(f,m)$ is the source coding distortion, $RFD(f, f-1, m)$ is encoder reconstructed frame difference and $D_{MB}^{C}(f-1,m)$ is channel distortion caused by packet loss [81].

Next, we study $D_{MB}^{R}(f,m)$ in (3.1), which depends on the type of the MB. In this chapter, we consider real-time H.264/AVC applications with only I-frames and P-frames. The encoder uses arbitrary MB boundary slice mode, and each slice is encoded into one Network Adaptation Layer (NAL) unit. There are four MB types in our system: I-MB in I-slices, I-MB in P-slices, P-MB in P-slices, and SKIP-MB. Among them, I-MBs in I-slices are not affected by errors in other MBs, since intra predictions are not allowed to cross slice boundary and all MBs within I-slices are intra-coded. All other MB types can be affected by the errors in other MBs, including I-MBs in P-slices when the unconstrained intra prediction is used.

Our derivation in [81] shows that if an I-MB in an I-slice is received, its distortion is only caused by quantization, *i.e.*, $D^R_{MB}(f,m) = D^S_{MB}(f,m)$. When other types of MBs are received, $D^R_{MB}(f,m) = D^S_{MB}(f,m) + D^C_{MB}(f-1,\boldsymbol{m'})$, where $D^C_{MB}(f-1,\boldsymbol{m'})$ is the channel distortion for MB $\boldsymbol{m'}$ which is used for either Inter or Intra prediction. The detailed expression for $D^C_{MB}(f-1,\boldsymbol{m'})$ for a P-MB is discussed in [81]. In this chapter, we will find a closed-form recursive expression for $D^C_{MB}(f-1,\boldsymbol{m'})$ when $\boldsymbol{m'}$ is an I-MB in a P-slice.

Now we are ready to describe the treatment on unconstrained intra prediction (UIP), which is the new contribution in this thesis. Since UIP is enabled, the I-MB in the P-slice is affected by error propagation. This was not considered in [74, 92, 35, 26]. In this case, the reconstructed encoder prediction residual $\hat{e}(f,m,i)$ and decoder prediction residual $\tilde{e}(f,m,i)$ are identical. Therefore, we have the following relationship.

$$\begin{aligned} D^C_{MB}(f,m) &= E\{[\hat{F}(f,m,i) - \tilde{F}(f,m,i)]^2\} \\ &= E[\hat{e}(f,m,i) + \hat{F}(f,\boldsymbol{m'},i) - \tilde{e}(f,m,i) - \tilde{F}(f,\boldsymbol{m'},i)]^2 \\ &= E[\hat{F}(f,\boldsymbol{m'},i) - \tilde{F}(f,\boldsymbol{m'},i)]^2 = D^C_{MB}(f,\boldsymbol{m'}), \end{aligned} \tag{3.2}$$

$$D^C_{MB}(f,m) = E[\hat{F}(f,\boldsymbol{m'},i) - \tilde{F}(f,\boldsymbol{m'},i)]^2 = D^C_{MB}(f,\boldsymbol{m'}), \tag{3.3}$$

where $\hat{F}(f,\boldsymbol{m'},i)$ and $\tilde{F}(f,\boldsymbol{m'},i)$ are the intra-predictions from neighboring MBs $\boldsymbol{m'}$ at encoder and decoder. Therefore $D^C_{MB}(f,m)$ can be obtained recursively from the channel distortions of its neighboring MBs $\boldsymbol{m'}$. Note that $D^C_{MB}(f,\boldsymbol{m'})$ in (3.3) should be calculated by assuming MBs $\boldsymbol{m'}$ are received, since they come from the same received slice.

In [81, 86], a simple approximation is used to estimate $D^C_{MB}(f,\boldsymbol{m'})$. That is, when the intra prediction of the current MB only involves the top or left neighboring MB, $D^C_{MB}(f,\boldsymbol{m'})$ is set to be that of the neighboring MB. When both the top and left MBs are used, the average of their channel distortions is used. In this chapter, we consider each intra prediction mode separately, and derive the closed-form expression of the weighting parameter for each neighboring MB.

Suppose pixel $F(f,m_c,i)$ from current MB m_c is intra-predicted using pixel $F(f,m_l,j)$ from the left MB m_l and pixel $F(f,m_t,k)$ from the top MB m_t as $F(f,m_c,i) = 0.5F(f,m_l,j) + 0.5F(f,m_t,k)$, the pixel-level encoder and decoder mismatch for pixel i in the current block can be written as

$$\begin{aligned} D^C_p(f,m_c,i) &= E\left[\frac{1}{2}\left(\hat{F}(f,m_l,j) + \hat{F}(f,m_t,k)\right) - \frac{1}{2}\left(\tilde{F}(f,m_l,j) + \tilde{F}(f,m_t,k)\right)\right]^2 \\ &= \frac{1}{4}E[(\hat{F}(f,m_l,j) - \tilde{F}(f,m_l,j))]^2 + \frac{1}{4}E[(\hat{F}(f,m_t,k) - \tilde{F}(f,m_t,k))]^2 \\ &+ \frac{1}{2}E\left[(\hat{F}(f,m_l,j) - \tilde{F}(f,m_l,j))(\hat{F}(f,m_t,k) - \tilde{F}(f,m_t,k))\right]. \end{aligned} \tag{3.4}$$

To simplify this, we assume that nearby pixels in a natural image are very similar, thus the mismatches between the encoder and decoder for two such pixels are equal. That is,

$$E[(\hat{F}(f,m_l,j)-\tilde{F}(f,m_l,j))-(\hat{F}(f,m_t,k)-\tilde{F}(f,m_t,k))]^2=0. \tag{3.5}$$

Consequently,

$$\begin{aligned}&E[(\hat{F}(f,m_l,j)-\tilde{F}(f,m_l,j))(\hat{F}(f,m_t,k)-\tilde{F}(f,m_t,k))]\\&=\frac{1}{2}\Big(E[(\hat{F}(f,m_l,j)-\tilde{F}(f,m_l,j))]^2+E[(\hat{F}(f,m_t,k)-\tilde{F}(f,m_t,k))]^2\Big)\\&=\frac{1}{2}(D_p^C(f,m_l,j)+D_p^C(f,m_t,k)).\end{aligned} \tag{3.6}$$

Plugging this into (3.4), we get

$$D_p^C(f,m_c,i)=\frac{1}{2}\left(D_p^C(f,m_l,j)+D_p^C(f,m_t,k)\right), \tag{3.7}$$

which shows that the distortion for one pixel in the current intra-coded MB has contribution from one pixel in the left MB and one pixel in the top MB.

To find the distortion for the entire intra-coded MB we add the distortion for each pixel. Even though the resulting summation will have distortion contribution from different pixels, we do not distinguish them individually due to the fact that we lack pixel level statistics. Rather, we add up their contribution along with their scaling factor as long as they come from the same MB. For example, using the same example, suppose every pixel in the current MB is predicted using the same expression, i.e. $F(f,m_c,i)=\frac{1}{2}(F(f,m_l,j)+F(f,m_t,k))$ for $i=0...255$, we have

$$\begin{aligned}D_{MB}^C(f,m_c)&=256\times\frac{1}{2}\frac{D_{MB}^C(f,m_l)}{256}+256\times\frac{1}{2}\frac{D_{MB}^C(f,m_t)}{256}\\&=\frac{1}{2}\left(D_{MB}^C(f,m_l)+D_{MB}^C(f,m_t)\right).\end{aligned} \tag{3.8}$$

Note that the weighting factor used on the neighboring MB distortion is found by the contribution from the summation process divided by the total number of pixels in the MB. It is quite similar to how inter-coded MB distortion is propagated from the MB in the previous frame.

In [12], a similar assumption to Eq. (3.5) is also made, but it is used to simplify the second moment calculations. Here we use Eq. (3.5) to represent the cross-correlation terms by the available expectation values, such that the distortion contributions from the cross-correlation terms at pixel level can be transformed into the distortion contributions at block level.

A computer program can be constructed to perform equation expansion and find the weighting parameters for each intra prediction mode. The results are given in Appendix 3.A.

Since intra prediction is essentially a pixel level filtering operation, our method to convert distortion at pixel level to MB level can be applied to other filtering operations such

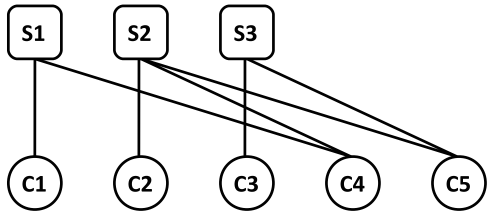


Figure 3.1: Tanner graph of a binary systematic FEC code with $N = 5$, $K = 3$, and $D = 2$.

as deblocking filter. However, in this chapter, our embedded system does not have the computation resource to perform deblocking filter. Consequently deblocking filter is not used in our system.

3.2.2 Recursive Formulas for Channel Distortion

It can be seen from the derivations above that $D_{MB}(f, m)$ depends on the channel distortion $D^C_{MB}(f, m)$. $D^C_{MB}(f, m)$ can be recursively calculated by considering the different coding modes at the MB level. The derivation details can be found in our previous work [81]. Similarly, UIP also causes distortion propagation in the channel distortion term. Consequently, instead of using the averaging operations for intra coded MBs in P-slices as described in [81], we use Eq. (3.3) for each Intra prediction mode.

3.3 Design of Low-Complexity FEC

In this section, we introduce a family of LDPC-like systematic FEC codes with different rates and the following properties: XOR-based simple encoding and decoding operations, linear encoding/decoding complexity, and low delay. Our design can be viewed as a special form of rateless code. The FEC code design used in this thesis is the same as the one from our previous work [81]. Therefore, we briefly summarize the code design.

3.3.1 Code Construction

Let $\boldsymbol{S}$ be an $K \times 1$ vector containing the source symbols, and $\boldsymbol{C}$ an $N \times 1$ vector representing the coded symbols, with $N > K$. We are interested in FEC codes that only involve simple XOR operations in encoding and decoding. Therefore the $N \times K$ generator matrix $\boldsymbol{G}$ has binary entries. We then have

$$\boldsymbol{C} = \boldsymbol{G}\boldsymbol{S}. \tag{3.9}$$

Since systematic codes are desired, the first K rows of $\boldsymbol{G}$ form an identity matrix. The last $N - K$ rows yield the redundant or parity symbols.

The relationship between the source symbols and the coded symbols in (3.9) can be represented by a Tanner graph, as shown in Figure 3.1 (a), where the degree of a node is the number of edges connected to it. In this chapter, to get the same protection to each symbol, each input symbol is designed to have roughly the same degree D, *i.e.*, each column in $\boldsymbol{G}$ has about D ones.

Since the first K rows of $\boldsymbol{G}$ form an identity matrix, each column in its last $N-K$ rows can have $D-1$ ones, so the total number of possible codes is $\binom{N-K}{D-1}^K$. This is a very large number, and it is very difficult to find the decoding failure rate of each case. However, simulations using some small numbers show that there is not much difference between the decoding failure probabilities of different codes. Therefore we randomly generate a code with constant source node degree, as long as its last $N-K$ rows have full rank.

In our system, a group of FEC codes with different rates is produced offline and their generator matrices are saved in the encoder and the decoder. An optimization scheme described later is used to select the optimal FEC code for each frame.

In the conventional LDPC design, avoiding short cycles in the code is a major concern. Since our code is very short and is systematic, many short cycles can be easily broken up, as will be shown later.

3.3.2 Encoding and Decoding

The encoding and decoding of the proposed FEC code are straightforward. The parity symbols can be obtained by applying the XOR operations to the inputs based on the generator matrix $\boldsymbol{G}$. Clearly the encoding complexity is linear. It scales with the number of parity symbols as well as the degree of these symbols. The decoding process is similar to solving linear systems. The details for the encoding and decoding are presented in [81].

3.4 Adaptive FEC Rate Allocation Optimization

Given the estimation of the end-to-end distortion of each frame and the performance of the FEC code, we can optimize the allocation of FEC protection to different frames. This is a new contribution to our previous work [81].

The performance of the FEC code can be represented by its final packet loss rate after the FEC decoding. This is denoted by $P_f(K_f, N_f, \rho_f)$, where K_f is the number of slices in frame f, N_f is the number of coded packets after FEC encoding, and ρ_f is the packet loss rate for this frame before FEC decoding. $P_f(K_f, N_f, \rho_f)$ can be obtained from the lookup table generated during the FEC code design. Therefore the estimated distortion for frame f after FEC decoding can be written as $D_f(P_f(K_f, N_f, \rho_f))$.

In this chapter, we minimize the total distortions of M frames in a group by allocating the FEC redundancy level of each frame, subject to a total redundancy constraint. The problem can be formulated as

$$\begin{aligned} \mathrm{argmin}_{N_f, f=1,\ldots,M} \sum_{f=1}^{M} D_f(P_f(K_f, N_f, \rho_f)), \\ \text{s.t.} \quad \sum_{f=1}^{M} (N_f - K_f) = r_M, \end{aligned} \tag{3.10}$$

where r_M is the desired number of parity packets for this group of frames.

For video conferencing applications, low end-to-end delay is critical. Therefore M is chosen to be 2 in this chapter. That is, two frames are gathered before the optimization process takes place. In this case, there are only $r_M + 1$ combinations of the redundancy allocation for the two frames.

It can be seen from Eq. (3.10) that during the optimization, the distortion for each frame needs to be evaluated under a number of loss probabilities after applying different channel code rates. However, the channel distortion term D_{MB}^{C} depends on the loss probability [81]. Therefore, each time the loss probability changes, the entire distortion estimation algorithm has to be re-run. This will take too much time. In addition, the main processor and the video coprocessor in our embedded system are two asynchronous devices; hence it is not easy for the main processor to interrupt the video coprocessor and to re-estimate the distortion using a different configuration, yet still meets the real-time constraint. Therefore a fast approximation method is needed.

Note that with the same source coding configuration, more redundancy introduced by the channel coding will lead to lower frame distortion, as long as it does not cause network jam. In addition, the redundant FEC packets cannot reduce the distortion below the quantization distortion. Therefore the decoder-side distortion of a frame is upper bounded by $D_f(\rho_f)$, the distortion induced by the channel loss probability without any FEC coding, and lower bounded by the quantization distortion $D_f^S(\rho_f)$. Moreover, experimental results show that when the distortion of each frame is measured by the sum of squared error (SSE), it is approximately linearly related to the loss probability. Therefore instead of running the distortion estimation multiple times, we only need to run it once using the packet loss probability ρ_f (without FEC), and then apply the following linear interpolation method to find $\hat{D}_f(P_f(K_f, N_f, \rho_f))$, the fast approximation of $D_f(P_f(K_f, N_f, \rho_f))$,

$$\hat{D}_f(P_f(K_f, N_f, \rho_f)) = D_f^S(f) + \frac{D_f(\rho_f) - D_f^S(f)}{\rho_f} P_f(K_f, N_f, \rho_f), \tag{3.11}$$

where $D_f^S(f)$ is the quantization distortion for frame f. The accuracy of the linear interpolation will be demonstrated in Sec. 3.5.

Since the optimization in Eq. (3.10) only involves two frames, there are only $r_M + 1$ possible solutions. Also, r_M is usually less than 30. Therefore a simple exhaustive search can be used to find the optimal FEC rate allocation.

3.5 Experimental Results

In this section, the performance of the proposed fast MB-level end-to-end distortion estimation, low-complexity FEC code, and the adaptive FEC rate optimization are presented.

3.5.1 Performance of the Fast MB-level Distortion Estimation

We first present the performance of the proposed fast MB-level distortion estimation algorithm. The encoder encodes the raw video sequence first with the distortion estimator running in parallel to produce the estimation result for each frame. Constant bit rate (CBR) is used to avoid packet loss caused by the sudden increase of packets in some scenarios. The encoded video clip is then fed to the decoder for decoding. During the decoding process, NAL units are randomly dropped according to uniform distribution and the specified loss probability, and the lost MBs are concealed by copying the co-located MBs from the previous frame. The decoded video frames are compared to the original sequence to obtain the sum of squared error (SSE). It should be noted that bursty loss can be tackled using more sophisticated loss models.

The decodings are performed 200 times, and the average SSE is used as the EED for each frame, which is then compared with the estimated distortion calculated by the encoder. The following Average SSE Mismatch Ratio (ASMR) is used to measure the accuracy of the encoder-estimated distortion:

$$ASMR = \frac{1}{N}\sum_{f=1}^{N}\left|\frac{D_{enc}(f) - D_{dec}(f)}{D_{dec}(f)}\right|, \tag{3.12}$$

where N is the number of frames in this video sequence, $D_{enc}(f)$ is the SSE for frame f estimated by the encoder, and $D_{dec}(f)$ is the SSE observed by the decoder for frame f averaged over 200 experiments.

In order to fully investigate the response of the distortion estimator to various dynamic behaviors of video coding, different test sequences with different behaviors are chosen, including CIF (352×288) sequences Akiyo, Coastguard, Container, Foreman, News, Silent and Stefan, and 525P (720×480) sequences City, Football, Suzie and Train. Each CIF sequence has 300 frames and is encoded at the typical rate of 512kbps of our system, with slice sizes of about 300 bytes. Each 525P sequence has 250 frames and is encoded at the typical rate of 2Mbps, with about 1000 bytes per slice. All sequences are evaluated with group of picture (GOP) size of 30 frames and 60 frames respectively. The first frame in each GOP is coded as an I-frame and the rest are coded as P-frames. A single reference frame is used for P-frames. The results are summarized in Table 3.1. Some results of 525P sequences with 1Mbps and 600 bytes/slice can be found in our previous work [86].

Columns 2 to 5 of Table 3.1 show the ASMR for all sequences with different GOP sizes and packet loss rates, by considering unconstrained intra prediction (UIP) and using MV

Table 3.1: ASMR (%) of all testing sequences with different configurations.

Sequence	GOP30 5% loss	GOP30 10% loss	GOP60 5% loss	GOP60 10% loss	GOP30 5% loss	GOP30 10% loss	GOP30 5% loss	GOP30 10% loss
	UIP Considered				UIP Unconsidered		UIP Considered	
	MV Average				MV Average		MC Partitions	
Akiyo	10.63	16.71	19.60	29.22	9.74	15.41	11.36	17.81
Coastguard	7.24	18.42	7.84	23.40	5.49	12.30	8.72	20.57
Container	8.14	6.74	11.54	5.60	5.74	4.45	8.09	6.94
Foreman	9.10	11.58	11.19	13.73	16.28	17.54	9.08	11.73
News	9.82	11.54	15.41	15.34	23.19	26.17	10.12	13.03
Silent	6.47	7.33	9.24	10.23	21.03	23.51	6.10	8.32
Stefan	8.02	25.56	10.81	30.30	9.18	12.21	12.44	30.09
City	9.12	11.37	15.73	16.87	12.99	9.74	8.15	16.19
Football	11.56	14.59	13.03	14.11	44.16	46.98	13.09	17.62
Suzie	11.12	12.48	14.85	16.91	19.10	23.83	9.17	10.02
Train	13.20	19.42	14.68	14.83	7.06	11.78	15.70	20.95
Average	9.49	14.16	13.08	17.32	15.81	18.54	10.18	15.75
Standard Deviation	2.01	5.55	3.38	7.54	11.27	11.54	2.74	6.73

average to propagate distortion from inter-coded MB, which shows that our EED estimation is quite accurate in most cases.

Since our method operates at MB level, and the algorithm design is also constrained by the hardware architecture and the real-time requirement of our video conferencing system, it is difficult to precisely compare our results with other methods in the same testing configuration. Nevertheless, some rough comparisons can still be obtained. For example, the ASMR results of the improved pixel-level ROPE method in [74] are between 12.05% and 16.80% with 100kbps, 5% intra ratio and loss probability of 5%. The ASMR results of the compressed-domain method in [35] are between 6.8% and 9.3% with 300 or 600 kbps and $3 \sim 10\%$ loss rate. Our average results in the first two columns of Table 3.1 with GOP size of 30 are 9.49% and 14.16% (GOP size is not reported in [74, 35]). This is acceptable in comparison with [74, 35], since our method is MB-based and can run in real time.

Columns 6 and 7 in Table 3.1 shows the results without considering UIP, but MV average is still used. In this case, the average ASMR is increased by 67% and 31% at 5% and 10% loss rate respectively. This demonstrates the importance of accounting for UIP in the distortion estimation algorithms. In particular, for sequences with more intra-coded MBs such as Silent and Football, ignoring the effect of UIP can deteriorate the ASMR by $3 \sim 4$ times. We also notice from columns 6 and 7 in Table 3.1 that the standard deviation on ASMR is much higher when UIP is not considered. This is due to the variation in the amount of Intra-coded MBs in each sequence. The more Intra-coded MBs are involved, the higher the estimation error is when UIP is not considered.

On the other hand, when UIP is not considered, some test sequences actually shows lower ASMR, due to the fact that our fast estimation method lacks pixel level statistics. It was observed in our experiments that the distortion estimation of inter-coded MBs tends to overestimate the result. When UIP is not considered, the intra-coded MBs in P frames are assumed to carry no distortion propagation; hence the SSE for each frame decreases,

Table 3.2: The average PSNR (dB) at 5% and 10% packet loss rate for different FEC configurations

Sequence	5% Loss				10% Loss			
	No FEC	Fixed FEC	Optimized FEC	Gain of Optimization	No FEC	Fixed FEC	Fixed FEC	Gain of Optimization
Coastguard	25.59	28.66	28.87	0.21	23.72	26.80	27.13	0.33
Container	34.21	35.75	35.85	0.10	32.45	34.98	35.16	0.18
Foreman	26.16	30.19	30.37	0.18	23.84	27.68	27.99	0.31
News	31.10	35.93	36.23	0.30	28.55	33.12	33.43	0.31
Silent	30.59	33.66	33.80	0.14	28.63	31.83	32.09	0.26
Stefan	20.95	25.72	26.12	0.40	18.91	22.62	23.13	0.51
City	26.86	32.42	33.02	0.60	24.37	28.65	29.00	0.35
Football	23.48	27.60	27.99	0.39	21.22	24.70	25.03	0.33
Suzie	32.39	36.73	36.98	0.25	29.65	33.91	34.05	0.14
Train	20.15	24.38	24.55	0.17	17.97	21.49	21.79	0.30
Average	27.15	31.10	31.38	0.28	24.93	28.58	28.88	0.30

which damps the overestimation and could yield a smaller ASMR. However, in most cases, it is beneficial to consider UIP.

The last two columns in Table 3.1 are obtained by considering UIP, but the distortion propagation is obtained for each of the Inter partitions rather than using the MV average. Compared to Columns 2 and 3 where the average of MVs is used, there is no improvement in the ASMR. Therefore using the average MV is preferred, as it can significant speed up the algorithm without sacrificing the estimation accuracy.

As an example, the decoder-side and encoder-estimated PSNRs of the News sequence with 10% loss rate and bit rate of 512kbps are plotted in Fig. 3.2, which shows that the estimated distortion agrees quite well with the actual one. The average SSE and PSNR mismatches are 11.54% and $0.19dB$, respectively, and the worst PSNR mismatch is $1.75dB$. The figure also includes the result without considering UIP. In this case, the SSE and PSNR mismatches increases to 26.17% and $1.37dB$, respectively, and the worst PSNR mismatch is $3.95dB$. Note that its mismatch grows much faster towards the end of the GOP.

3.5.2 Performance of the Low-Complexity FEC Codes

The detailed performance analysis for our FEC code can be found in [81]. In summary, our experiments confirm that the FEC design has linear encoding and decoding complexity. The performance of the FEC code is tested with a specific short input length at various loss rates and parity redundancy levels. We also investigate the asymptotic performance of the FEC code design. While keeping the same parity redundancy level, as the number of input symbols increases, the loss protection capability increases.

3.5.3 Performance of the Adaptive FEC Allocation Optimization

We first show the accuracy of the linear interpolation method in Eq. (3.11) of the redundancy optimization. The Foreman sequence is used in this example. For each frame, the linear interpolation is performed with a step size of 1% for the loss probability from 2% to 15%. For example, when the loss probability is 10%, the distortions for the loss probability from 1% to 9% are linearly interpolated from the distortion at 10% and the quantization distortion. The interpolation error is defined as

$$error = \frac{|\hat{D}_f(P_f(K_f, N_f, \rho_f)) - D_f(P_f(K_f, N_f, \rho_f))|}{D_f(P_f(K_f, N_f, \rho_f))} \tag{3.13}$$

The average interpolation error over all loss probabilities for each frame is shown in Fig. 3.3. The overall interpolation error is found by averaging the interpolation errors of all frames, which is 5.64%. Therefore it is reasonable to employ the linear interpolation during channel code rate allocation.

Next, we study the performance of our proposed short FEC code in both the fixed and optimized rate allocation schemes. Unconstrained intra prediction is enabled in all cases. A GOP size of 30 is used for all the test sequences. 5% and 10% packet loss rates are used. A 30% redundancy level is used for the fixed FEC allocation scheme, where there is no distortion estimation. For the optimized FEC allocation scheme, the redundancy level for each frame in a group of two frames is adaptively determined with the overall redundancy not exceeding 30%. We can observe from Table 3.2 that the fixed FEC allocation scheme has an average gain of 3.95 dB and 3.65 dB respectively over the unprotected video stream at the two loss rates, with a maximum of 5.56 dB in the City sequence. The optimized FEC allocation scheme can further improve the results by $0.28dB$ and $0.30dB$ respectively.

Fig. 3.4 and Fig. 3.5 show the PSNR difference between the optimized FEC scheme and the fixed scheme for Stefan sequence at 5% loss probability and Silent sequence at 10% loss probability. The average improvement is $0.40dB$ and $0.26dB$ respectively. In most frames the optimized scheme has higher PSNR than the fixed method, and many frames have much higher improvements than the average (up to 2dB and 0.8dB respectively), leading to improved visual quality. Due to the various approximations employed in the distortion estimation algorithm and the small group size in FEC rate allocation, the optimized method can occasionally have slightly lower results.

Note that the optimized FEC rate allocation could have larger gain over the fixed FEC scheme at other redundancy levels and optimization window sizes. For example, if there is no real-time constraint, the window size can be increased to get better FEC rate allocation performance, making the proposed distortion estimation and FEC schemes useful for many other applications.

3.6 Summary

This chapter presents various algorithms for error-resilient real-time H.264/AVC-based video conferencing over Internet, including a fast MB-level end-to-end distortion estimation algorithm, a family of very short FEC codes, and a fast adaptive FEC rate allocation scheme. Several techniques are developed to meet the real-time requirement and the hardware constraints of the embedded system, without sacrificing too much of the performance. The proposed scheme has been successfully implemented on a real-time embedded video conferencing system with resolution up to 1024×576 pixels, 30 frames per second (fps) and 4 megabits per second (Mbps).

3.A Formulas for Channel Distortion Term

In this appendix, we give the expressions of the channel distortion term in different intra prediction modes. Due to the complexity involved in manually deriving channel distortion expressions for each intra prediction mode, a program is constructed to accomplish this task. The essence of the program is to perform equation expansion and simplification, similar to Eq. (3.4).

In the first step, the program places pixels used by intra prediction into two duplicated lists with their coordinates and coefficients. The coordinates and coefficients for each pixel in each prediction mode is defined by the H.264/AVC standard. The two lists of pixels are then multiplied. This is the same as expanding the square in Eq. (3.4). Two categories of pixels are produced after the multiplication process. One category consists of self-multiplied pixels. Another consists of the cross-multiplications of different pixels.

In the next step, a term conversion process is performed. From Eq. (3.6) we see that two cross-multiplied pixels can be simplified into a summation of two self-multiplied pixels. The program uses this to convert every cross-multiplied pixel pairs into self-multiplied pixel summations.

In the final step, since only self-multiplied pixels are left, the program gathers all the self-multiplied pixels by adding up their coefficients. As long as the pixels originate from the same MB, their coefficients are summed up to produce the weighting factor for that particular MB. The final output from the program is the weighting factors for all the neighboring MB used during the intra prediction process. The results for Intra16×16 mode and Intra4×4 mode are presented below. We omit Intra8×8 prediction since it is the same as Intra4×4 case except for the block size.

3.A.1 Intra16×16 Mode

Let m_l, m_t and m_{tl} denote the left, top, and top left neighboring MB respectively. Due to the non-linear behavior of clipping operations, we assume that in the plane prediction mode, the prediction values are within the clipping range. The following expressions can be found by the program for each of the intra prediction mode in Intra16×16 mode.

Vertical:

$$D_{MB}^{C}(f, m) = D_{MB}^{C}(f, m_t). \tag{3.14}$$

Horizontal:

$$D^C_{MB}(f,m) = D^C_{MB}(f,m_l). \tag{3.15}$$

DC-Top Available:

$$D^C_{MB}(f,m) = D^C_{MB}(f,m_t). \tag{3.16}$$

DC-Left Available:

$$D^C_{MB}(f,m) = D^C_{MB}(f,m_l). \tag{3.17}$$

DC-Top Left Available:

$$D^C_{MB}(f,m) = \frac{D^C_{MB}(f,m_t)}{2} + \frac{D^C_{MB}(f,m_l)}{2}. \tag{3.18}$$

DC-None Available:

$$D^C_{MB}(f,m) = 0. \tag{3.19}$$

Plane:

$$D^C_{MB}(f,m) = \frac{130.5D^C_{MB}(f,m_t)}{256} + \frac{130.5D^C_{MB}(f,m_l)}{256} - \frac{5D^C_{MB}(f,m_{tl})}{256} \tag{3.20}$$

3.A.2 Intra4×4 Mode

The same process also applies to Intra 4×4 prediction mode. The only difference is that the neighboring block size is reduced to 4×4. Let b_l, b_t, b_{tl} and b_{tr} denote the 4×4 block on the left, top, top left and top right, respectively. During the estimation process, each of the 4×4 blocks in the MB will use the following expressions, depending on its prediction mode. If the neighboring block resides in a 16×16 MB, the neighboring 4×4 block level distortion is weighted down by a factor of 16 from the MB level distortion. This way distortion calculation are always propagated using MB level distortion values that are previously calculated. The distortion for the MB is found by summing the distortions from each of 4×4 blocks. D^C_b in the following expressions denotes distortion for a 4×4 block.

Vertical:

$$D^C_b(f,b) = D^C_b(f,b_t). \tag{3.21}$$

Horizontal:

$$D^C_b(f,b) = D^C_b(f,b_l). \tag{3.22}$$

DC-Top Available:

$$D^C_b(f,b) = D^C_b(f,b_t). \tag{3.23}$$

DC-Left Available:

$$D^C_b(f,b) = D^C_b(f,b_l). \tag{3.24}$$

DC-Top Left Available:

$$D^C_b(f,b) = \frac{D^C_b(f,b_l)}{2} + \frac{D^C_b(f,b_t)}{2}. \tag{3.25}$$

DC-None Available:

$$D^C_b(f,b) = 0. \tag{3.26}$$

Diagonal Down Left:

$$D_b^C(f,b) = \frac{6.25D_b^C(f,b_t)}{16} + \frac{9.75D_b^C(f,b_{tr})}{16}. \tag{3.27}$$

Diagonal Down Right:

$$D_b^C(f,b) = \frac{6.25D_b^C(f,b_l)}{16} + \frac{3.5D_b^C(f,b_{tl})}{16} + \frac{6.25D_b^C(f,b_t)}{16}. \tag{3.28}$$

Vertical Right:

$$D_b^C(f,b) = \frac{2.25D_b^C(f,b_l)}{16} + \frac{2.75D_b^C(f,b_{tl})}{16} + \frac{11D_b^C(f,b_t)}{16}. \tag{3.29}$$

Horizontal Down:

$$D_b^C(f,b) = \frac{11D_b^C(f,b_l)}{16} + \frac{2.75D_b^C(f,b_{tl})}{16} + \frac{2.25D_b^C(f,b_t)}{16}. \tag{3.30}$$

Vertical Left:

$$D_b^C(f,b) = \frac{11D_b^C(f,b_t)}{16} + \frac{5D_b^C(f,b_{tr})}{16}. \tag{3.31}$$

Horizontal Up:

$$D_b^C(f,b) = D_b^C(f,b_l). \tag{3.32}$$

3.B Figures for the Experimental Results

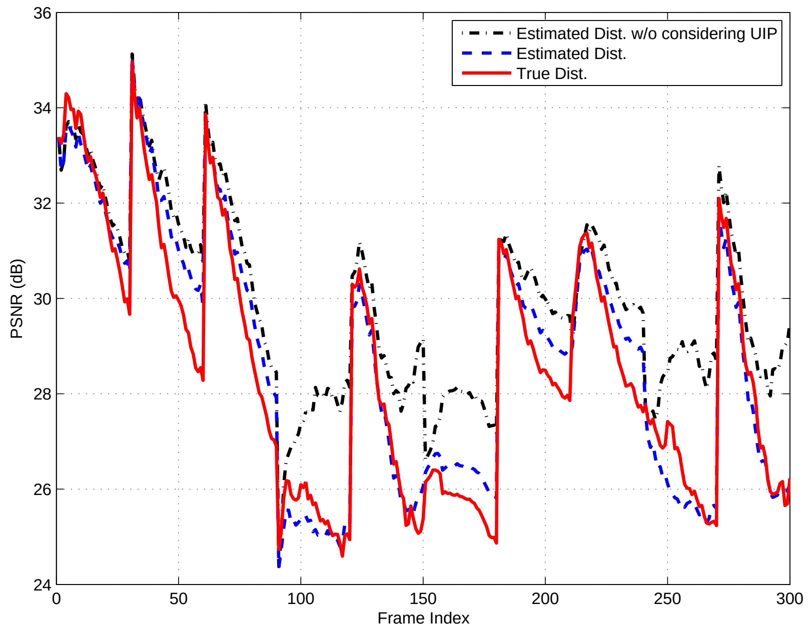


Figure 3.2: Estimated and the actual PSNR: News sequence with 512kbps and 10% packet loss. GOP is 30 frames.

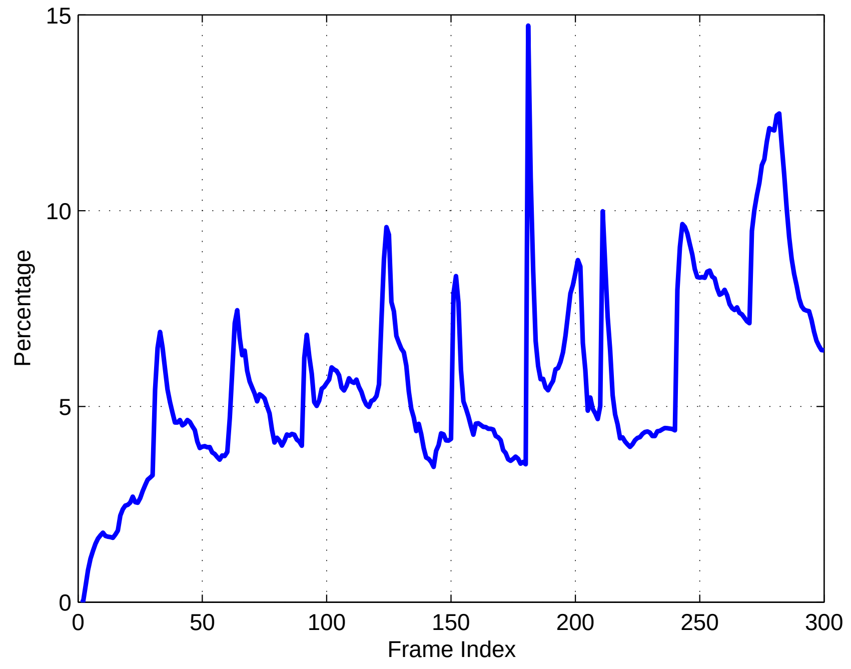


Figure 3.3: The average linear interpolation error for each frame in Foreman sequence.

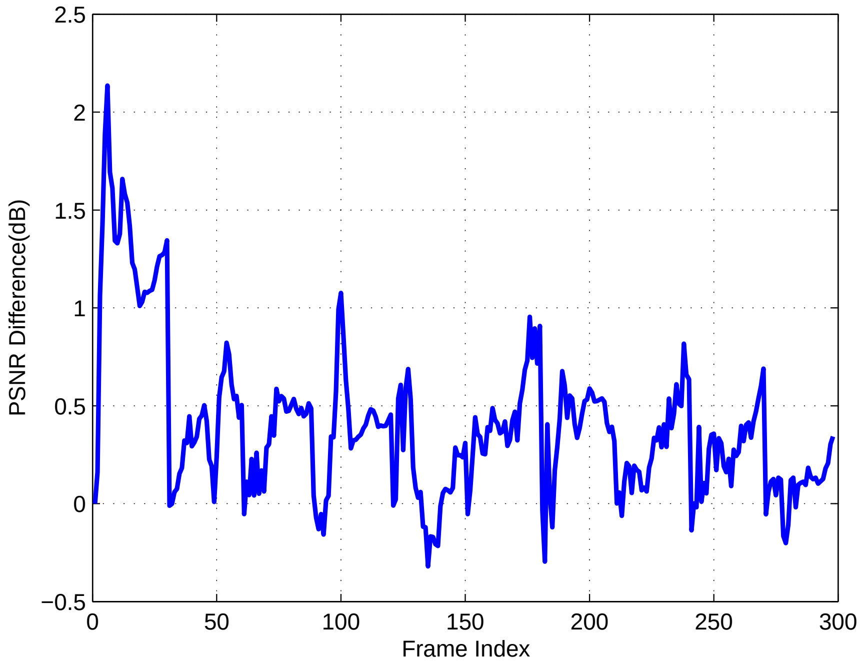


Figure 3.4: The PSNR difference between the optimized and the fixed schemes for sequence Stefan. 5% loss rate. Average difference: $0.40dB$.

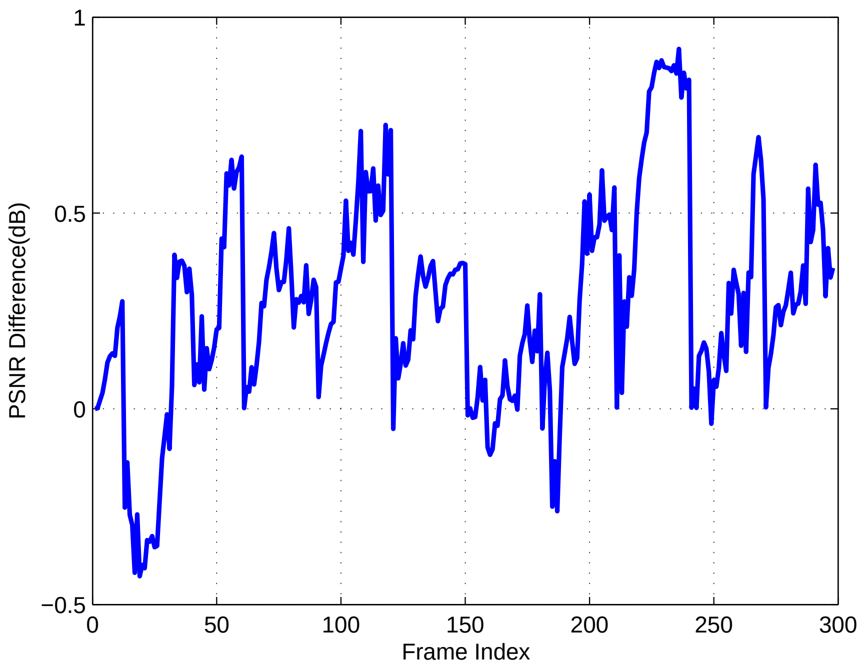


Figure 3.5: The PSNR difference between the optimized and the fixed schemes for sequence Silent. 10% loss rate. Average difference: $0.26dB$.

Chapter 4

View Synthesis Distortion Estimation With a Graphical Model and Recursive Calculation of Probability Distribution

4.1 Introduction

In this chapter, we aim to develop a ROPE-like scheme to estimate the decoder-side distortion of the synthesized view from the encoder. We first develop a probabilistic graphical model method to accurately calculate the synthesized view distortion when the reference depth maps contain random errors with known distributions. The graph can precisely represent the warping competition operation in the DIBR algorithms. We then consider the setup where the texture and depth images are independently encoded by video codecs such as H.264/AVC. The coded texture and depth bitstreams are assumed to be affected by packet loss during transmission. In this case, to calculate the distortion in the synthesized view caused by the packet loss, we develop a recursive optimal distribution estimation (RODE) method to estimate from the encoder the entire probability distribution of each reconstructed reference texture or depth pixel at the decoder. Finally, we integrate the RODE and the graphical model-based methods to estimate the distortion of the synthesized view when there are packet losses during the transmission of the reference texture and depth maps. The estimation theories presented in chapter are published in [85, 83].

The rest of the chapter is organized as follows. Sec. 4.2 reviews the general and 1D parallel view synthesis algorithms in MPEG-VSRS. Sec. 4.3 describes the overall framework for distortion estimation. In Sec. 4.4, depth maps are assumed to be corrupted by random noises with known distributions, and a graphical model-based method is designed to estimate distortions in the synthesized view. Both the general and 1D parallel view synthesis algorithms are considered. Sec. 4.5 introduces the recursive optimal distribution estimation (RODE) method, which is then combined with the graphical model method. Sec. 4.6 presents the experimental results, and Sec. 4.7 concludes the discussion.

Notations: In this chapter, we use V with appropriate index and subscript to represent a vertex in a graph, which corresponds to a pixel. The texture and depth levels of the vertex are denoted by T and D respectively, with the same index and subscript. We also represent any quantity under the influence of random errors by a tilde symbol $\tilde{}$ over the quantity, whereas the same quantity without error does not have the tilde.

4.2 View synthesis algorithm overview

In this section, we briefly review the main operations involved in the two DIBR-based view synthesis algorithms in MPEG-VSRS. Further details can be found in [65, 27].

4.2.1 The General View Synthesis Algorithm

The general view synthesis algorithm in MPEG-VSRS includes four main steps. First, for each reference view, the forward and reverse warping homography matrices for each of the 256 depth levels (0 to 255) are calculated using the camera parameters and four pairs of matching coordinates.

The second step is called depth forward warping, where each of the left and right reference depth maps D_0 and D_1 is used to warp a depth map for the virtual view, denoted as D_{w_0} and D_{w_1}. At the beginning, the depth level of each virtual view pixel is set to 0. After that, each reference pixel is processed in a raster scan order and is warped to a location in the virtual view according to the following equation [27]

$$\begin{bmatrix} x_{dst} \\ y_{dst} \\ 1 \end{bmatrix} = \mathbf{H}(d) \begin{bmatrix} x_{src} \\ y_{src} \\ 1 \end{bmatrix}, \tag{4.1}$$

where (x_{src}, y_{src}) and (x_{dst}, y_{dst}) are the locations in the source and destination view respectively (the latter is rounded to integer), and $\mathbf{H}(d)$ is the forward warping homography matrix found in the first step, which depends on the depth level d of the source pixel.

After finding the destination in the virtual view, the algorithm next decides if the depth level of the reference pixel can be copied to the virtual pixel location. An important fact is that multiple pixels in the reference depth map can be warped to the same location in the virtual view. This is called the warping competition problem [65]. In this case, the algorithm only updates the depth level of a virtual pixel when the depth level of a new reference pixel is greater than or equal to the current depth level of the virtual pixel. This is because a larger depth level is closer to the camera and therefore is less likely to be occluded. It is also possible that some pixels in the virtual view could not be warped by any pixel in the reference view, leading to some holes in the warped depth map. To improve the quality, median filtering is applied to reduce the noise, and hole dilation is used to avoid synthesis artifacts [27].

The third step is texture reverse warping, where each of the synthesized depth map is used to find a texture image of the virtual view. For each pixel in the virtual view, by using its warped depth level, the corresponding reverse warping homography matrix, and Eq. (4.1), a corresponding pixel in the reference texture view T_0 or T_1 can be located, and its texture value is copied to the synthesized texture image.

Finally, the warped texture images from both references are merged. Let $T_{w_0}(i)$ and $T_{w_1}(i)$ denote the i-th texture pixel in the virtual image warped from left and right reference views respectively, $T_s(i)$ the i-th texture pixel in the final synthesized virtual view, and $a \in [0, 1]$ a scaling factor that depends on the location of the virtual view. If both $T_{w_0}(i)$ and $T_{w_1}(i)$ are available, the merged texture is their linear combination $T_s(i) = (1 - a)T_{w_0}(i) + aT_{w_1}(i)$. If only one is available, it is used directly as $T_s(i)$. Otherwise, $T_s(i)$ is a hole and will be filled by an inpainting method [9, 64].

4.2.2 The 1D parallel View Synthesis Algorithm

When the cameras capturing the scene only have horizontal shifts, the simpler 1D parallel view synthesis algorithm can be used, which only requires a single forward warping process. In this case, the warped location of a reference pixel in the virtual view only has a horizontal displacement called disparity, which can be computed from the depth level of the reference pixel, the camera focal length, and the distance of the two views.

Given the disparity and virtual pixel location, the reference texture value can be copied to the virtual pixel without using reverse texture warping. However, warping competition can still happen, *i.e.*, different reference pixels can be warped to the same location. Therefore each virtual pixel still needs to keep track of its depth level. The texture value from a new reference pixel can only replace the current texture value of a virtual pixel when the depth level of the new reference pixel is greater than the current depth level associated with the virtual pixel.

The warping process is performed for both the left and right reference views. Merging the two warped texture images and the handling of holes are similar to the general algorithm.

4.3 Overall Framework of View Synthesis Distortion Estimation

In this section, we present the general framework to estimate the distortions in the synthesized view. The framework is applicable to both the general and 1D parallel view synthesis algorithms. We assume that both the reference texture and depth images contain random errors. Therefore, each synthesized pixel at the receiver is treated as a random variable.

Let random variable $\tilde{T}_s(i)$ denote the texture value of the i-th pixel in the synthesized view at the receiver when reference texture and depth images contain random errors. Using the mean squared error (MSE) as distortion metric, the overall expected distortion (ED) of the i-th texture pixel in the synthesized view is commonly defined as (cf. Eq. (1) in [89]):

$$\begin{aligned} ED(i) &= E\left\{\left(T_s(i) - \tilde{T}_s(i)\right)^2\right\} \\ &= T_s(i)^2 - 2T_s(i)E\left\{\tilde{T}_s(i)\right\} + E\left\{\tilde{T}_s(i)^2\right\}, \end{aligned} \tag{4.2}$$

where $T_s(i)$ is the correctly synthesized texture pixel and is deterministic. Therefore we need to find the first and second moments of the synthesized texture value $\tilde{T}_s(i)$.

Next, we consider the merging process shared by the two DIBR algorithms. Let $\tilde{T}_{w_m}(i)$ $(m = 0, 1)$ denote the i-th texture pixel in the virtual view warped from the left and right texture images respectively in the presence of errors. Its probability mass function (PMF) is denoted as $P_{\tilde{T}_{w_m}(i)}(t)$, where t is the possible texture value of the pixel. Further, we use $P_{\tilde{T}_{w_m}(i)}(\phi)$ to represent the probability of $\tilde{T}_{w_m}(i)$ being a hole. After the warped texture images are merged, each merged pixel belongs to one of four cases depending on whether $\tilde{T}_{w_0}(i)$ and $\tilde{T}_{w_1}(i)$ are holes or not. Therefore, assuming that the virtual views warped from the two views are independent, the first and second moments of $\tilde{T}_s(i)$ can be expressed as follows.

$$\begin{aligned} &E\left\{\tilde{T}_s(i)^k\right\} = \sum_{t_0 \neq \phi} \sum_{t_1 \neq \phi} ((1-a)t_0 + at_1)^k P_{\tilde{T}_{w_0}(i)}(t_0) P_{\tilde{T}_{w_1}(i)}(t_1) \\ &+ \sum_{t_0 \neq \phi} t_0^k P_{\tilde{T}_{w_0}(i)}(t_0) P_{\tilde{T}_{w_1}(i)}(\phi) + \sum_{t_1 \neq \phi} t_1^k P_{\tilde{T}_{w_1}(i)}(t_1) P_{\tilde{T}_{w_0}(i)}(\phi) \\ &+ P_{\tilde{T}_{w_0}(i)}(\phi) P_{\tilde{T}_{w_1}(i)}(\phi) E\left\{\tilde{T}_s^I(i)^k\right\}, \quad k = 1, 2, \end{aligned} \tag{4.3}$$

where t_0 and t_1 represent possible texture values of $\tilde{T}_{w_0}(i)$ and $\tilde{T}_{w_1}(i)$ respectively, $\tilde{T}_s^I(i)$ is the value of the i-th texture pixel if it is inpainted due to being a hole pixel.

After expanding the first term and grouping similar terms, the two moments can be rewritten as

$$\begin{aligned} &E\left\{\tilde{T}_s(i)\right\} = (1-a)(1 - P_{\tilde{T}_{w_1}(i)}(\phi)) E_{\backslash\phi}\left\{\tilde{T}_{w_0}(i)\right\} \\ &+ a(1 - P_{\tilde{T}_{w_0}(i)}(\phi)) E_{\backslash\phi}\left\{\tilde{T}_{w_1}(i)\right\} \\ &+ P_{\tilde{T}_{w_1}(i)}(\phi) E_{\backslash\phi}\left\{\tilde{T}_{w_0}(i)\right\} + P_{\tilde{T}_{w_0}(i)}(\phi) E_{\backslash\phi}\left\{\tilde{T}_{w_1}(i)\right\} \\ &+ P_{\tilde{T}_{w_0}(i)}(\phi) P_{\tilde{T}_{w_1}(i)}(\phi) E\left\{\tilde{T}_s^I(i)\right\}, \end{aligned} \tag{4.4}$$

$$\begin{aligned}
&E\left\{\tilde{T}_s(i)^2\right\} = (1-a)^2(1-P_{\tilde{T}_{w_1}(i)}(\phi))E_{\setminus\phi}\left\{\tilde{T}_{w_0}(i)^2\right\} \\
&+ a^2(1-P_{\tilde{T}_{w_0}(i)}(\phi))E_{\setminus\phi}\left\{\tilde{T}_{w_1}(i)^2\right\} \\
&+ 2(1-a)aE_{\setminus\phi}\left\{\tilde{T}_{w_0}(i)\right\}E_{\setminus\phi}\left\{\tilde{T}_{w_1}(i)\right\} \\
&+ P_{\tilde{T}_{w_1}(i)}(\phi)E_{\setminus\phi}\left\{\tilde{T}_{w_0}(i)^2\right\} + P_{\tilde{T}_{w_0}(i)}(\phi)E_{\setminus\phi}\left\{\tilde{T}_{w_1}(i)^2\right\} \\
&+ P_{\tilde{T}_{w_0}(i)}(\phi)P_{\tilde{T}_{w_1}(i)}(\phi)E\left\{\tilde{T}_s^I(i)^2\right\},
\end{aligned} \tag{4.5}$$

where $E_{\setminus\phi}\left\{\tilde{T}_{w_m}(i)^k\right\}$ denotes the partial k-th moment of $\tilde{T}_{w_m}(i)$ when it is not in a hole.

In the general algorithm in MPEG-VSRS, the hole pixels are inpainted using known pixels around the hole region. The known pixel selection process is based on solving the Navier-Stokes equations or the fast marching algorithm [9, 64]. Similarly, the 1D parallel algorithm in MPEG-VSRS also inpaints the holes using neighboring known pixels depending on whether they belong to the foreground or the background objects. Since some pixels after texture merging have non-zero probabilities of being a hole, geometric randomness is introduced. Therefore, accurately accounting for the inpainting process increases estimation complexity.

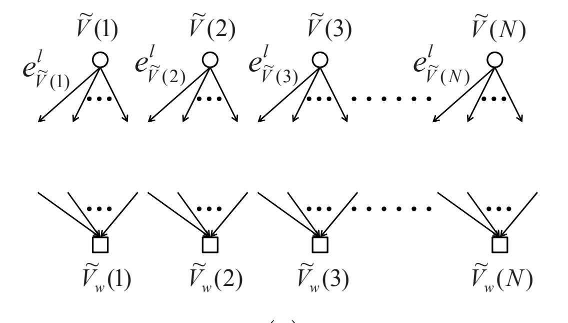
To facilitate distortion estimation, we use a simpler hole filling method with similar performance, which will be verified in Section 4.6. In our method, when a hole pixel is detected, the left pixel is first used for filling. If it is not available, the top pixel is chosen. When both neighbors are unavailable, a constant of 128 is used. This simple inpainting method is applied to each pixel according to the raster scan order.

Based on the simplified hole filling method, if a pixel needs to be inpainted, its first and second moments are given as follows ($k = 1, 2$).

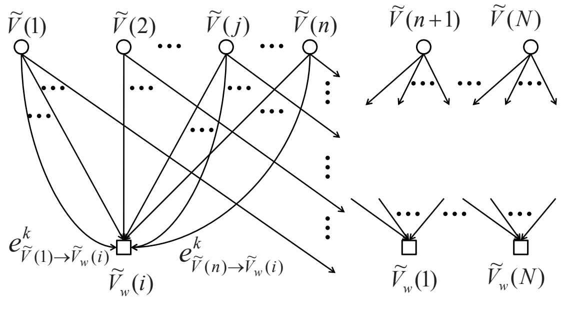
$$E\left\{\tilde{T}_s^I(i)^k\right\} = \begin{cases} E\left\{\tilde{T}_s(i-1)^k\right\}, & \text{left available,} \\ E\left\{\tilde{T}_s(i-W)^k\right\}, & \text{top available,} \\ 128^k, & \text{otherwise,} \end{cases} \tag{4.6}$$

where W is the width of the image.

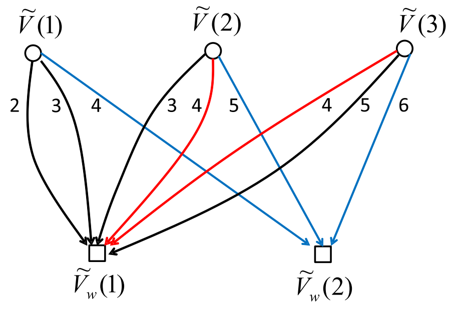
As shown in Eq. (4.4) and (4.5), to estimate the synthesized view distortion, all we need is to calculate the texture pixel hole probabilities $P_{\tilde{T}_{w_m}(i)}(\phi)$ ($m = 0, 1$) and the two partial moments in the warped texture images $E_{\setminus\phi}\left\{\tilde{T}_{w_m}(i)^k\right\}$ ($m = 0, 1$ and $k = 1, 2$). However, in both the general and 1D view synthesis algorithms, the distributions of the synthesized texture pixels are determined by those of the reference depth maps. Hence, in the next section, we develop a graphical model method to capture the relationship between the reference view and the synthesized view, from which we will obtain the distributions and the distortions of the synthesized texture pixels.



(a)



(b)



(c)

Figure 4.1: Graphical model for the general view synthesis algorithm. (a) Original bipartite probabilistic graph. (b) A re-arranged graph. (c) An example.

4.4 Graphical Model-based View Synthesis Distortion Estimation

In this section, we assume that only the reference depth images contain errors. Each pixel in the reference depth image is affected by a random noise signal with a known distribution, and the noises are independent from pixel to pixel (In Sec. 4.5, we will show how to recursively estimate the error distribution in the reference view when there are transmission errors). A probabilistic graphical model-based method is designed for the two synthesis algorithms in MPEG-VSRS to estimate the distortion in the synthesized view.

4.4.1 Graphical model-based distortion estimation for the general view synthesis algorithm

The general view synthesis algorithm in MPEG-VSRS consists of two warping steps. We first present the details of a probabilistic graphical model to obtain the depth pixel distribution and hole probability for each pixel after the forward warping process. The main difficulty for the formulation is warping competition, where multiple pixels in the reference depth map warp to the same location in the virtual depth map, and the largest depth level (closest to the camera) among them is chosen as the final winning depth.

The relationship between a reference depth map and its warped virtual depth map can be represented by a bipartite probabilistic graph as shown in Fig. 4.1 (a), where vertex $\tilde{V}(j)$ represents the j-th pixel in the reference view, and $\tilde{V}_w(i)$ is the i-th pixel in the virtual view. To simplify the notation, the subscript m is dropped in this section, since the result can be used for warping from either the left or right view. Due to random error, different depth levels can be assumed by a depth pixel in the reference view, with the corresponding probabilities. Since each depth level has its own homography matrix, an erroneous reference depth pixel could lead to multiple possible warping destinations in the virtual view, as shown in Eq. (4.1). This is represented by edges emitting from $\tilde{V}(j)$ to a number of $\tilde{V}_w$ vertices. Each edge corresponds to one possible warping path with one depth level for $\tilde{V}(j)$ and the corresponding probability. The l-th edge among all the edges emitting from the vertex $\tilde{V}(j)$ is denoted as $e^l_{\tilde{V}(j)}$. Each vertex $\tilde{V}(j)$ can be connected to any subset of all $\tilde{V}_w$ vertices. The subset can also be empty, *i.e.*, a pixel in the reference view does not warp to any location in the virtual view. Moreover, sometimes a depth pixel in the reference view corrupted by different noises can result in the same warping destination [94]. Consequently, a vertex $\tilde{V}(j)$ can also have multiple edges connecting to the same vertex $\tilde{V}_w(i)$.

During the forward warping step, each pixel in the reference depth map is processed sequentially and independently in the raster scan order. Therefore, $\tilde{V}(j)$ is added after $\tilde{V}(j-1)$ in the graph. Initially the depth levels of the $\tilde{V}_w$ vertices are set to 0. For the k-th edge among all the edges between $\tilde{V}(j)$ and $\tilde{V}_w(i)$, the edge and the depth level it represents are denoted as $e^k_{\tilde{V}(j)\to\tilde{V}_w(i)}$ and $D(e^k_{\tilde{V}(j)\to\tilde{V}_w(i)})$. At any time, when $\tilde{V}(j)$ emits an edge to $\tilde{V}_w(i)$, the depth level of the virtual pixel, $\tilde{D}_w(i)$, will only take the depth level $D(e^k_{\tilde{V}(j)\to\tilde{V}_w(i)})$ when $D(e^k_{\tilde{V}(j)\to\tilde{V}_w(i)})$ is greater than or equal to the current level of $\tilde{D}_w(i)$, according to the depth warping competition rule.

In this section, the depth level of the edge $e^k_{\tilde{V}(j)\to\tilde{V}_w(i)}$ and its corresponding probability are assumed to be known. From the graphical model defined above, we are now ready to derive the depth level distribution of each depth pixel in the virtual depth map.

We use $P_{win}(e^k_{\tilde{V}(j)\to\tilde{V}_w(i)})$ to represent the probability that the depth level of virtual pixel $\tilde{V}_w(i)$ takes the depth level $D(e^k_{\tilde{V}(j)\to\tilde{V}_w(i)})$ from edge $e^k_{\tilde{V}(j)\to\tilde{V}_w(i)}$ after warping all pixels with possible warping competitions. We call this the winning probability of edge $e^k_{\tilde{V}(j)\to\tilde{V}_w(i)}$.

The winning probability of an edge can be calculated by marginalizing the joint PMF properly. To simplify the notation, the graph can be re-arranged to that in Fig. 4.1 (b), where the vertex of interest, $\tilde{V}_w(i)$, and its connected reference vertices $\tilde{V}(j)$ $(j = 1, ..., n)$ are grouped to the left. These connected reference vertices are added chronologically, with $\tilde{V}(n)$ being the latest vertex connecting to $\tilde{V}_w(i)$. Other reference vertices that do not have any connection to $\tilde{V}_w(i)$ are grouped to the right, and are denoted by $\tilde{V}(z)$ $(z = n+1, ..., N)$.

Since vertices unconnected to $\tilde{V}_w(i)$ are irrelevant to the depth probability of $\tilde{V}_w(i)$, we only need to consider $\tilde{V}(1)$ to $\tilde{V}(n)$. Based on the warping competition rule, when the edge $e^k_{\tilde{V}(j)\to\tilde{V}_w(i)}$ from $\tilde{V}(j)$ is the final winning edge, all the previous vertices $\tilde{V}(z)$ $(z = 1, ..., j-1)$ cannot emit edges to $\tilde{V}_w(i)$ with values greater than $D(e^k_{\tilde{V}(j)\to\tilde{V}_w(i)})$, and all the subsequent vertices $\tilde{V}(z)$ $(z = j+1, ..., n)$ cannot emit edges to $\tilde{V}_w(i)$ with values greater than or equal to $D(e^k_{\tilde{V}(j)\to\tilde{V}_w(i)})$. Therefore let $P(e^k_{\tilde{V}(j)\to\tilde{V}_w(i)})$ denote the probability of the edge $e^k_{\tilde{V}(j)\to\tilde{V}_w(i)}$, and assume that the depth pixels are affected by noise signals independently, the winning probability of an edge can be written as

$$\begin{aligned}&P_{win}(e^k_{\tilde{V}(j)\to\tilde{V}_w(i)}) = P(e^k_{\tilde{V}(j)\to\tilde{V}_w(i)})\times\\&\prod_{z=1}^{j-1}\left(1-\sum_{l:D(e^l_{\tilde{V}(z)\to\tilde{V}_w(i)})>D(e^k_{\tilde{V}(j)\to\tilde{V}_w(i)})}P(e^l_{\tilde{V}(z)\to\tilde{V}_w(i)})\right)\times\\&\prod_{z=j+1}^{n}\left(1-\sum_{l:D(e^l_{\tilde{V}(z)\to\tilde{V}_w(i)})\geq D(e^k_{\tilde{V}(j)\to\tilde{V}_w(i)})}P(e^l_{\tilde{V}(z)\to\tilde{V}_w(i)})\right).\end{aligned} \tag{4.7}$$

From the winning probability of each edge, the probability that the depth level of a synthesized pixel, $\tilde{D}_w(i)$, takes a particular value d can be obtained by summing up the winning probabilities of all edges connected to $\tilde{V}_w(i)$ with depth level of d, *i.e.*,

$$P_{\tilde{D}_w(i)}(d) = \sum_{j=1}^{n}\sum_{k:D(e^k_{\tilde{V}(j)\to\tilde{V}_w(i)})=d} P_{win}(e^k_{\tilde{V}(j)\to\tilde{V}_w(i)}). \tag{4.8}$$

We also use $P_{\tilde{D}_w(i)}(\phi)$ to represent the probability of $\tilde{D}_w(i)$ taking no value from any edge, *i.e.*, $\tilde{V}_w(i)$ is in a hole. This is simply given by

$$P_{\tilde{D}_w(i)}(\phi) = \prod_{j=1}^{n}(1-\sum_{k}P(e^k_{\tilde{V}(j)\to\tilde{V}_w(i)})). \tag{4.9}$$

Eq. (4.8) and (4.9) define the complete depth PMF $P_{\tilde{D}_w(i)}(d)$ of the synthesized depth pixel $\tilde{V}_w(i)$.

To illustrate Eq. (4.7) to (4.9), a simple example with three reference pixels and two virtual pixels is shown in Fig. 4.1 (c), where each reference pixel can take three possible depth levels as shown in the figure with equal probability. Consider, for example, the probability $P_{\tilde{D}_w(1)}(4)$. According to the warping competition, there are two possible winning edges with depth level of 4, which come from $\tilde{V}(2)$ and $\tilde{V}(3)$ respectively. The former is chosen when $\tilde{D}(1)$ takes any value in Fig. 4.1 (c), $\tilde{D}(2) = 4$, and $\tilde{D}(3) = 6$, with a winning probability of 1/9. On the other hand, the latter will be selected when both $\tilde{D}(1)$ and $\tilde{D}(2)$ take any value in Fig. 4.1 (c), and $\tilde{D}(3) = 4$, with a winning probability of 1/3. Together, we get $P_{\tilde{D}_w(1)}(4) = 4/9$. Similarly, we can get that $P_{\tilde{D}_w(1)}(2) = 1/27$, $P_{\tilde{D}_w(1)}(3) = 4/27$, and $P_{\tilde{D}_w(1)}(5) = 1/3$. Moreover, $\tilde{V}_w(1)$ is a hole when all reference pixels connect to $\tilde{V}_w(2)$. Therefore $P_{\tilde{D}_w(1)}(\phi) = 1/27$. It can be verified that these results agree with Eq. (4.7) to (4.9).

After the forward warping, median filtering and hole dilation are applied. Finding the depth distribution after these operations is quite complicated. Therefore in this chapter we disable these operations to facilitate distortion estimation, and leave it as a future work to incorporate them in the analysis.

Once the depth PMF $P_{\tilde{D}_w(i)}(d)$ of each synthesized pixel is known, we can find the partial texture moments required by Eq. (4.4) and Eq. (4.5). During the texture reverse warping using the reverse homography matrix and Eq. (4.1), different depth levels of the virtual pixel could be warped to different reference pixel locations. The corresponding warped reference texture values will be chosen as the virtual texture value with different probabilities. Therefore, let d_l $(l = 1, ..., L)$ be the l-th possible non-empty depth level of $\tilde{D}_w(i)$, j_l the corresponding index of the reversely warped reference pixel found by Eq. (4.1), and $T(j_l)$ its texture value. The non-hole partial moment of the texture value of the virtual pixel is simply

$$E_{\backslash\phi}\left\{\tilde{T}_w(i)^k\right\} = \sum_{l=1}^{L} P_{\tilde{D}_w(i)}(d_l)\, T(j_l)^k. \tag{4.10}$$

This equation can be used for warping from both the left and the right views ($m = 0, 1$) in Eq. (4.4) and Eq. (4.5). Finally, the virtual texture hole probabilities is the same as the virtual depth hole probability in Eq. (4.9).

4.4.2 Graphical model-based Distortion Estimation for the 1D parallel synthesis algorithm

Although the 1D parallel synthesis algorithm in MPEG-VSRS only employs one warping process, warping competition still exists. Therefore the same graphical model in Fig. 4.1 can still be applied. The difference is that we can get the texture distribution of a virtual pixel without having to explicitly find its depth distribution first, thanks to the simplified 1D parallel view synthesis.

Since only the reference depth has error in this section, the reference texture value $T(j)$ does not contain random error, but the virtual texture value $\tilde{T}_w(i)$ is still subject to error, as it is affected by the depth-based warping. Due to rounding operation, there could be multiple warping paths between the same pair of $\tilde{V}(j)$ and $\tilde{V}_w(i)$, but all of these edges have the same reference texture value $T(j)$.

The texture and depth levels of vertex $\tilde{V}_w(i)$ are initialized to 0. According to the MPEG-VSRS warping competition rule, when $\tilde{V}(j)$ emits an edge $e^k_{\tilde{V}(j)\rightarrow\tilde{V}_w(i)}$ to $\tilde{V}_w(i)$, if $D(e^k_{\tilde{V}(j)\rightarrow\tilde{V}_w(i)}) > \tilde{D}_w(i)$, then both the texture and depth of the virtual vertex $\tilde{V}_w(i)$ will be updated simultaneously by $T(j)$ and $D(e^k_{\tilde{V}(j)\rightarrow\tilde{V}_w(i)})$ respectively. Therefore the winning probability $P_{win}(e^k_{\tilde{V}(j)\rightarrow\tilde{V}_w(i)})$ is similar to Eq. (4.7). The only difference is that the 1D parallel algorithm does not update the virtual view when the depth of a new reference pixel equals to the current depth of the virtual pixel. Therefore the $>$ and $\geq$ conditions in the second and the third lines in Eq. (4.7) should be switched.

Since there is no texture reverse warping in the 1D parallel algorithm, *i.e.*, the texture is updated together with the depth as described above, the probability that $\tilde{T}_w(i)$ takes the texture value of $T(j)$ from vertex $\tilde{V}(j)$ can be directly obtained by summing up the winning probabilities of all edges between $\tilde{V}(j)$ and $\tilde{V}_w(i)$. We denote this by

$$P_{win}(e_{\tilde{V}(j)\rightarrow\tilde{V}_w(i)}) = \sum_k P_{win}(e^k_{\tilde{V}(j)\rightarrow\tilde{V}_w(i)}), \tag{4.11}$$

From this, we can get the required partial moments $E_{\setminus\phi}\left\{\tilde{T}_{w_m}(i)^k\right\}$ in Eq. (4.4) and (4.5)

$$E_{\setminus\phi}\left\{\tilde{T}_w(i)^k\right\} = \sum_{j=1}^{n} P_{win}(e_{\tilde{V}(j)\rightarrow\tilde{V}_w(i)})\, T(j)^k, \tag{4.12}$$

where the summation is over all reference pixels that can be warped to the target virtual pixel. Note that it is slightly different from Eq. (4.10) because there is no reverse warping in the 1D synthesis algorithm. This equation can also be used for warping from both the left and the right views ($m = 0, 1$) in Eq. (4.4) and (4.5). The hole probability $P_{\tilde{T}_w(i)}(\phi)$ is also given by Eq. (4.9).

4.5 Recursive Optimal Distortion Estimation (RODE) and Integration with the Graphical Model Method

In this section, we consider the case where the reference texture and depth images are independently encoded with codecs such as the H.264/AVC encoder. Slice mode is enabled in the encoder with several rows of macroblocks (MBs) grouped into one slice. Each slice in the encoded bitstream is subject to packet loss with probability p. When a packet is lost, the co-located pixels in the previously reconstructed frame are copied to the current frame to conceal the error.

In order to utilize the graphical model method developed above to estimate the synthesized view distortion, the complete PMFs of both the depth and texture values of each reference pixel are needed. The ROPE algorithm in [89, 73] can only estimate the first two moments of each pixel. In this section, we generalize the ROPE method and develop a recursive optimal distribution estimation (RODE) method, which can recursively estimate the complete PMF of each depth or texture pixel. We will also show how to integrate it with the graphical model method.

Let $\hat{X}(i,n)$ and $\tilde{X}(i,n)$ denote the i-th depth or texture pixel in frame n reconstructed by the encoder and decoder respectively, $\hat{e}(i,n)$ the encoder-side reconstructed motion-compensated residue signal for pixel i in frame n, and $P_{\tilde{X}(i,n)}(x)$ the PMF for pixel $\tilde{X}(i,n)$.

For a pixel in an Intra-coded MB, we have the following.

$$\tilde{X}(i,n) = \begin{cases} \hat{X}(i,n), & \text{if the pixel is received,} \\ \tilde{X}(i,n-1), & \text{if the pixel is lost,} \end{cases} \tag{4.13}$$

where we assume that constrained intra prediction is used [87], which prevents intra MB from using neighboring Inter MBs for prediction.

To find the PMF of the intra-coded pixel, note that when its data are received, its PMF is simply a Kronecker delta function with a value of 1 at the location of the encoder reconstruction and 0 elsewhere. If the pixel is lost, the PMF from the previous frame will be propagated to the current frame due to the error concealment described above. Therefore

$$P_{\tilde{X}(i,n)}(x) = (1-p)\delta\left(x - \hat{X}(i,n)\right) + pP_{\tilde{X}(i,n-1)}(x). \tag{4.14}$$

For a pixel in an Inter-coded MB, we have the following.

$$\tilde{X}(i,n) = \begin{cases} \hat{e}(i,n) + \tilde{X}(j,n-1), & \text{if the pixel is received,} \\ \tilde{X}(i,n-1), & \text{if the pixel is lost,} \end{cases} \tag{4.15}$$

where $\tilde{X}(j,n-1)$ is the pixel in frame $n-1$ pointed to by the motion vector for pixel $\tilde{X}(i,n)$. In the first case, the PMF of $\tilde{X}(i,n)$ is shifted from that of $\tilde{X}(j,n-1)$ by $\hat{e}(i,n)$. Therefore,

$$P_{\tilde{X}(i,n)}(x) =(1-p)P_{\tilde{X}(j,n-1)}(x - \hat{e}(i,n)) + pP_{\tilde{X}(i,n-1)}(x). \tag{4.16}$$

Pixel averaging operations such as deblocking filter and sub-pixel motion estimation present a challenge for PMF estimation, because the corresponding joint PMF will require the convolution of individual PMFs, which is quite complicated. Consequently, in this chapter, we also disable these operations, and leave it as a future research.

The graphical model method assumes that only the reference depth has error. When both the reference depth and texture have errors, the RODE method and the graphical model method can be easily integrated. All we need is to replace $T(j_l)$ and $T(j)$ in Eq. (4.10) and (4.12) by their expected values $E\{\tilde{T}(j_l)\}$ and $E\{\tilde{T}(j)\}$.

One drawback of the RODE method is its high memory usage, because in the worst case 256 numbers are needed to record the PMF of each 8-bit pixel. To reduce memory usage, a quantization process can be used by grouping some neighboring outcomes together, at the expense of estimation accuracy. If Q outcomes are to be combined into one outcome, the original outcome with the largest probability will take the total probabilities of the Q outcomes. This will be demonstrated in the next section.

Due to the fact that slice encoding is used, a depth pixel may not be independent from its neighboring depth pixels. However, jointly calculating the distribution for all the depth pixels in a slice drastically increases storage requirement and computation complexity. Therefore, in this chapter, we assume that the depth pixels are independent and use the marginal PMFs calculated by the RODE method as the input to the graphical model-based estimation framework. We will demonstrate with experimental results that the estimation results are quite accurate under this assumption.

4.6 Experimental results

To validate the theory developed in this chapter, we conduct three groups of experiments. In the first group, the estimation accuracy of the graphical model method is verified for both the general and 1D parallel algorithms. Next, the RODE method is independently tested. Finally, we test the integrated RODE and the graphical model scheme when there are transmission errors in both the reference texture and depth maps.

In all the experiments, we compare our estimated distortion with the simulated distortion. To find the simulated distortion, the virtual view is first synthesized in an error-free environment and used as a benchmark. Random errors are then injected. The resulting erroneously synthesized texture view is compared to the error-free benchmark and the MSE per pixel is recorded for each frame. The impaired synthesis process is performed many times and the average MSE is used as the simulated distortion. The following Average MSE Mismatch Ratio (AMMR) is then obtained.

$$AMMR = \frac{1}{N} \sum_{n=1}^{N} \left| \frac{ED_{est}(n) - ED_{sim}(n)}{ED_{sim}(n)} \right|, \tag{4.17}$$

where N is the number of frames in the sequence, $ED_{est}(n)$ is the encoder-estimated MSE for frame n, and $ED_{sim}(n)$ is the receiver-side MSE for frame n obtained by simulation and averaged over 200 iterations. Only the luminance component is used in the MSE calculation.

4.6.1 Quadratic Model Method in [41]

We also compare our method to that in [41], which is currently the only available method to estimate packet-loss-induced synthesized distortion, where a simple quadratic model is used to relate the depth error in the reference views to the synthesized texture distortion. We next describe how to modify their quadratic model in order to integrate into our estimation framework.

The distortion estimation algorithm in [41] is MB-based. It also uses the sum of absolute differences (SAD) as the performance metric. Since our scheme is based on the more popular MSE, we modify their method to use the MSE metric. Therefore, the distortion of the j-th MB in the synthesized view $ED_{MB(j)}$ can be written as follows.

$$\begin{aligned}
ED_{MB(j)} &= \sum_{i \in MB(j)} E\{(T_s(i) - \tilde{T}_s(i))^2\} \\
&= \sum_{i \in MB(j)} E\{(aT_{w_0}(i) + (1-a)T_{w_1}(i) \\
&- a\tilde{T}_{w_0}(i) - (1-a)\tilde{T}_{w_1}(i))^2\} \\
&\approx \sum_{i \in MB(j)} a^2 E\{(T_{w_0}(i) - \tilde{T}_{w_0}(i))^2\} \\
&+ (1-a)^2 E\{(T_{w_1}(i) - \tilde{T}_{w_1}(i))^2\} \\
&\triangleq a^2 ED^0_{MB(j)} + (1-a)^2 ED^1_{MB(j)},
\end{aligned} \tag{4.18}$$

where the summation is over all pixels in the MB, and $E\{(T_{w_m}(i) - \tilde{T}_{w_m}(i))^2\}$ $(m = 0, 1)$ is the distortion in the synthesized view caused by texture and depth errors in the left and right views. As in [57], we assume the left and right views are independently affected by errors, therefore the cross term $E\{(T_{w_0}(i) - \tilde{T}_{w_0}(i))(T_{w_1}(i) - \tilde{T}_{w_1}(i))\} \approx 0$.

Eq. (4.18) is similar to Eq. (17) in [41]. In [41], the following quadratic model is used to approximate $ED^m_{MB(j)}$

$$ED^m_{MB(j)} = e^m_{MB(j)} + \frac{1}{2}\alpha^m_{MB(j)}(\epsilon^m_{MB(j)})^2, \tag{4.19}$$

where $e^m_{MB(j)}$ and $\epsilon^m_{MB(j)}$ are the *absolute* texture and depth errors for the j-th MB in the left and right views, and $\alpha^m_{MB(j)}$ is a quadratic model parameter for the MB.

Since our distortion is measured by MSE instead of SAD, $e^m_{MB(j)}$ and $\epsilon^m_{MB(j)}$ in our case are both *squared* errors. However, we can still use the quadratic model in Eq. (4.19). An example is given in Fig. 4.3, which shows the relationship between the squared reference depth error and the squared synthesized texture error of a pixel. The ground truth point does not have any depth error, and the synthesized distortion is 0. In Fig. 4.3 (a), the depth level is decreased until the squared synthesized error reaches a predefined threshold, which corresponds to a boundary point [41]. A quadratic curve is then fitted to pass the ground truth point and the boundary point. Similarly, in Fig. 4.3 (b), the depth level is increased, and another quadratic curve is fitted. As in [41], the sharper of the two quadratic curves is chosen as the quadratic model for the pixel. The parameter $\alpha^m_{MB(j)}$ for the MB is the average of all the quadratic parameters of pixels in the MB.

The results in Fig. 4.3 is very similar to Fig. 3 (a) in [41]. We also get similar results to Fig. 3 (b-c) in [41]. This verifies that the quadratic model is still a reasonable choice when using the MSE metric.

In our simulations, $e^m_{MB(j)}$ and $\epsilon^m_{MB(j)}$ are both given by our per-pixel RODE algorithm, which is more accurate than the MB-based estimation algorithm used in [41].

4.6.2 Performance of the Graphical Model Method

In this part, we only introduce random noises to the reference depth maps. The noise signal is discrete and is uniformly distributed in the range of $[-5, 5]$. Off-range noisy depth levels are rounded to the closest valid value. It should be noted that the derivation of the graphical model does not depend on the distribution of the depth error. Consequently, it works with other noise distributions.

In this part, the Kendo and Balloons sequences [3] are used. 90 frames (frame 90 to frame 179) from each sequence are selected so that the test data contain a significant amount of motion. Views 1 and 5 are used as the reference views, and View 3 is chosen as the virtual view. Simulation results are obtained from the average of 60 experiments.

First, we evaluate the performance of the simple hole filling method described in Sec. 4.4. Fig. 4.4 shows that the PSNR difference between the original inpainting technique and our simplified hole filling method is only 0.07 dB for the Kendo sequence and 0.02 dB for the Balloons sequence. Therefore the simplified method is quite effective for hole filling.

Next we validate our graphical model-based distortion estimation method for the general view synthesis algorithm. Fig. 4.5 shows the MSE comparison for Kendo and Balloons sequences. The simple hole filling method is used. It can be seen that our distortion estimation agrees well with simulation. The AMMRs are 7.13% and 8.77% for the Kendo and Balloons sequence respectively. We also conducted the same tests with other error distributions such as Gaussian distribution and got similar estimation accuracy.

The reason for the mismatch is that the simplified hole filling method can propagate the localized mismatch from one pixel to other pixels in the hole. To verify this, a simple experiment is conducted by always using the value 128 to fill up a hole. The results are shown in Fig. 4.6. In this case, the AMMRs reduce to 0.05% and 0.04% for the two sequences, which are negligible.

Table 4.1: AMMR (%) of the RODE method

	Loss Rate					
	2%	5%	8%	2%	5%	8%
Sequence	GOP=30			GOP=60		
Akiyo	1.97	1.62	2.42	1.37	2.11	1.49
Coastguard	4.12	3.36	3.57	2.21	2.22	2.40
Container	1.14	1.47	2.12	1.37	1.70	1.77
Foreman	3.76	3.23	3.59	3.82	2.74	2.61
News	3.40	3.91	3.64	3.24	1.87	2.16
Silent	2.45	2.38	3.17	3.22	2.07	1.86
Stefan	6.56	5.54	6.04	4.39	2.49	1.74
Average	3.34	3.07	3.51	2.80	2.17	2.00

We next validate the graphical model method for the 1D parallel algorithm using the same way. Fig. 4.7 shows that the simplified hole filling technique has an average loss of 0.05 and 0.02 dB for the Kendo and Balloons sequences respectively. Fig. 4.8 shows that when simplified hole filling method is used, Kendo sequence has an AMMR of 7.20% and Balloons sequence has an AMMR of 8.81%. Fig. 4.9 demonstrates that the estimation mismatch is due to the propagative nature of the simple hole filling method. When constant hole filling is used, the AMMRs are only 0.05% and 0.04% for the Kendo and Balloons sequences respectively.

We also test the performance of the quadratic method in [41]. When only the depth maps contain zero mean uniform errors in the range of [-5,5], $e^m_{MB(j)} = 0$ and $\epsilon^m_{MB(j)} = \sum_{i=-5}^{5} i^2/11$ for $m = 0, 1$ in Eq. (4.19). Fig. 4.10 illustrates the estimation accuracy of the quadratic model method for the Kendo and Balloons sequences. Their corresponding AMMRs are 119.2% and 550.6%, even when constant hole filling is used in the tests. In fact, the quadratic method does not consider the hole filling step. Therefore, its estimated distortion does not change regardless of the hole filling method used. Therefore our graphical model method significantly outperforms the quadratic method.

The graphical model also outperforms the quadratic model in terms of complexity in this test. The quadratic model takes 13.8s per frame on average, whereas the proposed graphical model only needs 4.3s. Note that the complexity of the proposed graphical model grows with the number of outcomes in the error distribution. Therefore it is not expected to be faster than the quadratic method under all circumstances.

Table 4.2: AMMR (%) of the integrated distortion estimation scheme

	GOP Size	Loss Rate		
Sequence		2%	5%	8%
Balloons	60	10.93	9.86	9.17
Kendo	30	3.56	6.03	6.54
Pantomime	60	3.83	4.87	6.12
Champagne	60	21.73	18.73	17.75
Dog	60	7.32	7.00	4.63
Dancer	30	6.53	5.93	5.48
PoznanHall	30	10.37	6.76	6.18
PoznanStreet	60	8.79	7.84	7.32
Book	30	5.00	6.40	5.25
Average		8.67	8.16	7.61

4.6.3 Performance of the RODE

In this part, we introduce packet loss to the transmission of some conventional single-view video sequences, and use the RODE method to estimate the PMF of each decoded pixel. Similar experiments can also be done for the transmission of depth sequences.

It is challenging to validate the PMF at each pixel location due to the amount of data that need to be compared. However, the PMFs at each pixel location can be used to calculate the first and second pixel moments and therefore the expected distortion of the decoded pixel as given in Eq. (4.2).

In our tests, 7 CIF sequences are encoded using H.264/AVC. Slice encoding is enabled with each slice containing 3 rows of macroblocks. Packet loss is applied independently to each slice. Each encoded test sequence is decoded for 200 times with random packet loss. The distortion for each decoded frame is averaged over the 200 tests and used as $ED_{sim}(n)$. The encoder on the other hand calculates $ED_{est}(n)$ for each frame. Fig. 4.11 and Fig. 4.12 show the estimation accuracy for test sequences Akiyo, Foreman, Stefan and Silent. Table 4.1 summarizes the AMMRs for the 7 test sequences. GOP sizes of 30 and 60 are tested. Only I and P frames are used. B frames are disabled. The loss rates tested are 2%, 5% and 8%. Judging from the results, the RODE method is very accurate. We also tested RODE with one row of macroblocks for each slice to confirm that the slice size does not affect estimation accuracy.

We also tested the method to reduce the number of possible outcomes for the PMF estimation. Fig. 4.2 shows the results for the Foreman sequence at 5% loss rate. Quantization on the PMFs is done by grouping Q outcomes into a single outcome, where Q is 8, 16 or 32. The corresponding AMMRs are 3.65%, 6.83% and 16.84%. Therefore the estimation is still quite accurate even when $Q = 16$, but the memory usage can be reduced by 93.75%.

4.6.4 Performance of the Integrated Distortion Estimation Scheme

We now test the performance of our integrated scheme with both the RODE and graphical model methods. Both the texture and depth images are independently encoded using H.264/AVC. Each slice contains 3 rows of macroblocks. The slices in both the texture and depth bitstreams are subject to loss rates of 2%, 5% and 8%. All the texture and depth image bitstreams containing lost packets are decoded first. After applying error concealment, the decoded texture and depth images are used for view synthesis. The resulting erroneously synthesized virtual image is compared to the correctly synthesized virtual image. This process is repeated for 200 times, each time with a different loss pattern. The average simulated distortion for a particular frame $ED_{sim}(n)$ is obtained by averaging the distortions obtained from the 200 tests. For this test, 1D parallel algorithm with the simple hole filling method is selected. Nine test sequences [3, 5, 4] are used. Each sequence has 200 frames except for the Book sequence because its original sequence only contains 100 frames. GOP sizes of 30 and 60 are used. The view in the middle of the left and right reference views is selected as the virtual view.

Table 4.2 presents the estimation accuracy of our integrated scheme for all the test sequences. Frame-by-frame estimation accuracy is illustrated in Fig. 4.13 (a), Fig. 4.14 (a) and Fig. 4.15 (a) for Balloons, Kendo and Dancer sequences at 2%, 5% and 8% respectively.

For comparison, Fig. 4.13 (b), Fig. 4.14 (b) and Fig. 4.15 (b) show the results of the quadratic model method in [41] for the same test configurations, whose accuracies are clearly much worse than our method. The results are also much worse than those in Fig. 4.10. This is because the quadratic model mismatch increases with the depth error. The depth error in Fig. 4.10 is in the small range of $[-5, 5]$. When there are packet loss, the depth error could be much larger.

4.7 Summary

In this chapter, we develop a scheme to estimate the distortion of the synthesized view when transmission error is introduced to the encoded reference texture and depth images. A graphical model-based method is first formulated under the assumption that only the depth images are affected by random noises with known distributions. The method is capable of modeling the warping competition effect such that the necessary depth and texture distributions in the synthesized views can be accurately obtained. Then a recursive optimal distribution estimation (RODE) method is introduced such that pixel level distribution due to transmission error can be precisely calculated. Finally, the RODE method is integrated into the graphical model-based method and synthesized view distortion due to transmission error can be estimated. Experimental results demonstrate the accuracy of the proposed scheme. The estimation accuracy of the proposed scheme can be higher than the block-based methods for 2D video presented in [87]. Even though the complexity of the proposed scheme is higher than the block-based algorithm in [87], the results in this chapter could provide valuable insights into the view synthesis problem from a theoretical perspective, and help to develop fast approximations in the future.

4.A Figures for the Experimental Results

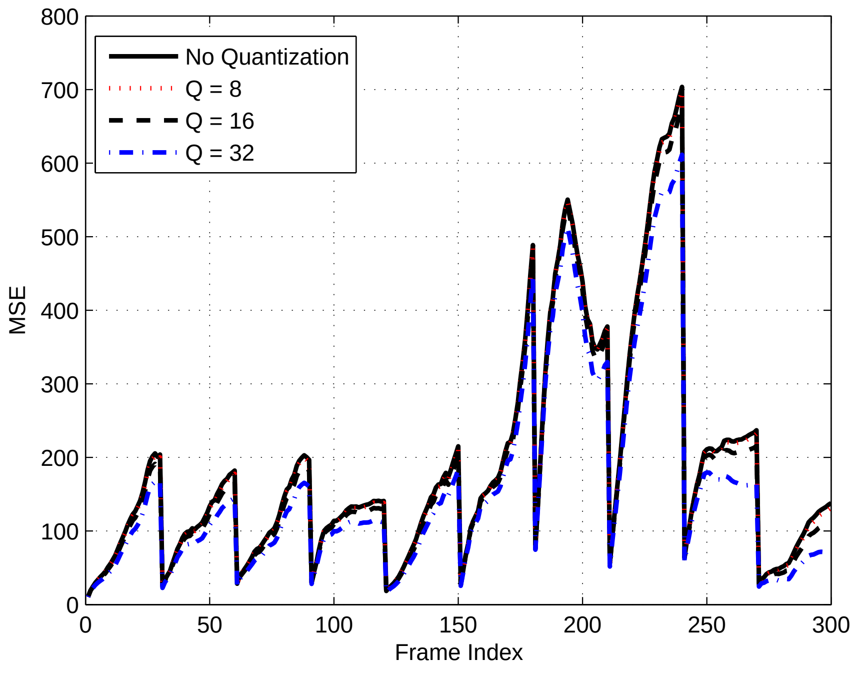
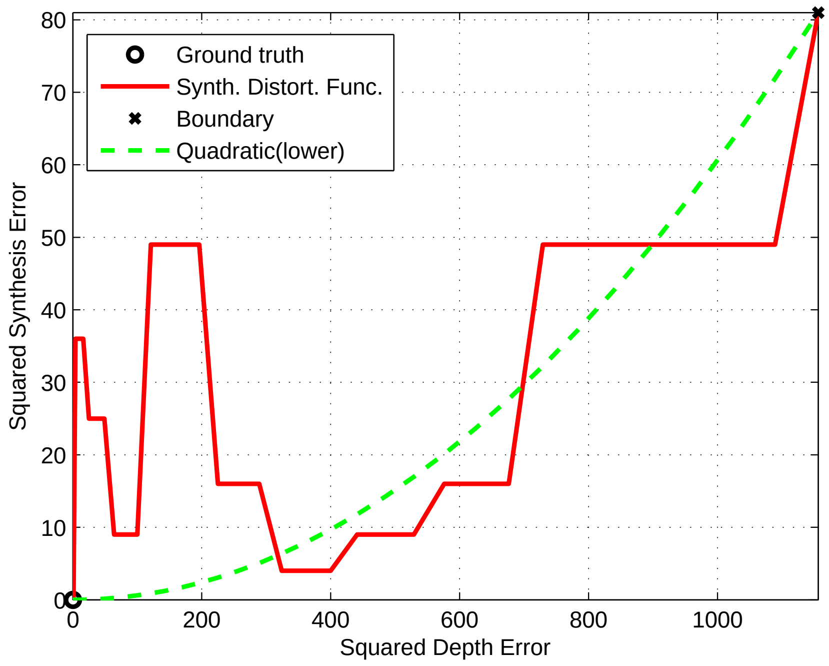
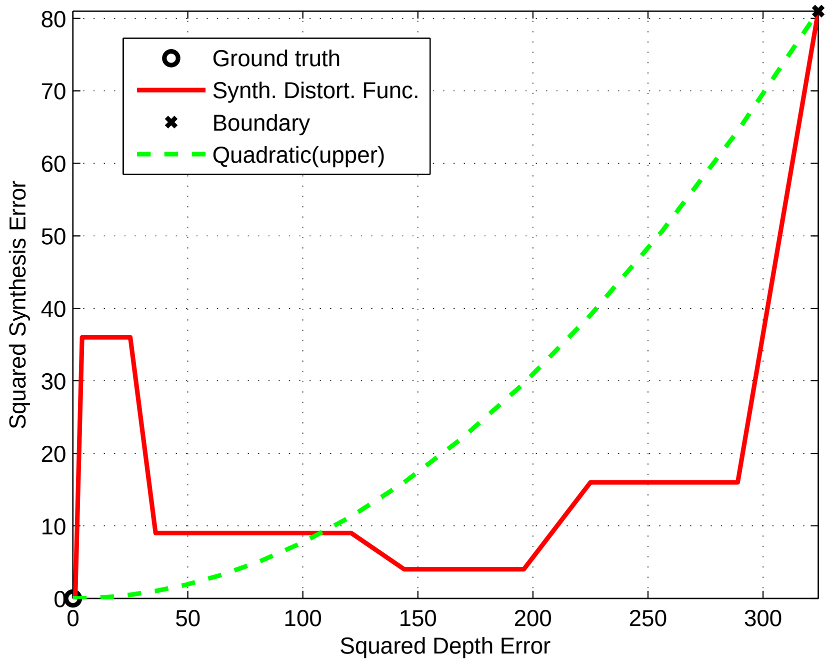


Figure 4.2: Quantization of probability mass functions for Foreman sequence.

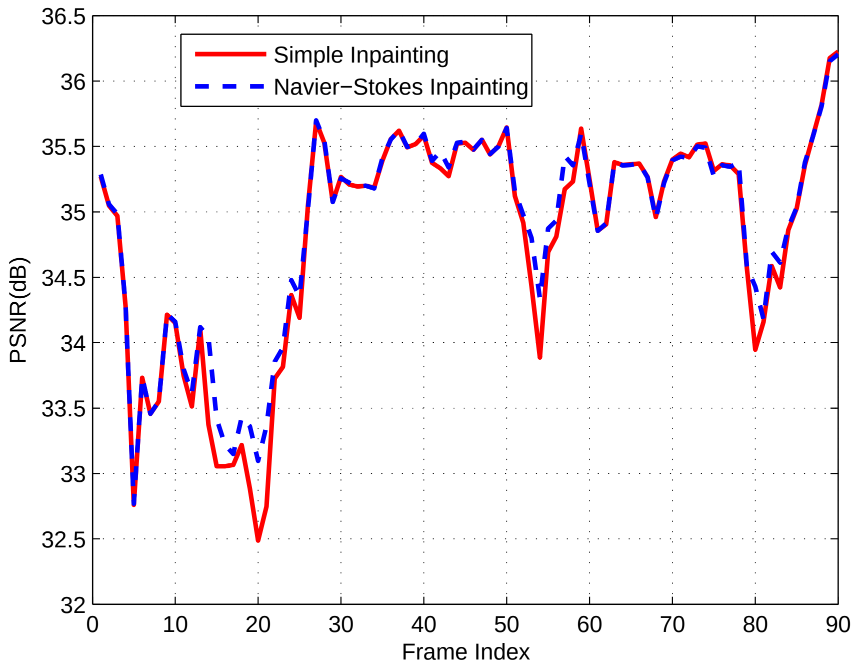


(a)

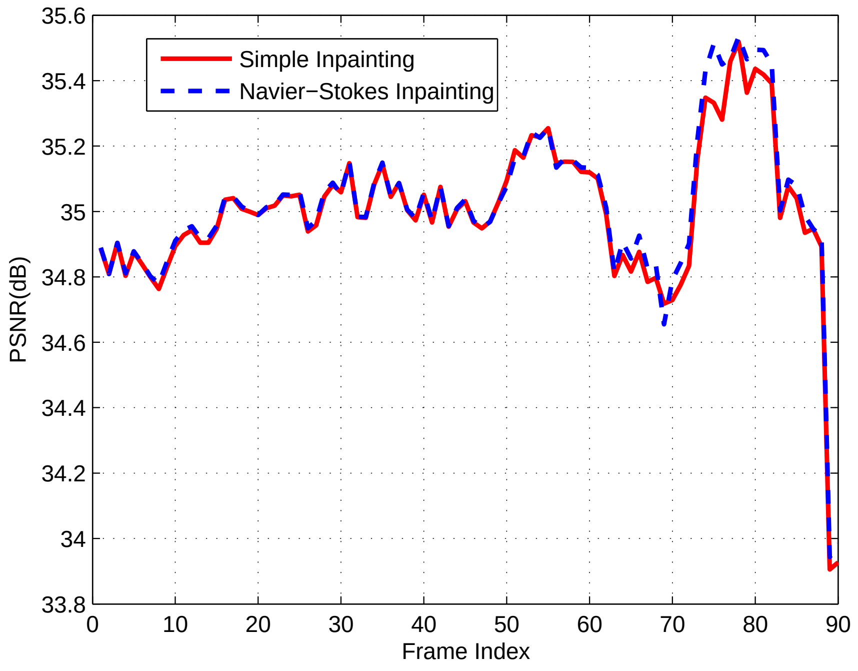


(b)

Figure 4.3: An example of the quadratic model method in [41] for one virtual pixel in the Dancer sequence. (a) Quadratic model constructed using depth levels lower than the true depth. (b) Quadratic model constructed using depth levels higher than the true depth.

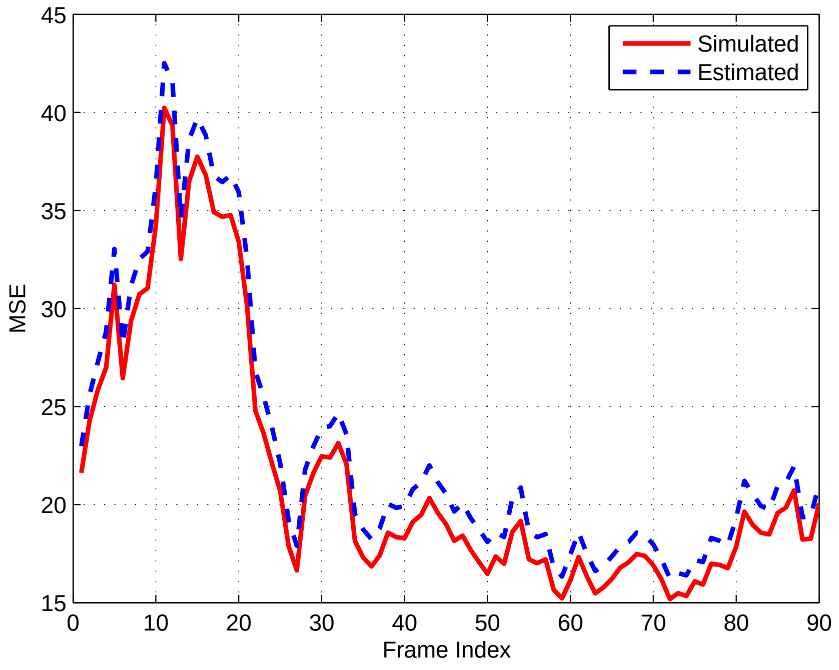


(a)

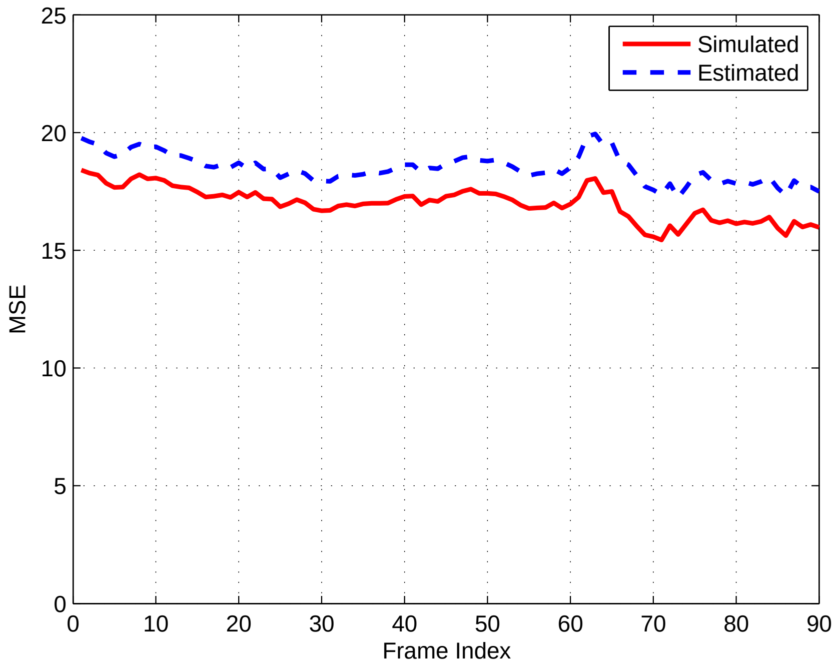


(b)

Figure 4.4: General algorithm hole filling algorithms comparison: (a) Kendo. (b) Balloons.

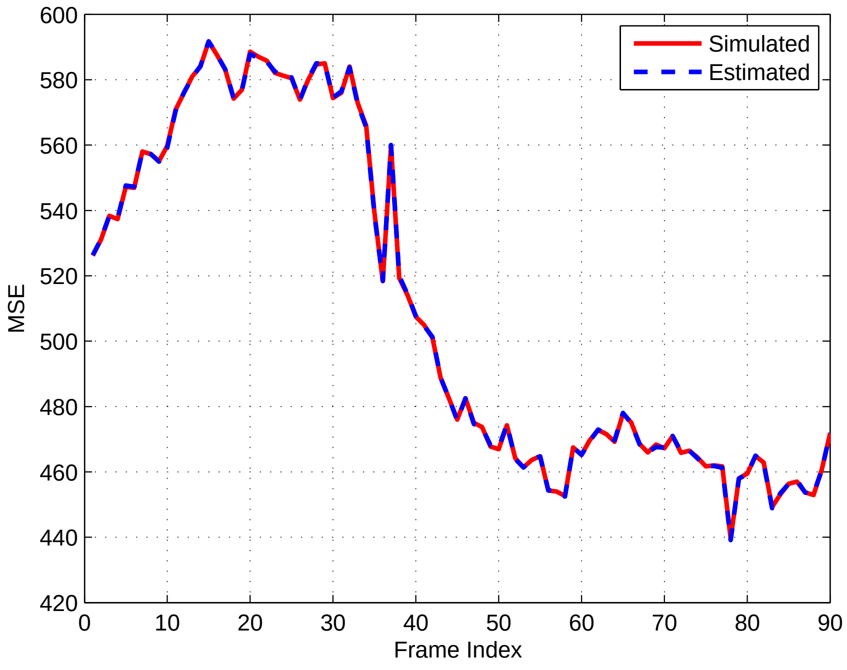


(a)

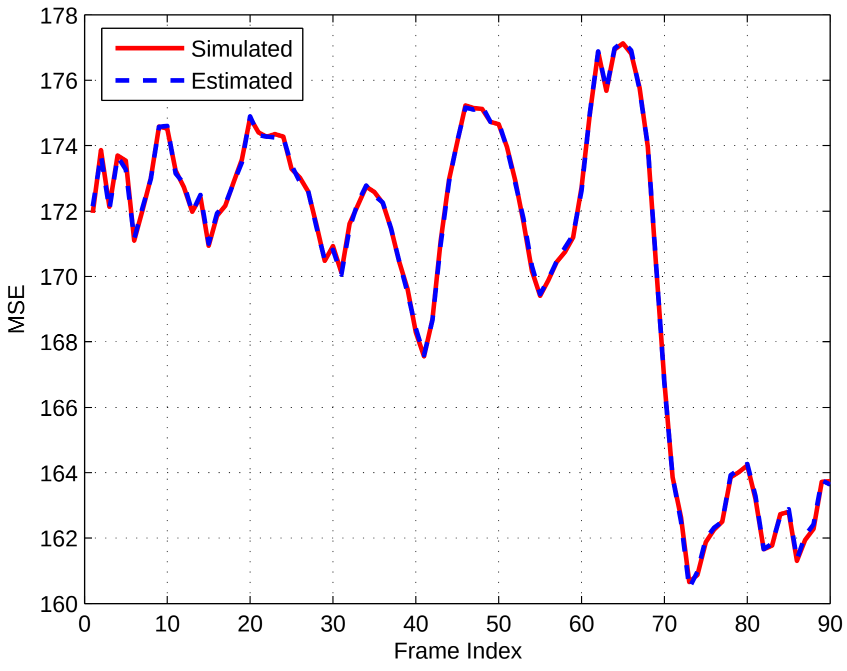


(b)

Figure 4.5: General algorithm distortion estimation with simple hole filling: (a) Graphical model accuracy for Kendo. (b) Graphical model accuracy for Balloons.

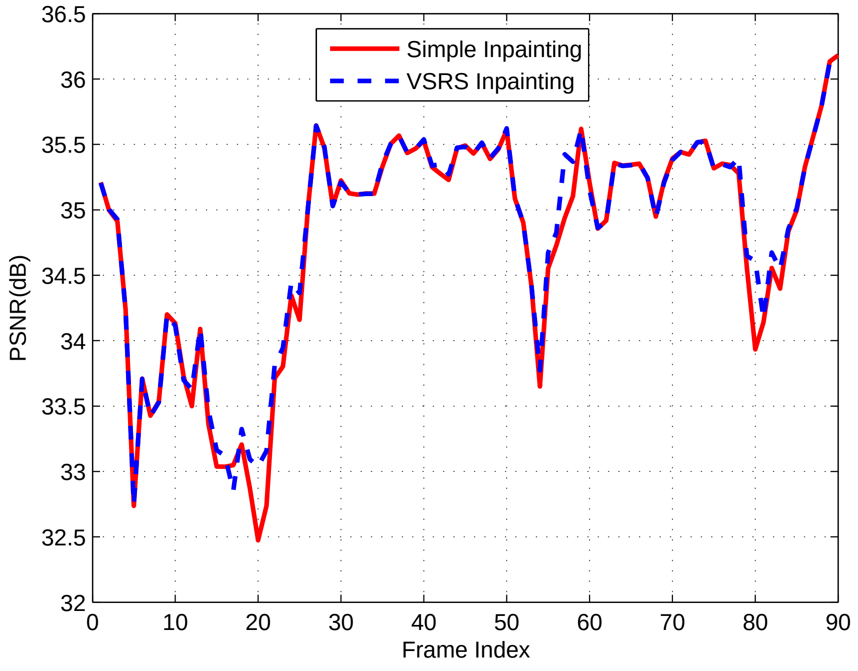


(a)

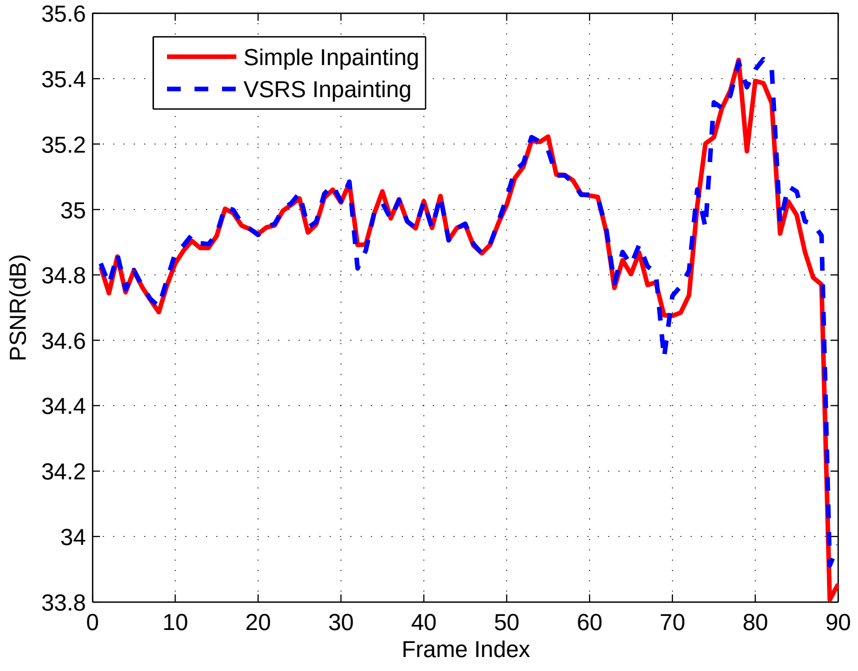


(b)

Figure 4.6: General algorithm distortion estimation with constant hole filling: (a) Graphical model accuracy for Kendo. (b) Graphical model accuracy for Balloons.

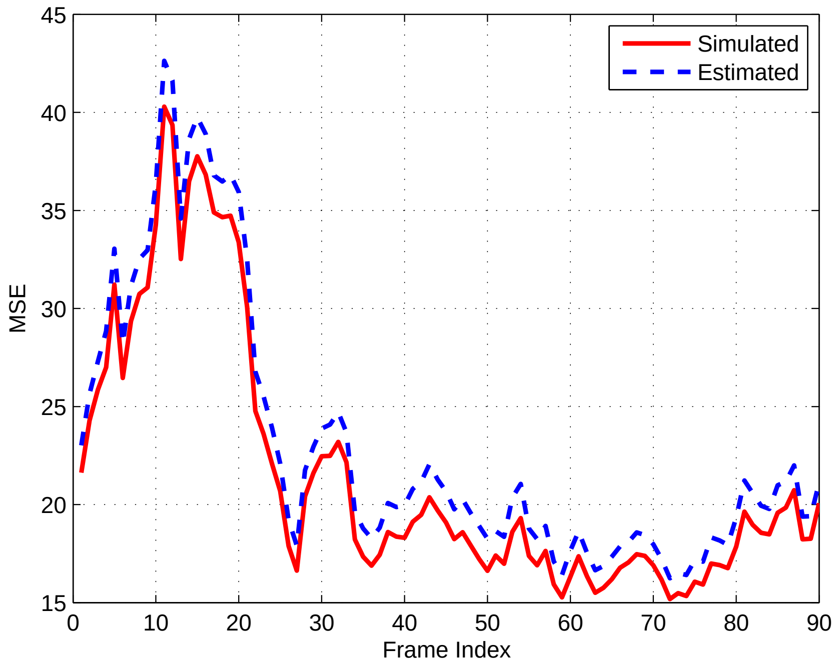


(a)

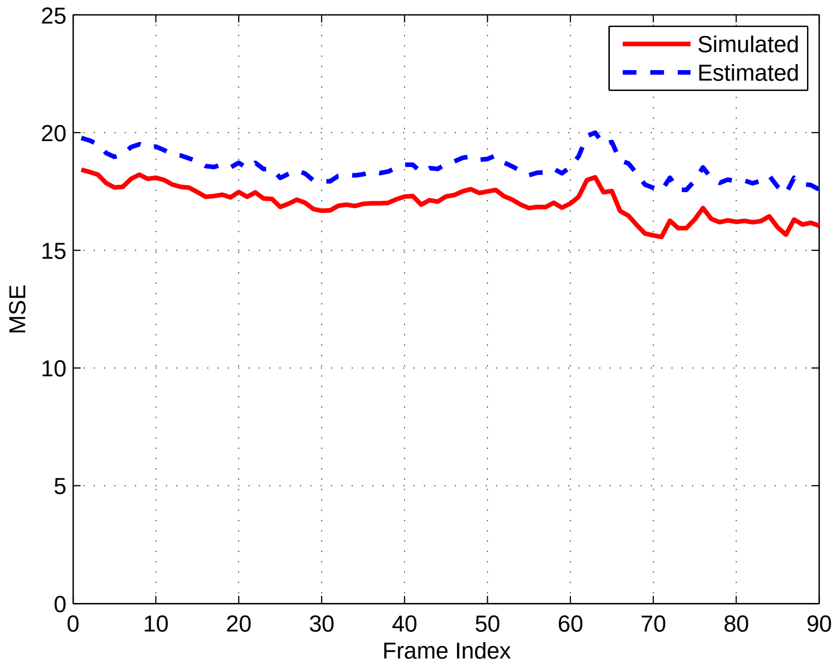


(b)

Figure 4.7: 1D parallel algorithm hole filling methods comparison: (a)Kendo. (b) Balloons

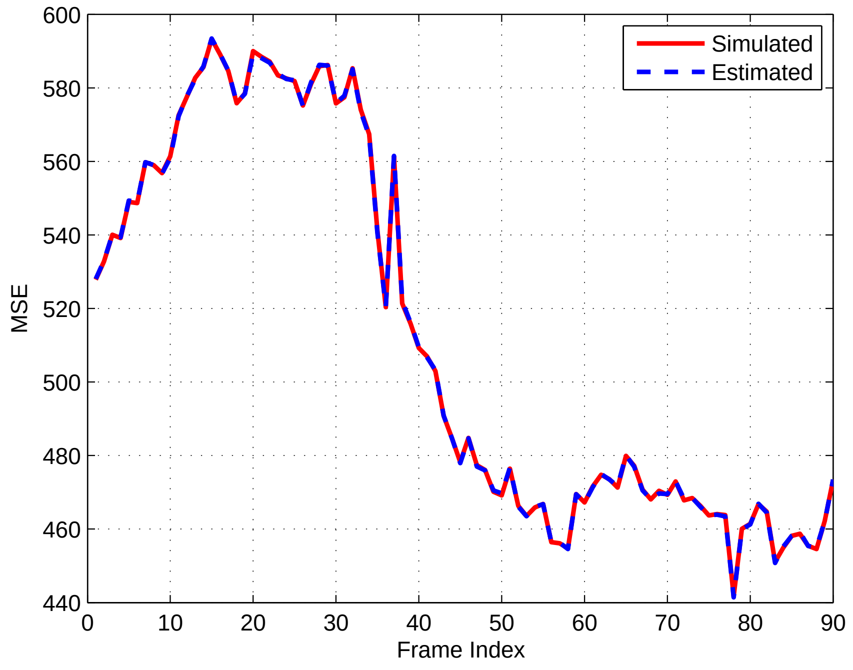


(a)

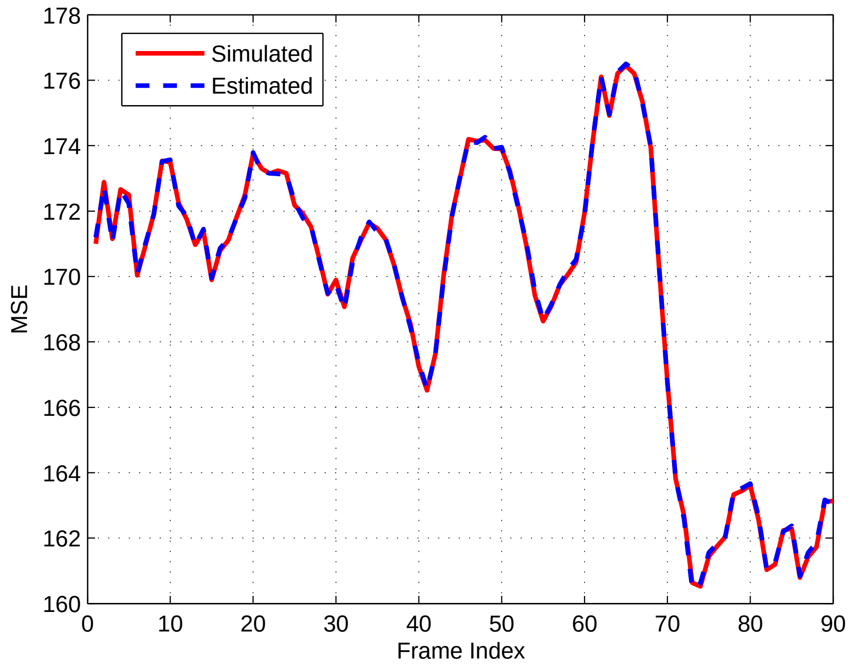


(b)

Figure 4.8: Graphical model accuracy on 1D parallel algorithm with simple hole filling: (a)Kendo. (b) Balloons

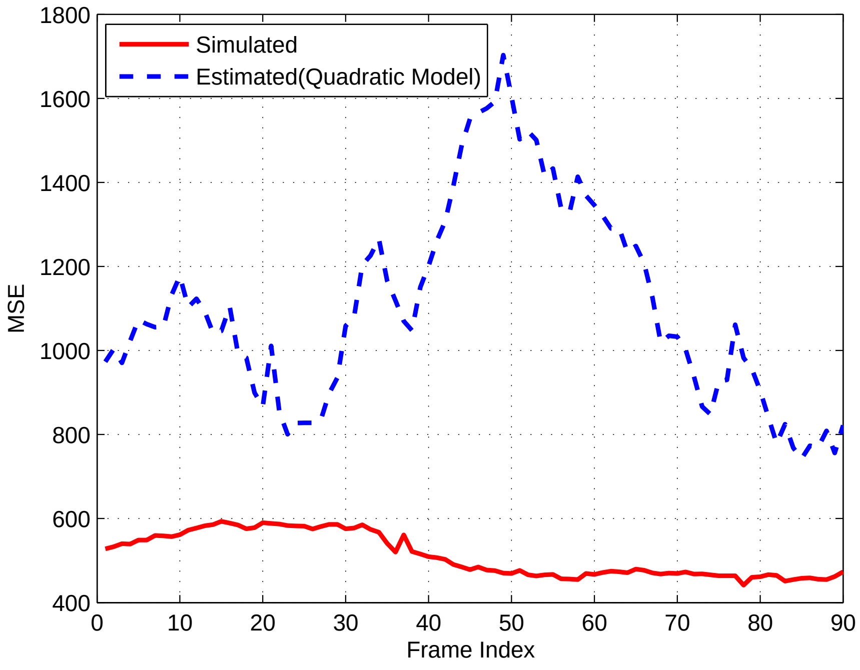


(a)

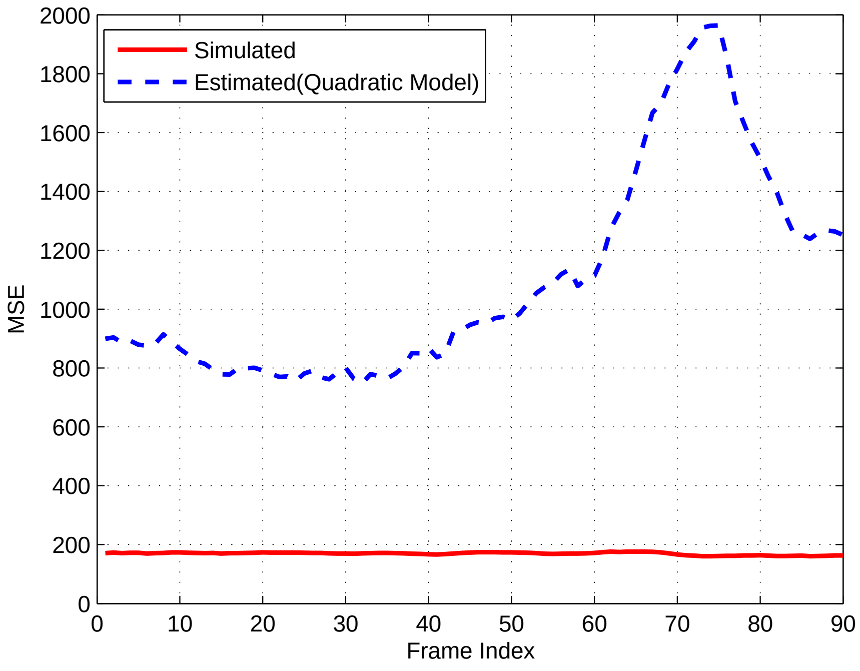


(b)

Figure 4.9: Graphical model accuracy on 1D parallel algorithm with constant value hole filling: (a)Kendo. (b) Balloons

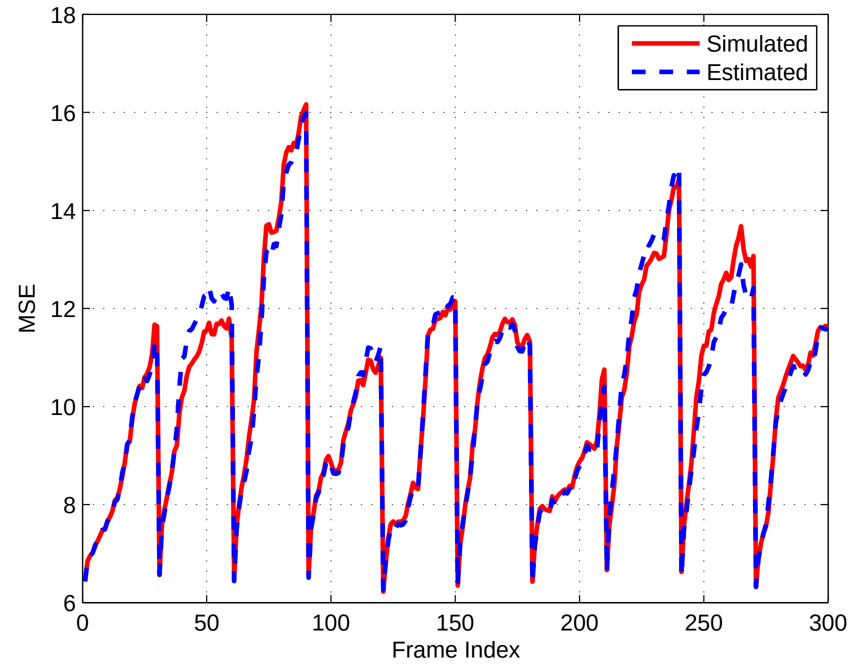


(a)

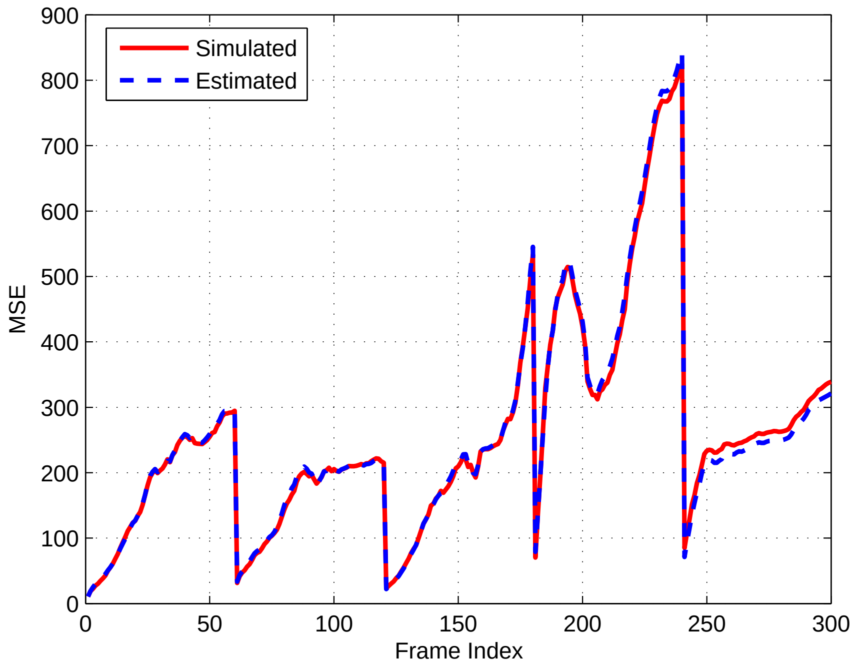


(b)

Figure 4.10: Quadratic model accuracy on 1D parallel algorithm with constant value hole filling.: (a)Kendo. (b) Balloons

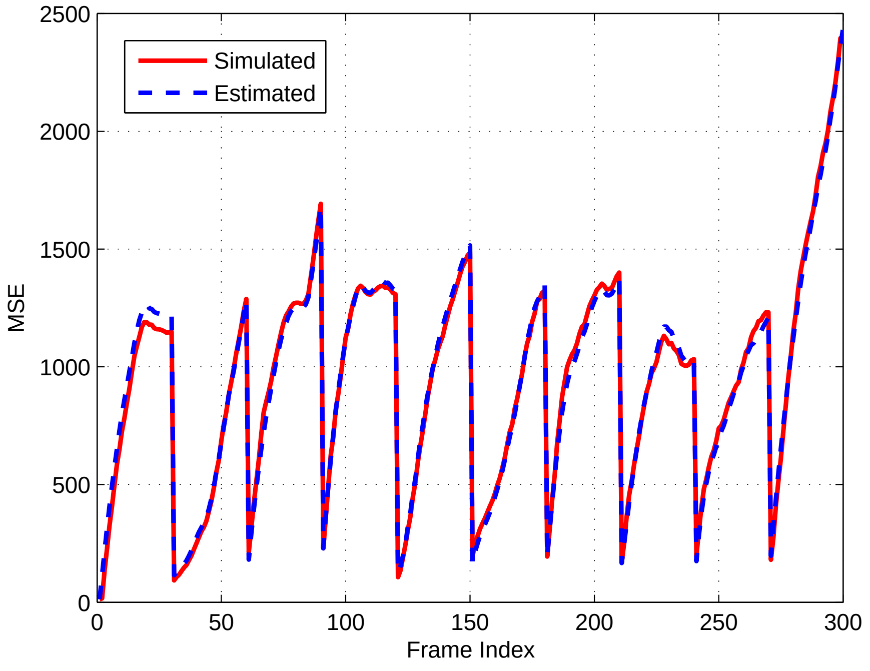


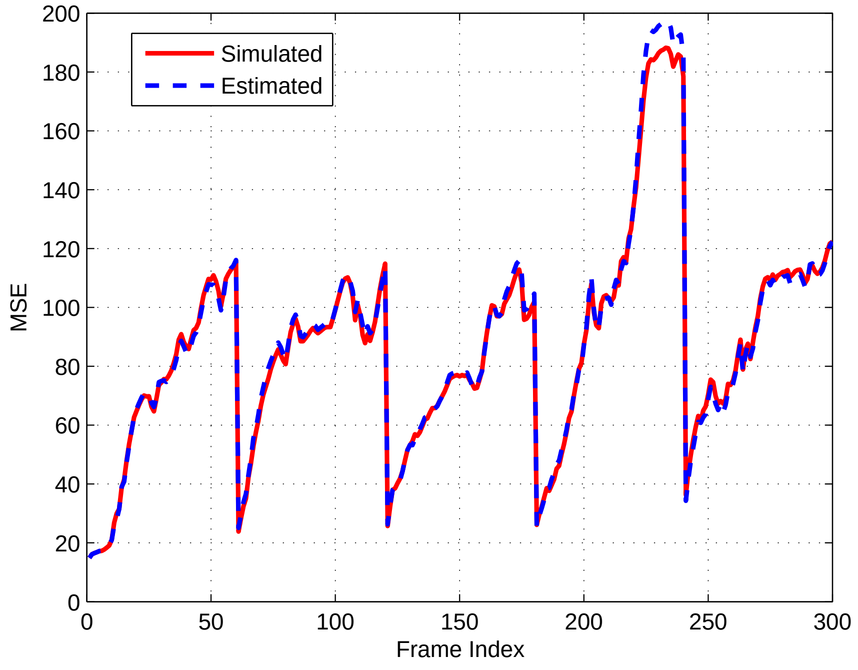
(a)



(b)

Figure 4.11: Estimated decoded frame distortion by RODE. (a) Akiyo: GOP=30, Loss Rate=2%. (b) Foreman: GOP=60, Loss Rate=5%.

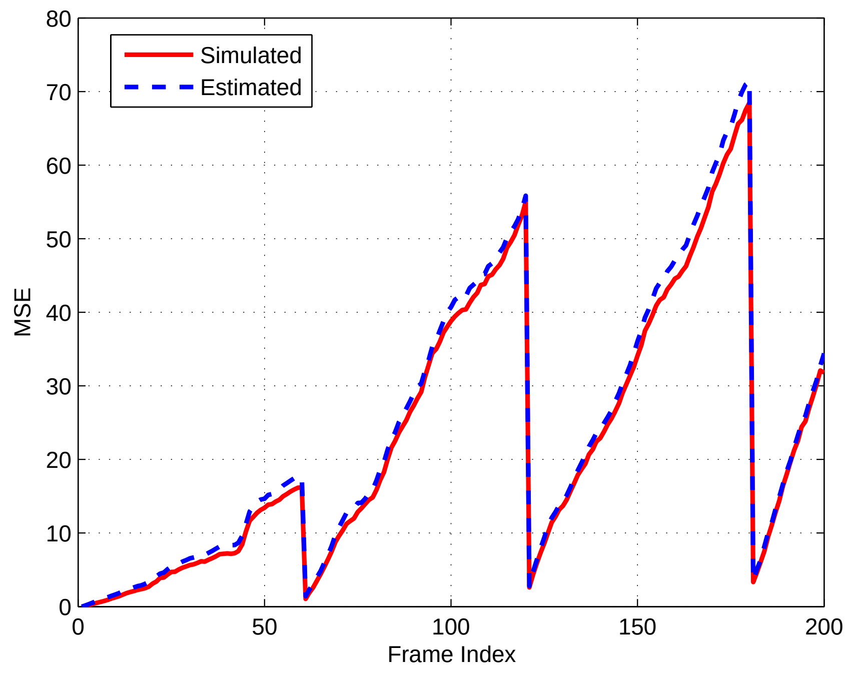




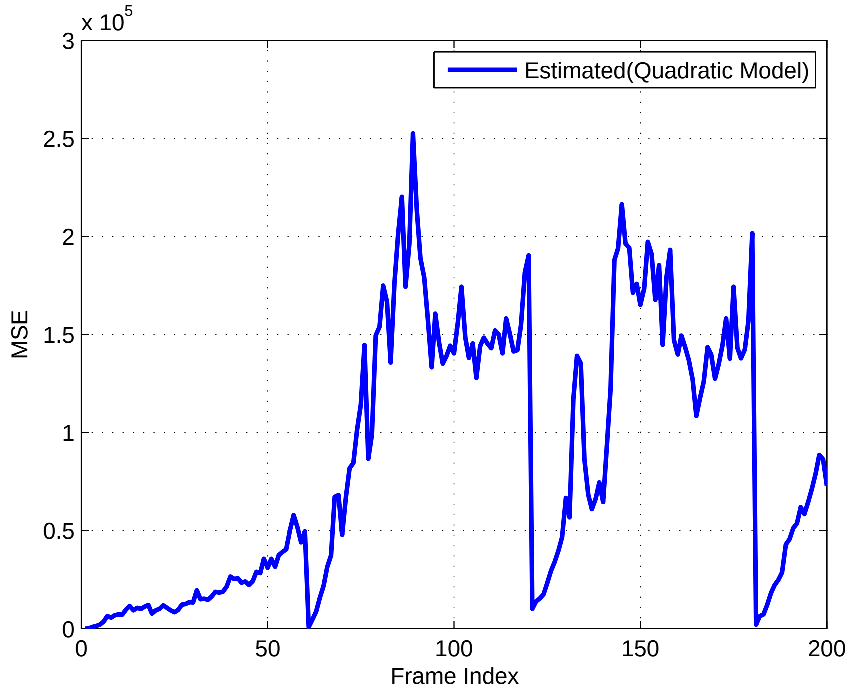
(a)

(b)

Figure 4.12: Estimated decoded frame distortion by RODE. (a) Stefan: GOP=30, Loss Rate=8%. (b) Silent: GOP=60, Loss Rate=8%.

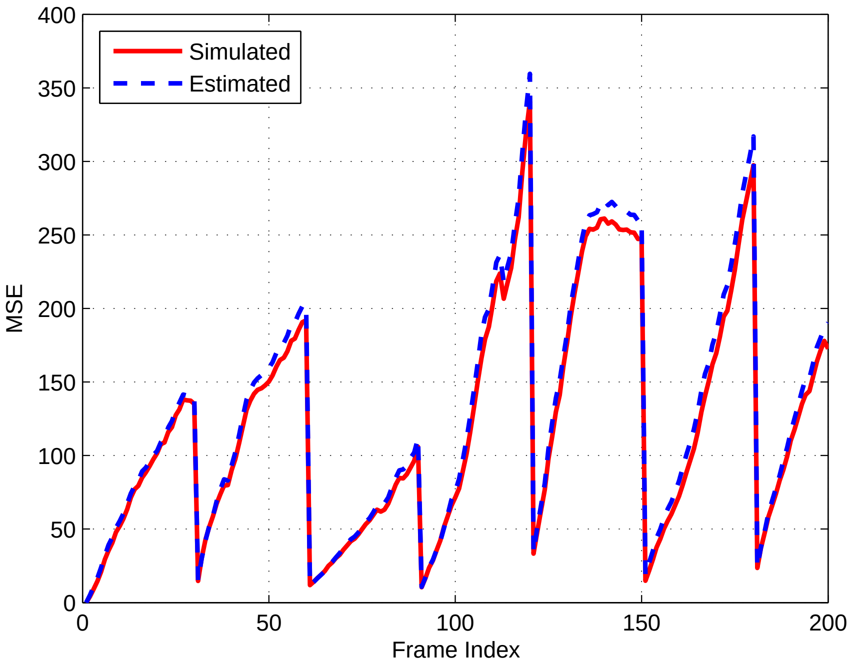


(a)

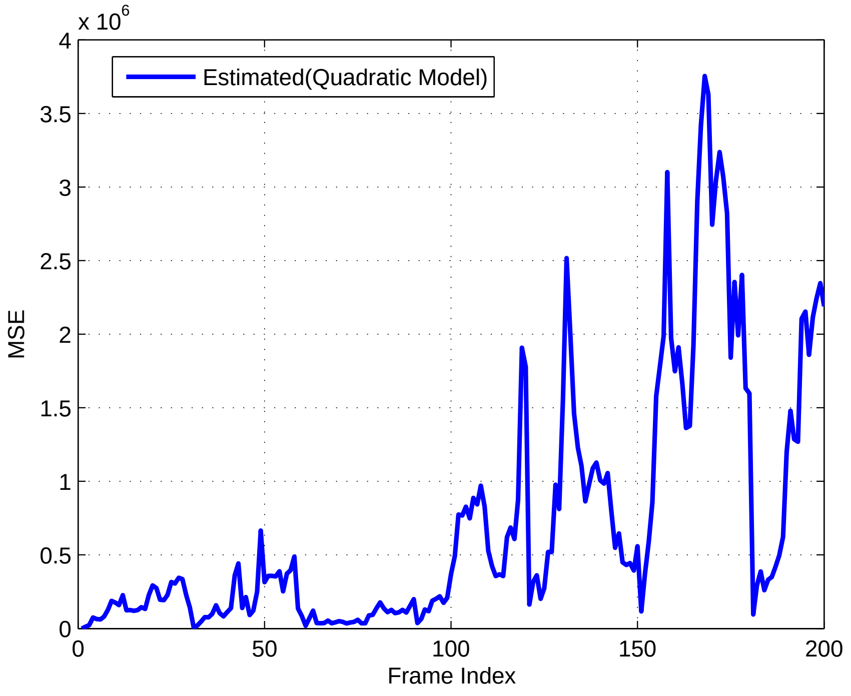


(b)

Figure 4.13: Performance comparison for Balloons(GOP=60, Packet Loss=2%). (a) Proposed graphical model and RODE. (b) Quadratic model in [41] when integrated with the RODE.

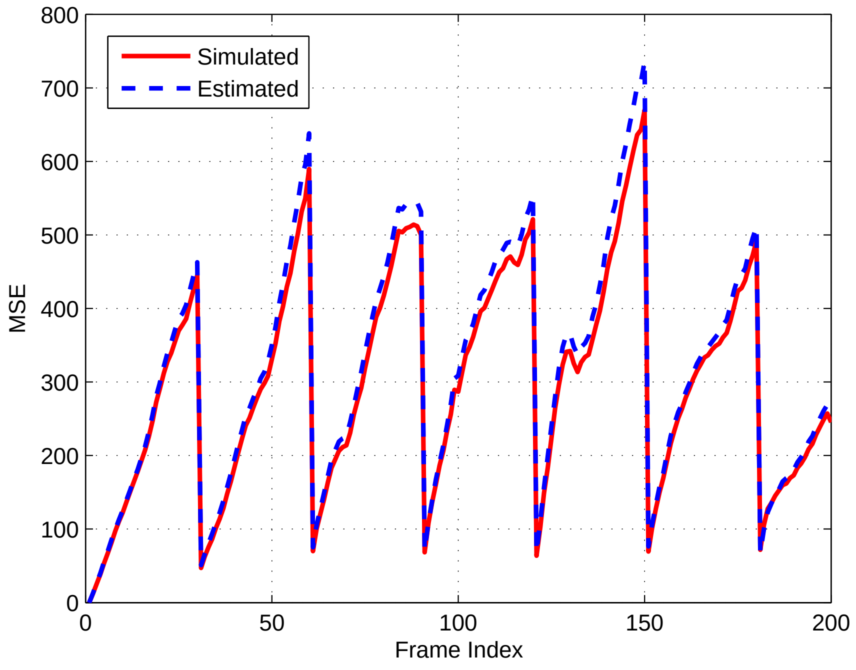


(a)

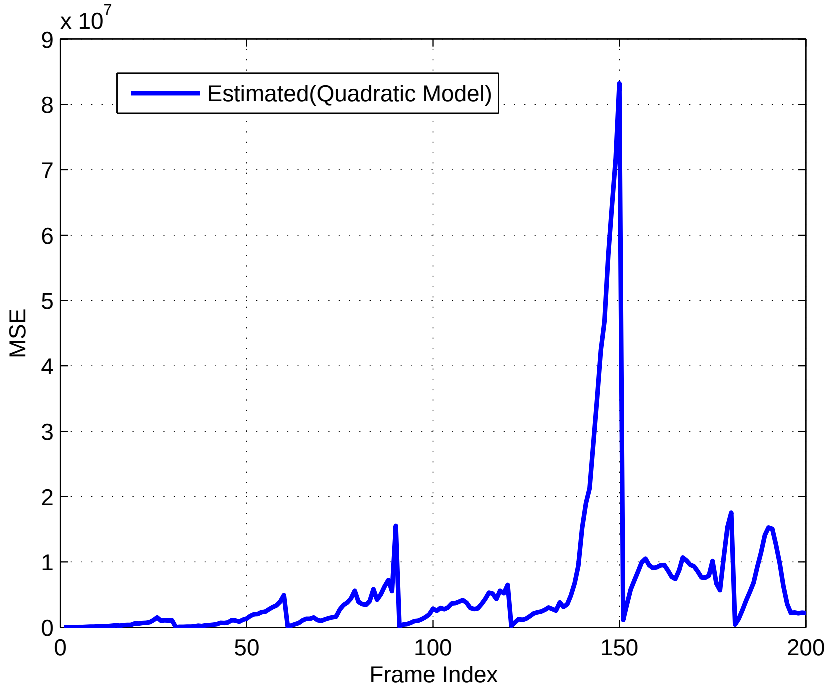


(b)

Figure 4.14: Performance comparison for Kendo(GOP=30, Packet Loss=5%). (a) Proposed graphical model and RODE. (b) Quadratic model in [41] when integrated with the RODE.



(a)



(b)

Figure 4.15: Performance comparison for Dancer(GOP=30, Packet Loss=8%). (a) Proposed graphical model and RODE. (b) Quadratic model in [41] when integrated with the RODE.

Chapter 5

Graph-based Transform for 2-D Piecewise Smooth Signals with Random Discontinuity Locations

5.1 Introduction

In this chapter, we aim to develop a single optimal graph-based transform for a class of 2-D piecewise smooth signals with similar edges or discontinuities. Each block in the class includes a discontinuity whose locations in different rows are randomly located within a confined region. The advantage is that one transform can be used for many similar blocks, thereby reducing the number of transforms that have to be computed by eigen-decomposition during the encoding and decoding. It also allows us to completely eliminate eigen-decomposition by precomputing the transforms. This work is currently being prepared for publication [82].

The reason for us to design the transform from the graph instead of the covariance matrix is that it is a lot easier to use graph to study the signals considered in this chapter. To illustrate this, consider a 2-D signal $\boldsymbol{X}$ where a sharp transition or discontinuity can randomly exist in each row within a bounded region. To find the covariance matrix for such a class of signals, suppose there are P possible geometries for the transitions within the confinement, and the occurrence of each geometry is independent from each other. Let $\boldsymbol{R}(\boldsymbol{X}|E_i)$ denote the conditional covariance matrix for $\boldsymbol{X}$ if the transition geometry is represented by E_i(for $i = 1, \ldots, P$). The covariance matrix for the entire family of $\boldsymbol{X}$ is thus given by

$$\boldsymbol{R}(\boldsymbol{X}) = \frac{1}{P}\sum_{i=1}^{P} \boldsymbol{R}(\boldsymbol{X}|E_i). \tag{5.1}$$

Each individual covariance matrix above is a dense matrix in general. Also, accounting for randomness using the covariance matrix is quite difficult. However, it is known (cf. [31])

that if we define a proper graph for the signal, then the inverse of the signal covariance matrix can approximate the Laplacian matrix of the graph very well. Since the covariance matrix and its inverse share the same eigenvectors [80], the optimal transform can thus be found from the graph Laplacian (the DCT was in fact derived from the inverse of the covariance matrix). Moreover, different from the covariance matrix, it is much easier to consider sharp discontinuities in the graph.

If all the covariance terms in Eq. (5.1) are invertible, the equation can be rewritten as

$$\boldsymbol{L}(\boldsymbol{X})^{-1} = \frac{1}{P}\sum_{i=1}^{P} \boldsymbol{L}(\boldsymbol{X}|E_i)^{-1}, \tag{5.2}$$

or equivalently,

$$\boldsymbol{L}(\boldsymbol{X}) = P\left(\sum_{i=1}^{P} \boldsymbol{L}(\boldsymbol{X}|E_i)^{-1}\right)^{-1}, \tag{5.3}$$

where $\boldsymbol{L}(\boldsymbol{X})$ is the covariance inverse of the class of signals, and $\boldsymbol{L}(\boldsymbol{X}|E_i)$ is the covariance inverse of a particular realization of the transition geometry. Both can be replaced by properly designed Laplacian matrices of a graph on the signal. Therefore it is easier to design the optimal transform for the signals of interest in this chapter using the graph theory.

The rest of the chapter is organized as follows. Sec. 5.2 derives the theory for piecewise smooth signals with known discontinuities. Sec. 5.3 generalizes it to the random case. Sec. 5.4 presents experimental results in depth image coding. Finally, Sec. 5.5 concludes our discussion. The proofs of the major theoretical results are provided in the Appendix. Preliminary results were reported in [84], without proofs and signal-dependent transform design.

5.2 Graph-based Transform for Piecewise Smooth Signals with Known Discontinuity Locations

In this section, we consider both 1-D and 2-D piecewise smooth signals with known discontinuities. We propose a special graph and an AR1 model for these signals, and show that the inverse of the covariance matrix of the signal model is equal to a biased version of the Laplacian matrix of the graph; hence we can use the graph to represent the signal, and derive the optimal transform from the eigen-decomposition of the graph Laplacian. In Sec. 5.3, we will generalize the results to piecewise smooth signals with random discontinuity locations.

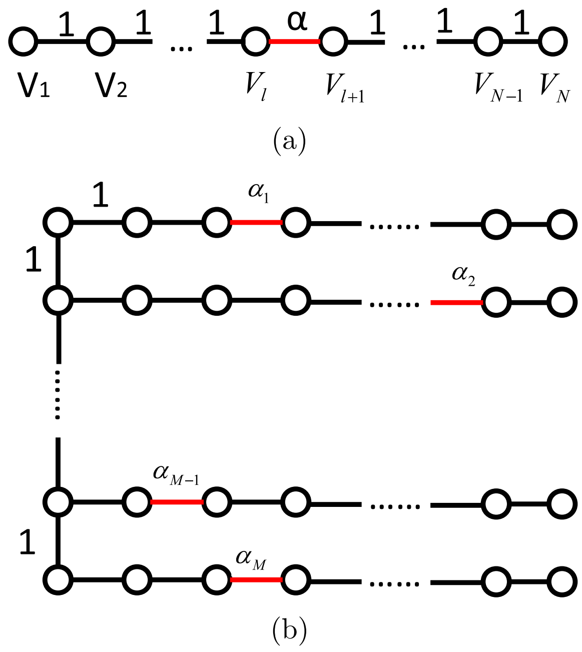


Figure 5.1: The proposed graphs for piecewise smooth signals with known discontinuity locations: (a) 1-D graph. (b) 2-D graph.

In this chapter, we call a matrx $\boldsymbol{R}$ a *layered* matrix if $R_{j,i} = R_{i,j} = R_{i,i} \triangleq R^{(i)}$ for all $j > i$. For example, a 3×3 layered matrix has the following format:

$$\boldsymbol{R} = \begin{bmatrix} R^{(1)} & R^{(1)} & R^{(1)} \\ R^{(1)} & R^{(2)} & R^{(2)} \\ R^{(1)} & R^{(2)} & R^{(3)} \end{bmatrix}. \tag{5.4}$$

5.2.1 The Inverses of the 1-D and 2-D Graph Laplacians

In this subsection, we first propose a 1-D graph for piecewise smooth signals with a known discontinuity location, and generalize it to a 2-D graph. We then derive the closed-form expressions of the inverses of the Laplacian matrices of the two graphs respectively. In Sec. 5.2.2, we will link the Laplacian inverses to the covariance matrices of properly defined 1-D and 2-D AR1 signal models. This allows us to obtain the optimal graph transform for the signals from the proposed graphs.

An undirected graph is made up of vertices with weighted edges among them. Let $\boldsymbol{D}$ be the diagonal degree matrix of the graph, whose i-th diagonal entry is the sum of the edge weights connected to the i-th vertex. The adjacency matrix $\boldsymbol{A}$ of the graph is a symmetric matrix, with $\boldsymbol{A}_{i,j} = \boldsymbol{A}_{j,i}$ being the edge weight between the i-th and the j-th vertices. The Laplacian matrix of the graph is defined as $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$.

We first consider a N-sample piecewise signal with a discontinuity between the l-th and the $(l+1)$-th samples. As in [31], we use a weighted undirected line graph with N vertices to represent this signal, where all edge weights equal to 1, except that the edge weight between the l-th and the $(l+1)$-th vertices is α. That is,

$$\boldsymbol{A}_{i,i+1} = \begin{cases} \alpha, & i = l, \\ 1, & \text{otherwise.} \end{cases} \tag{5.5}$$

The graph is illustrated in Fig. 5.1 (a). Since the sum of each row of the standard Laplacian matrix is 0, the matrix is singular. To make it invertible, we introduce a *bias* β to the first diagonal entry of the matrix, and we use $\boldsymbol{L}_\beta(l, \alpha)$ to denote this biased graph Laplacian for the signal of interest. The following result shows that the closed-form expression of the inverse of the biased graph Laplacian can be derived, and the inverse is a layered matrix as in Eq. (5.4).

Proposition 1. *The inverse of the biased graph Laplacian $\boldsymbol{L}_\beta(l, \alpha)$ for the 1-D graph in Eq. (5.5) is a layered matrix $\boldsymbol{R}_{\frac{1}{\beta}}(l, \alpha)$ whose layered entries are given by*

$$\boldsymbol{R}_{\frac{1}{\beta}}(l, \alpha)^{(i)} = \begin{cases} \frac{1}{\beta} + (i-1), & i = 1 \cdots l \\ \frac{1}{\alpha} + \frac{1}{\beta} + (i-2), & i = l+1 \cdots N. \end{cases} \tag{5.6}$$

The proof is given in Appendix 5.A.2.

A similar result can be found in [31] for the inverse of a biased line graph. However, the layered structure is not discussed. In comparison, our derivation is more general and implies that the biased line graph inverse can be easily expressed in a closed-form without performing actual inversions.

Next, we generalize the result above to 2-D signals with a known discontinuity location at each row. We propose a 2-D graph for this type of signals, which consists of M horizontal line graphs as defined in Prop. 1 that are connected vertically only at the left ends, and the vertical edge weights are all 1, as shown in Fig. 5.1 (b). It will become clear in Theorem 1 that this simple 2-D graph corresponds to our proposed 2-D AR1 model.

We use $\boldsymbol{L}_{M,\beta}(\boldsymbol{l}, \boldsymbol{\alpha})$ to denote the biased graph Laplacian of this 2-D graph, where $\boldsymbol{l} = [l_1, \ldots, l_M]$ and $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_M]$ specify the locations and edge weights of the discontinuities in all rows. β is the bias term added to the first diagonal element.

Since our 2-D graph is based on the 1-D graph, the biased graph Laplacian of the 2-D graph can be derived from that of the 1-D graph. In fact, since only the left-most vertices are connected vertically, the 2-D graph Laplacian is almost a block diagonal matrix, with some non-zero entries added to account for the vertical edges. The detailed expression of the 2-D graph Laplacian matrix is given in Eq. (5.7), where the size of $\boldsymbol{S}_{(M-i)N,N}$ is $(M-i)N \times N$, and its entries are all 0 except for a value of 1 at the first entry. The biased 2-D graph

$$\boldsymbol{L}_{M,\beta}(\boldsymbol{l},\boldsymbol{\alpha}) = \begin{bmatrix} \boldsymbol{L}_{\beta+1}(l_1,\alpha_1) & & & -\boldsymbol{S}^T_{(M-1)N,N} & & \\ & \boldsymbol{L}_2(l_2,\alpha_2) & & & -\boldsymbol{S}^T_{(M-2)N,N} & \\ & & \ddots & & \vdots & \\ -\boldsymbol{S}_{(M-1)N,N} & -\boldsymbol{S}_{(M-2)N,N} & \cdots & \boldsymbol{L}_2(l_{M-1},\alpha_{M-1}) & & -\boldsymbol{S}^T_{N,N} \\ & & & -\boldsymbol{S}_{N,N} & & \boldsymbol{L}_1(l_M,\alpha_M) \end{bmatrix}. \tag{5.7}$$

$$\boldsymbol{R}_{M,\frac{1}{\beta}}(\boldsymbol{l},\boldsymbol{\alpha}) = \begin{bmatrix} \boldsymbol{R}_{\frac{1}{\beta}}(l_1,\alpha_1) & & & \boldsymbol{C}_{(M-1)N,N}(\frac{1}{\beta})^T & & \\ & \boldsymbol{R}_{\frac{1}{\beta}+1}(l_2,\alpha_2) & & & \boldsymbol{C}_{(M-2)N,N}(\frac{1}{\beta}+1)^T & \\ & & \ddots & & \vdots & \\ \boldsymbol{C}_{(M-1)N,N}(\frac{1}{\beta}) & \boldsymbol{C}_{(M-2)N,N}(\frac{1}{\beta}+1) & \cdots & \boldsymbol{R}_{\frac{1}{\beta}+M-2}(l_{M-1},\alpha_{M-1}) & & \boldsymbol{C}_{N,N}(\frac{1}{\beta}+M-2)^T \\ & & & \boldsymbol{C}_{N,N}(\frac{1}{\beta}+M-2) & & \boldsymbol{R}_{\frac{1}{\beta}+M-1}(l_M,\alpha_M) \end{bmatrix}. \tag{5.8}$$

Laplacian matrix is constructed using properly biased line graph Laplacian matrices on the diagonal. Because the overall 2-D graph Laplacian matrix contains a bias value of β at the first entry and the line graph in the first row is connected to the second row by an edge of weight 1, the first diagonal block matrix is equivalent to a line graph Laplacian biased by $\beta+1$. Similarly, the remaining rows except the last one are connected to the top and bottom each by an edge of 1 at the left. Therefore, their corresponding entries on the diagonal are represented by line graph Laplacians biased by 2. Finally, the last row only has a single edge of 1 connected to the row above it. Therefore, the equivalent line graph Laplacian is biased by 1.

The following result shows that the inverse of the biased Laplacian matrix for the 2-D graph has a closed-form representation.

Proposition 2. *The inverse of the biased Laplacian matrix $\boldsymbol{L}_{M,\beta}(\boldsymbol{l},\boldsymbol{\alpha})$ in Eq. (5.7) for the 2-D graph in Fig. 5.1 (b) is given by $\boldsymbol{R}_{M,\frac{1}{\beta}}(\boldsymbol{l},\boldsymbol{\alpha})$ in Eq. (5.8), where $\boldsymbol{C}_{(M-i)N,N}(\frac{1}{\beta}+i-1)$, $i=1,\ldots,M$, is a constant matrix of value $\frac{1}{\beta}+i-1$ and size $(M-i)N\times N$.*

The proof is given in Appendix 5.A.3.

It can be seen from Prop. 2 that the biased 2-D graph Laplacian inverse can be easily expressed in terms of constant matrices and layered block matrices on the diagonal. In the next subsection, we will see that the covariance matrix of our proposed 2-D AR1 signal model has exactly the same structure. Consequently, this proposition serves as one of the two crucial steps in connecting the 2-D AR1 model to the 2-D graph.

5.2.2 A 2-D AR1 Model and its Covariance Matrix

Various 2-D AR models have been proposed in the past [7, 70, 91]. In this part, we first propose a 2-D AR1 model for 2-D piecewise smooth signals with known discontinuity locations, based on the 1-D AR1 model in [31]. We then prove that the covariance matrix of the model is equal to the inverse of the biased Laplacian of our proposed 2-D graph.

In [31], the covariance matrix for a 1-D AR1 model with a discontinuity was derived, as summarized by the following lemma.

Lemma 1. *[31] If a 1D signal $\boldsymbol{x} = \{x_k | k = 1, \ldots, N\}$ can be modeled by an AR1 process as follows,*

$$x_k = \begin{cases} e_1, & k = 1, \\ x_{k-1} + e_k, & 1 < k \leq N,\ k \neq l+1 \\ x_{k-1} + g + e_k, & k = l+1, \end{cases} \tag{5.9}$$

where $e_1 \sim N(0, \sigma_1^2)$, $e_k \sim N(0,1)$ (for $k = 2, \ldots, N$) are i.i.d random variables and independant of e_1, and $g \sim N(m_g, \sigma_g^2)$ is another independent random variable that models the discontinuity, then the covariance matrix of $\boldsymbol{x}$ is a layered matrix $\boldsymbol{R}_{\sigma_1^2}(l, \frac{1}{\sigma_g^2+1})$.

Next, we generalize this result to a 2-D AR1 model and show that the covariance matrix for the proposed 2-D AR1 model contains the structure defined in Eq. (5.8).

Suppose a 2-D block $\boldsymbol{X}_M$ (the subscript M is added to facilitate the induction-based proof below) has M rows of samples in a block and N samples in each row. Let $x_{i,j}$ denote the sample in the i-th row and the j-th column. We define the following 2-D AR1 model.

$$\begin{aligned} &\text{For } j = 1: \\ &x_{i,1} = \begin{cases} e_{1,1}, & i = 1, \\ x_{i-1,1} + e_{i,1}, & 1 < i \leq M, \end{cases} \\ &\text{For } 1 \leq i \leq M: \\ &x_{i,j} = \begin{cases} x_{i,j-1} + e_{i,j}, & 1 < j \leq N,\ j \neq l_i + 1\ , \\ x_{i,j-1} + g_i + e_{i,j}, & 1 < j \leq N,\ j = l_i + 1\ , \end{cases} \end{aligned} \tag{5.10}$$

where $e_{1,1} \sim N(0, \sigma_{1,1}^2)$ and $e_{i,j} \sim N(0,1)$ (for $(i,j) \neq (1,1)$) are i.i.d. Gaussian noises, and $g_i \sim N(m_{g_i}, \sigma_{g_i}^2)$ $(i = 1, \ldots, M)$ are independent Gaussian random variables that model the discontinuity between x_{i,l_i} and x_{i,l_i+1} in each row. Note that if a row does not contain a discontinuity, then m_{g_i} and $\sigma_{g_i}^2$ are both zero.

In the proposed 2-D model, each sample in each row depends on its immediate left neighbor, and each sample in the left most column depends on its immediate top neighbor. In [31], strong and weak edges receive separate treatments. In our analysis, a unified model is used to represent any type of discontinuities. Consequently, our discussion is consistent and more general. It is also pointed out in [53] that using disconnected graph to build transforms creates multiples DC components. Thus coding efficiency may be lower than using connected graphs.

To find the autocovariance matrix of $\boldsymbol{X}_M$, we need to represent $\boldsymbol{X}_M$ in terms of the noise terms. Therefore, we define a square matrix $\boldsymbol{F}$ of size $N \times N$ and its inverse as follows [31].

$$\boldsymbol{F} = \begin{bmatrix} 1 & & & \\ -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix} \boldsymbol{F}^{-1} = \begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ 1 & 1 & \ddots & \\ \vdots & \vdots & & 1 \end{bmatrix} \tag{5.11}$$

We can use F as building blocks to construct a matrix F_M of size $MN \times MN$:

$$\boldsymbol{F}_M = \begin{bmatrix} \boldsymbol{F} & 0 & & \\ -\boldsymbol{S}_{(M-1)N,N} & \boldsymbol{F} & 0 & \\ & -\boldsymbol{S}_{(M-2)N,N} & \ddots & \end{bmatrix}, \tag{5.12}$$

where $\boldsymbol{S}_{(M-i)N,N}$ are defined in Eq. (5.7).

The inverse of $\boldsymbol{F}_M$ is

$$\boldsymbol{F}_M^{-1} = \begin{bmatrix} \boldsymbol{F}^{-1} & 0 & & \\ \boldsymbol{O}_{(M-1)N,N} & \boldsymbol{F}^{-1} & 0 & \\ & \boldsymbol{O}_{(M-2)N,N} & \ddots & \end{bmatrix}, \tag{5.13}$$

where $\boldsymbol{O}_{(M-i)N,N}$ is a $(M-i)N \times N$ matrix with the first column equal to $\mathbf{1}$ and remaining entries equal to 0.

Let $\boldsymbol{x}_i$ be the i-th row of $\boldsymbol{X}_M$, and $\bar{\boldsymbol{x}}_M = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M]^T$ be the 1D representation of $\boldsymbol{X}_M$ by stacking all rows into a long vector. Similarly, let

$$\boldsymbol{b}_i = [e_{i,1}, e_{i,2}, \ldots, (e_{i,l_i+1} + g_i), \ldots, e_{i,N}], \tag{5.14}$$

and $\bar{\boldsymbol{b}}_M = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_M]^T$ be the vectorized noise.

We can then write $\boldsymbol{X}_M$ as follows based on the 2-D AR1 process in Eq. (5.10).

$$\boldsymbol{F}_M \bar{\boldsymbol{x}}_M = \bar{\boldsymbol{b}}_M, \;\; \bar{\boldsymbol{x}}_M = \boldsymbol{F}_M^{-1} \bar{\boldsymbol{b}}_M. \tag{5.15}$$

Denote the mean of each row of $\boldsymbol{X}_M$ as $\boldsymbol{u}_i = [0, \ldots, 0, m_{g_i}, \ldots, m_{g_i}]$, where m_{g_i} first appears at the $(l_i + 1)$-th entry, and the vectorized mean of $\boldsymbol{X}_M$ as $\bar{\boldsymbol{u}} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_M]^T$.

The covariance matrix of X_M can be written as

$$\begin{aligned} \boldsymbol{R}_M &= E\left\{(\bar{\boldsymbol{x}}_M - \bar{\boldsymbol{u}}_M)(\bar{\boldsymbol{x}}_M - \bar{\boldsymbol{u}}_M)^T\right\} \\ &= \boldsymbol{F}_M^{-1} E\left\{\bar{\boldsymbol{b}}_M \bar{\boldsymbol{b}}_M^T\right\} \boldsymbol{F}_M^{-T} - \bar{\boldsymbol{u}}_M \bar{\boldsymbol{u}}_M^T. \end{aligned} \tag{5.16}$$

Next, we find the covariance matrix for the proposed 2-D AR1 signal model.

Proposition 3. *If $\boldsymbol{X}_M = \{x_{i,j} | 1 \leq i \leq M, 1 \leq i \leq N\}$ follows the 2-D AR1 signal modeled in Eq. (5.10), then its covariance matrix is given by $\boldsymbol{R}_{M,\sigma_{1,1}^2}(\boldsymbol{l}, \boldsymbol{\alpha})$ defined by Eq. (5.8), where $\boldsymbol{l} = \{l_i | 1 \leq i \leq M\}$ and $\boldsymbol{\alpha} = \left\{\alpha_i = \frac{1}{\sigma_{g_i}^2+1} | 1 \leq i \leq M\right\}$.*

The proof is given in Appendix 5.A.4.

Prop. 2 and Prop. 3 proves that the inverse of the Laplacian of the proposed 2-D graph is the same as the covariance matrix of the proposed 2-D AR1 model. Therefore, we have the following theorem.

Theorem 1. *If $\boldsymbol{X}_M = \{x_{i,j} | 1 \leq i \leq M, 1 \leq i \leq N\}$ follows the 2-D AR1 modeled in Eq. (5.10), and if we define its 2-D graph as in Fig. 5.1 (b), where all the edge weights are* 1 *except that in the i-th row, the edge between x_{i,l_i} and x_{i,l_i+1} has a weight of $\frac{1}{\sigma_{g_i}^2+1}$, then the inverse of the covariance matrix of the signal is exactly the biased Laplacian of the 2-D graph, which is given by $\boldsymbol{L}_{M,\frac{1}{\sigma_{1,1}^2}}(\boldsymbol{l}, \boldsymbol{\alpha})$ with $\boldsymbol{l} = \{l_i | 1 \leq i \leq M\}$ and $\boldsymbol{\alpha} = \left\{\alpha_i = \frac{1}{\sigma_{g_i}^2+1} | 1 \leq i \leq M\right\}$.*

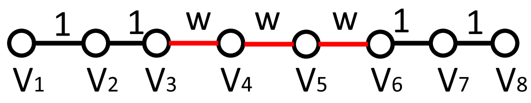
Theorem 1 allows us to use the proposed 2-D graph to precisely represent piecewise smooth signals with a known discontinuity in each row, and derive the optimal transform of the signal via the eigen-decomposition of the graph Laplacian. To the best of our knowledge, this is the first optimal graph-based transform for a 2-D AR1 signal model.

Even though the definition in Eq. (5.10) only represents a block with discontinuities extending from the top to the bottom or right boundary, by transposing and flipping (both horizontally and vertically) the input signal, the model can easily fit into any scenario where a discontinuity travels from one boundary to any of the other three block boundaries. This will be elaborated in Sec. 5.4. This is precisely the reason that the corresponding 2-D graph in Fig. 5.1 (b) are always connected by edges of 1 at the left.

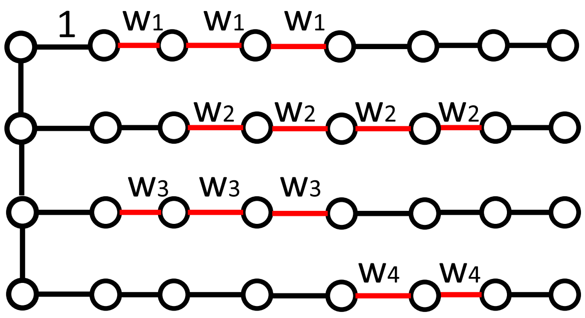
Theorem 1 also indicates that the edge weights a_i in the graph can be easily calculated from the random variables which model the discontinuities in the 2-D AR1 model. This is more convenient than relating a 2-D graph Laplacian to the precision matrix of a GMRF model where the edge weights are defined by the conditional correlations, which are very difficult to calculate in practice. An example to illustrate this argument can be found in [53].

5.3 Graph-based Transform for Piecewise Smooth Signals with Random Discontinuity Locations

In this section, based on the results in the previous section, we consider piecewise smooth signals with randomly distributed discontinuities within a bounded region. We first look at 1-D signals and then generalize to 2-D signals. In each case, we will derive the structure of the optimal graph for the signals and the corresponding closed-form expression of the Laplacian matrix, from which the optimal transform can be obtained via eigen-decomposition.



(a)



(b)

Figure 5.2: Examples of the proposed graphs for piecewise smooth signals with random discontinuity locations: (a) 1-D graph. (b) 2-D graph.

5.3.1 The 1-D Case

We consider a 1-D AR1 signal $\boldsymbol{x} = \{x_j | j = 1, \ldots, N\}$ containing a single random transition, whose location can be anywhere between x_I and x_{I+P}. We assume each possible transition can happen between x_i and x_{i+1}(for $i = I, \ldots, I+P-1$) with probability $\frac{1}{P}$. The following proposition proves the structure of the optimal graph that represents this class of random signals.

Proposition 4. *For the 1-D AR1 signal $\boldsymbol{x}$ with a random discontinuity between x_I and x_{I+P}, if the graph for each possible transition is defined as in Eq. (5.5), then the graph for the entire family of $\boldsymbol{x}$ is a line graph with all the edge values equal to 1 except that those between x_I and x_{I+P} equal to $w = \frac{P\alpha}{(P-1)\alpha+1}$. That is,*

$$\boldsymbol{A}_{i,i+1} = \begin{cases} w, & i = I, \ldots, I+P-1, \\ 1, & \textit{otherwise}. \end{cases} \tag{5.17}$$

The corresponding biased graph Laplacian is denoted as $\tilde{\boldsymbol{L}}_\beta(I, I+P, w)$.

Proof. The biased graph Laplacian for each possible transition is $\boldsymbol{L}_\beta(l,\alpha)$ ($l = I, \ldots, I + P-1$). According to Eq. (5.2), the covariance matrix for the entire random family of $\boldsymbol{x}$ is

$$\tilde{\boldsymbol{R}}_{\frac{1}{\beta}}(I, I+P) = \frac{1}{P}\sum_{l=I}^{I+P-1} \boldsymbol{L}_\beta(l,\alpha)^{-1} = \frac{1}{P}\sum_{l=I}^{I+P-1} \boldsymbol{R}_{\frac{1}{\beta}}(l,\alpha). \tag{5.18}$$

It is known from Prop. 1 that $\boldsymbol{R}_{\frac{1}{\beta}}(l, \alpha)$ is a layered matrix. Therefore the linear combination above is also a layered matrix, and the layered entries $\tilde{R}^{(i)}_{\frac{1}{\beta}}(I, I+P)$ are

$$\tilde{R}^{(i)}_{\frac{1}{\beta}}(I, I+P) = \begin{cases} \frac{1}{\beta} + (i-1), & i = 1, \dots, I \\ \frac{i-I}{P}(\frac{1}{\alpha} - 1) + \frac{1}{\beta} + (i-1), & i = I+1, \dots, I+P-1, \\ \frac{1}{\beta} + \frac{1}{\alpha} + (i-2), & i = I+P, \dots, N. \end{cases} \tag{5.19}$$

Next, we need to show that the biased graph Laplacian $\tilde{\boldsymbol{L}}_\beta(I, I+P, \frac{P\alpha}{(P-1)\alpha+1})$ is the inverse of $\tilde{\boldsymbol{R}}_{\frac{1}{\beta}}(I, I+P)$. The details are shown in Appendix 5.A.5. $\square$

An example of the graph for this type of random signals is shown in Fig. 5.2 (a).

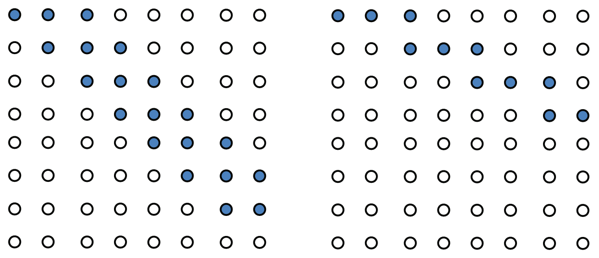
5.3.2 The 2-D Case

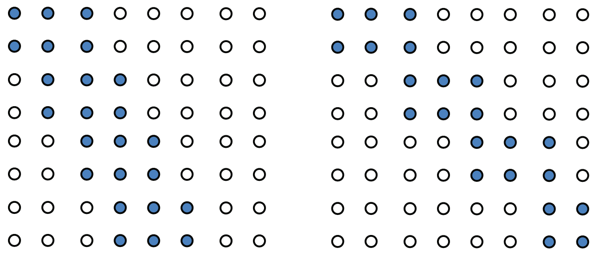
We now extend the result in Prop. 4 to 2-D signals. Suppose a 2-D signal $\boldsymbol{X} = \{x_{i,j} | i = 1, \dots, M, j = 1, \dots, N\}$ is modeled by the 2-D AR1 process in Eq. (5.10). In each row the signal has a single random transition bounded by P_i+1 (for $i = 1, \dots, M$) samples. The following theorem proves the structure of the optimal 2-D graph that represents this class of random signals.

Theorem 2. *For the 2-D AR1 signal $\boldsymbol{X}$ with a random discontinuity between I_i-th and $(I_i + P_i)$-th pixels in the i-th row, if the 1-D graph for each row is defined as in Eq. (5.17), then the biased graph for the collection of signals has M line graphs connected at the left. The edge values in the confined region in the i-th row are $w_i = \frac{P_i \alpha_i}{(P_i-1)\alpha_i+1}$, and the remaining edges are 1. The 2-D biased graph Laplacian is denoted as $\tilde{\boldsymbol{L}}_{M,\beta}(\boldsymbol{I}, \boldsymbol{I}+\boldsymbol{P}, \boldsymbol{w})$, where $\boldsymbol{I} = \{I_i | i = 1, \dots, M\}$, $\boldsymbol{P} = \{P_i | i = 1, \dots, M\}$, and $\boldsymbol{w} = \{w_i | i = 1, \dots, M\}$.*

The proof is given in Appendix 5.A.6. An example of the 2-D graph is shown in Fig. 5.2 (b).

In contrast to the conventional method in which the edge map needs to be encoded and transmitted in order for the decoder to employ the correct transform, Theorem 2 provides a very convenient way to design a univeral transform. If several confinement patterns are carefully selected such that they can encompass the majority of the edge shapes, then the coding of the edge maps can be reduced along with the complexity incurred by the eigen-decomposition. For some applications, it is also possible to completely eliminate the eigen-decomposition by precomputing all the graph transforms.


(a)


(b)

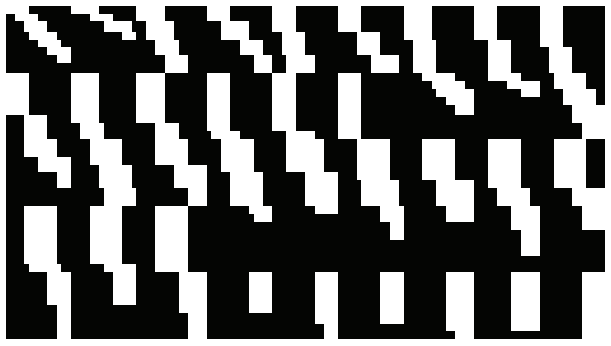

(c)

Figure 5.3: Discontinuity confinement patterns. (a) Patterns created by shifting horizontal bars of size 1×3. (b) Pattern created by shifting horizontal bars of size 2×3. (c) The 45 selected discontinuity confinement patterns.

5.4 Experimental results

In this section, we demonstrate two ways to apply the proposed graph transfrom in depth image coding. In the first approach, all graph-based transforms are pre-computed, whereas in the second method, the transforms are adaptively computed based on the input statistics.

5.4.1 Depth Coding with Pre-computed Transform

As mentioned earlier, one way to use Theorem 2 is to pre-compute the transform. Therefore, we design the following coding framework to test this method. First, some block discontinuity confinement patterns are generated in advance, and the corresponding graph transforms are obtained. During the encoding of a depth image, we first apply an edge detection algorithm to obtain its block edge maps, which are transposed or flipped and then compared with the predefined confinement patterns. If any confinement pattern bounds the edge, the corresponding transform is applied to the block. After quantization, the coefficients are encoded with the CABAC entropy coding in H.264. The transform indices for each block and the indices for the block orientation are also encoded with CABAC. At the decoder side, the decoder decodes the transform index to select the correct transform before decoding and inverse transforming the coefficients for reconstruction. If a block does not contain a discontinuity, it is coded by the DCT. If the discontinuity in the block cannot be completely captured by the confinement patterns, we choose the one with the most overlap. We then compare with the DCT. If the DCT has fewer nonzero coefficients after quantization, we revert back to the DCT.

Some high quality depth images from the Middlebury 2014 database [2] are used. For the ease of experiments, the depth images are downsampled such that the resolutions are reduced to about 500×700. The block size is chosen as 8×8.

To compare with our transform design, we choose DCT-based coding and a simplified version of the lookup table approach in [31], which produces the best rate distortion performance in depth image coding. In DCT-based approach, the encoding and decoding procedures are the same as our proposed framework, except that every block is encoded using DCT and no transform index is required.

For the lookup table approach in [31], first we use 10 training depth images to build the table as suggested. The graph isomorphism is used to find the unique common discontinuity patterns which covers 90% of all the edge blocks in the training set. Then the transforms for each common pattern are found by building 4-connected weighted graphs and stored in the table. In the training set, there are 6896 blocks with discontinuities. Graph isomorphism yields 2663 unique discontinuity patterns, out of which 1987 patterns cover 90% of the 6896 blocks. As a result, the offline table contains 1987 different transforms. During coding, edge detection algorithm is applied to the image first so that the blocks without discontinuities are transformed by DCT. The blocks with discontinuities are coded using one of the transforms from the table. To determine the optimal transform, we use the rate proxy function for the quantized AC coefficients as detailed in [31]. Therefore, during coding, the rate proxy for each transform in the lookup table is calculated. The transform which gives the lowest rate is selected. Then the image block is transformed, quantized and entropy coded using CABAC. The table index which indicates the selected transform is also entropy coded.

Note that the superior performance of the coding frame in [31] is attributed to the combination of lossless high resolution discontinuity coding, downsampling of the depth image, intra prediction and the transform design. However, our goal is to verify the theories of transform design rather than proposing a new depth image coding framework. Therefore, we simply compare our offline design to the table lookup method in [31].

We now describe a simple method to create arbitrary confinement patterns by stacking horizontal bars with various thicknesses in a staircase fashion. In Fig. 5.3(a), two examples are obtained via shifting a horizontal bar of size 1×3 by one and two pixels respectively to the right from one row to the next row. In Fig. 5.3(b), two examples are obtained by shifting a bar of size 2×3 by one and two pixels respectively. The purpose is to generate enough confinement patterns to capture discontinuities with various angles as much as possible. Fig. 5.3 (c) presents 45 selected confinement patterns generated by this process. The bright regions in each block correspond to the confinement patterns. Once the confinement patterns are found, the transform for each pattern is generated by finding the eigenvectors of the 2-D graph described by Theorem 2. Based on gathered pixel transition variance, we use $\alpha_i = 1.3636e^{-4}$ to build the 2-D graph.

The confinement patterns defined above only represent discontinuities running from the top of the block to the bottom or right boundary. Therefore, if necessary, each input block is flipped or transposed such that any discontinuity can be redirected to have the same orientation as in the confinement patterns. There are 8 possible orientations for an input block. This procedure is similar to the graph isomorphism used in [31].

5.4.2 Depth Coding with Adaptive Graph Transform

The arbitrary confinement patterns defined above are data independent. Its performance can be improved if data adaptation can be introduced. Therefore we also propose a second transform table design method by building on top of the table lookup approach in [31]. First, 1987 common discontinuity patterns are generated as described above. Then each of the common discontinuity patterns is spatially expanded to create 1987 confinement patterns. The transforms based on the confinement patterns are appended to the original lookup table to form a new expanded table with 3974 transforms. The intuition is that if the discontinuity geometry in a block is similar but not equal to the common patterns, one of the newly appended confinement patterns can still be selected due to the spatial tolerances. We also use $\alpha_i = 1.3636e^{-4}$ to build the 2-D graph for the confinement patterns.

5.4.3 Performance Comparisons

The rate-distortion performances of different transforms designs are shown in Appendix 5.B, where the method *Ref* refers to the reference table lookup method in [31], *New* refers to our proposed transform design based on expanding common patterns, *New Arb* refers to our proposed transform design using arbitrary confinement patterns, and *DCT* refers to DCT coding. We can see from plots in Appendix 5.B that our proposed training-based design is very close to the table lookup method in performance. The transform designs based on arbitrary patterns is roughly below the table lookup method by $2dB$ on average. Both the proposed transform design and the table lookup method significantly outperform DCT coding.

The complexities of different methods are tabulated in Table 5.1. Each entry in the table is the total time used to encode and decode a depth image. It can be seen that the table lookup method has significantly higher complexity than the other methods. This is due to the rate proxy function calculation used to determine the optimal transform, which is exponential as mentioned in [31]. On the other hand, our proposed transform designs use bitwise masking operations for the optimal transform selection thanks to Theorem 2. Consequently, the complexity of the coding framework can be drastically reduced. In particular, the transform based on the arbitrary patterns only has fractional complexity increase over DCT coding while still providing significant performance gain as seen in Appendix 5.B. On average, the transform designs based on the expanded common patterns and arbitrary patterns are respectively 86 times and 267 times faster than the reference table lookup method. This is very attractive to real-time depth image coding applications.

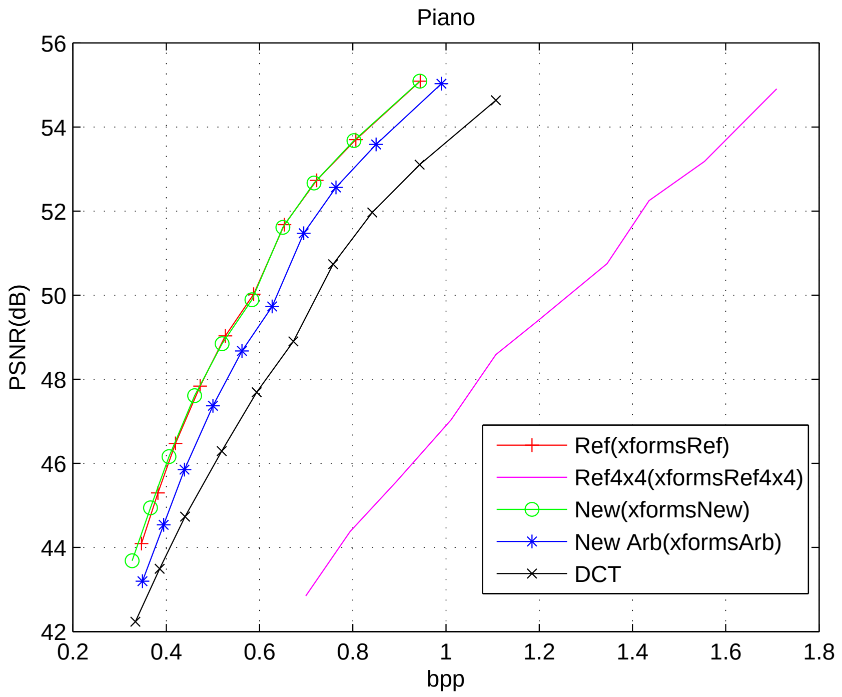


Figure 5.4: Rate distortion comparison for depth image Piano with 4×4 table lookup.

Since the complexity bottleneck of the table lookup method is in the rate proxy function, it is necessary to investigate the performance complexity tradeoff in order to find a better configuration. As noted in [31], one solution is to use a smaller size transform such as 4×4 to reduce the table size. Consequently, we repeat the same training procedure using 4×4 blocks to confirm the complexity saving. There are 13721 4×4 blocks in the training images which contain discontinuities, out of which 238 unique discontinuity patterns are found. In the 238 unique patterns, only 43 patterns are commonly used to cover 90% of all the blocks. Even though by using 4×4 blocks, the number of entries in the table is reduced

Table 5.1: Encoding and decoding times (sec.) of different transforms

Image	Ref	New	New Arb	DCT
Piano	995.71	7.82	3.97	3.26
Pipes	1623.70	14.45	3.90	3.32
Playroom	1019.30	10.74	3.39	3.00
Playtable	1052.24	11.02	3.22	2.85
Recycle	610.50	5.94	3.33	3.11
Shelves	1076.19	10.05	3.79	3.32
Shopvac	549.35	9.48	2.98	2.66
Sticks	1470.40	24.92	3.97	3.24
Storage	467.09	6.97	3.28	3.05
Sword1	1268.88	21.81	3.98	3.36
Sword2	1163.12	9.58	3.65	3.23
Umbrella	628.31	7.36	3.62	3.38
Vintage	468.26	3.85	3.34	3.18
Avg. Time	953.31	11.08	3.57	3.15
Speedup Factor	-	86	267	303

to 43, due to the reduction in transform block size, transform efficiency is expected to suffer. Furthermore, since the number of blocks is increased by a factor of 4, the bits required to signal the transform indices are roughly quadrupled. As an example, we use depth image Piano to illustrate the performance comparison in Figure 5.4. All the curves in Figure 5.4 are identical to those in Figure 5.5, except that the one labeled as *Ref4x4* corresponds to the table lookup method with 4×4 blocks. The performance decreases as expected. However, due to the smaller lookup table, the complexity of encoding and decoding the depth image Piano is reduced from 995.71s to 11.46s.

5.5 Summary

In this chapter, we consider a 2-D piecewise smooth signal class in which the location of the discontinuity is random but bounded within a region. A theory is proposed to derive the optimal graph transform for this type of signals. Depth image coding results reveal that our new transform designs outperform DCT and have comparable performance to the state-of-the-art method in the literature, and the complexity of our method is much lower, making it suitable to real time applications.

5.A Identities and Proofs

5.A.1 Matrix Inversion Formulas

In this chapter, there are a number of occasions where a partitioned matrix needs to be inverted, for which the following matrix inversion formula are used [28]. Let $\boldsymbol{L} = \boldsymbol{R}^{-1}$ and the two matrices are partitioned as follows.

$$\boldsymbol{R} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{bmatrix}, \quad \boldsymbol{L} = \begin{bmatrix} \boldsymbol{X} & \boldsymbol{Y} \\ \boldsymbol{Z} & \boldsymbol{U} \end{bmatrix}. \tag{5.20}$$

Then the submatrices of $\boldsymbol{L}$ can be obtained by

$$\begin{aligned} \boldsymbol{X} &= (\boldsymbol{A} - \boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{C})^{-1}, \\ \boldsymbol{Y} &= -\boldsymbol{A}^{-1}\boldsymbol{B}(\boldsymbol{D} - \boldsymbol{C}\boldsymbol{A}^{-1}\boldsymbol{B})^{-1}, \\ \boldsymbol{Z} &= -\boldsymbol{D}^{-1}\boldsymbol{C}(\boldsymbol{A} - \boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{C})^{-1}, \\ \boldsymbol{U} &= (\boldsymbol{D} - \boldsymbol{C}\boldsymbol{A}^{-1}\boldsymbol{B})^{-1}. \end{aligned} \tag{5.21}$$

A special case of the matrix inversion lemma is the following Woodbury matrix identity to find the inverse of a matrix $\boldsymbol{G}$ when it is adjusted by a rank-one matrix $\boldsymbol{H}$ [46], also known as Sherman-Morrison formula.

$$(\boldsymbol{G} + \boldsymbol{H})^{-1} = \boldsymbol{G}^{-1} - \frac{1}{1+g}\boldsymbol{G}^{-1}\boldsymbol{H}\boldsymbol{G}^{-1}, \tag{5.22}$$

where $g = trace(\boldsymbol{H}\boldsymbol{G}^{-1})$.

5.A.2 Proof of the inverse of the 1-D graph in Prop. 1

Proof. For the purpose of this proof, let us define a few notations. $\boldsymbol{L}_\beta(l, \alpha, N)$ and $\boldsymbol{R}_{\frac{1}{\beta}}(l, \alpha, N)$ denote the biased 1-D graph Laplacian with N vertices and a layered matrix of size $N \times N$ respectively. Let $\boldsymbol{A}^\tau$ denote the matrix transpose of $\boldsymbol{A}$ along the anti-diagonal. Let $\boldsymbol{A}^\rightarrow$ denote flipping matrix $\boldsymbol{A}$ horizontally. In this proof, we will show that $\boldsymbol{L}_\beta(l, \alpha, N)\boldsymbol{R}_{\frac{1}{\beta}}(l, \alpha, N) = \boldsymbol{I}$.

This proof is based on induction. We first show that Prop. 1 is true for $N = 2$. This is the base case. When $N = 2$, there is only a single possible edge. Consequently, the inverse relationship is easily verified as follows.

$$\boldsymbol{L}_\beta(1, \alpha, 2)\boldsymbol{R}_{\frac{1}{\beta}}(1, \alpha, 2) = \begin{bmatrix} \beta + \alpha & -\alpha \\ -\alpha & \alpha \end{bmatrix} \begin{bmatrix} \frac{1}{\beta} & \frac{1}{\beta} \\ \frac{1}{\beta} & \frac{1}{\alpha} + \frac{1}{\beta} \end{bmatrix} = \boldsymbol{I} \tag{5.23}$$

For the induction step, we assume that Prop. 1 is true at $N = K$ and we need to show that it is also true at $N = K + 1$. By the inductive assumption, we have

$\boldsymbol{L}_\beta(l,\alpha,K)\boldsymbol{R}_{\frac{1}{\beta}}(l,\alpha,K) = \boldsymbol{I}$ for $l = 1\ldots K-1$. At $N = K+1$, we need to consider two cases.

Case 1: The α valued edge is located within the first K vertices.

Based on the definition, $\boldsymbol{L}_\beta(l,\alpha,K+1)$ and $\boldsymbol{R}_{\frac{1}{\beta}}(l,\alpha,K+1)$ can be expressed as follows.

$$\boldsymbol{L}_\beta(l,\alpha,K+1) = \boldsymbol{\Lambda} + \begin{bmatrix} \boldsymbol{L}_\beta(l,\alpha,K) & \boldsymbol{0} \\ \boldsymbol{0} & 0 \end{bmatrix} \tag{5.24}$$

where $\boldsymbol{\Lambda} = \begin{bmatrix} \boldsymbol{S}^{\tau}_{K,K} & -(\boldsymbol{S}^{\rightarrow}_{1,K})^T \\ -\boldsymbol{S}^{\rightarrow}_{1,K} & 1 \end{bmatrix}$.

$$\boldsymbol{R}_{\frac{1}{\beta}}(l,\alpha,K+1) = \begin{bmatrix} \boldsymbol{R}_{\frac{1}{\beta}}(l,\alpha,K) & \boldsymbol{\Theta}^T \\ \boldsymbol{\Theta} & \frac{1}{\alpha}+\frac{1}{\beta}+K-1 \end{bmatrix} \tag{5.25}$$

where $\boldsymbol{\Theta}$ is equal to the last row of $\boldsymbol{R}_{\frac{1}{\beta}}(l,\alpha,K)$, or equavalently $\boldsymbol{\Theta}^T$ is equal to the last column of $\boldsymbol{R}_{\frac{1}{\beta}}(l,\alpha,K)$.

Due to the inductive assumption, we have $\boldsymbol{L}_\beta(l,\alpha,K)\boldsymbol{R}_{\frac{1}{\beta}}(l,\alpha,K) = \boldsymbol{I}$ and $\boldsymbol{L}_\beta(l,\alpha,K)\boldsymbol{\Theta}^T = (\boldsymbol{S}^{\rightarrow}_{1,K})^T$. Consequently, we have the following.

$$\begin{bmatrix} \boldsymbol{L}_\beta(l,\alpha,K) & \boldsymbol{0} \\ \boldsymbol{0} & 0 \end{bmatrix} \boldsymbol{R}_{\frac{1}{\beta}}(l,\alpha,K+1) = \begin{bmatrix} \boldsymbol{I} & (\boldsymbol{S}^{\rightarrow}_{1,K})^T \\ \boldsymbol{0} & 0 \end{bmatrix} \tag{5.26}$$

$$\boldsymbol{\Lambda}\boldsymbol{R}_{\frac{1}{\beta}}(l,\alpha,K+1) = \begin{bmatrix} \boldsymbol{0} & -(\boldsymbol{S}^{\rightarrow}_{1,K})^T \\ \boldsymbol{0} & 1 \end{bmatrix} \tag{5.27}$$

As a result, by summing Eq. (5.26) and Eq. (5.27) we have $\boldsymbol{L}_\beta(l,\alpha,K+1)\boldsymbol{R}_{\frac{1}{\beta}}(l,\alpha,K+1) = \boldsymbol{I}$ when the α valued edge is located within the first K vertices.

Case 2: The α valued edge is located between the K-th and $(K+1)$-th vertices.

In this case we can write $\boldsymbol{L}_\beta(K,\alpha,K+1)$ and $\boldsymbol{R}_{\frac{1}{\beta}}(K,\alpha,K+1)$ in a similar way.

$$\boldsymbol{L}_\beta(K,\alpha,K+1) = \alpha\boldsymbol{\Lambda} + \begin{bmatrix} \boldsymbol{L}_\beta(l,1,K) & \boldsymbol{0} \\ \boldsymbol{0} & 0 \end{bmatrix} \tag{5.28}$$

$$\boldsymbol{R}_{\frac{1}{\beta}}(K,\alpha,K+1) = \begin{bmatrix} \boldsymbol{R}_{\frac{1}{\beta}}(l,1,K) & \boldsymbol{\Theta}'^T \\ \boldsymbol{\Theta}' & \frac{1}{\alpha}+\frac{1}{\beta}+K-1 \end{bmatrix} \tag{5.29}$$

where $\boldsymbol{\Theta}'$ is equal to the last row of $\boldsymbol{R}_{\frac{1}{\beta}}(l,1,K)$ and the value for l is arbitrary.

Similar to the derivation for case 1, we have the following based on the inductive assumption.

$$\begin{bmatrix} \boldsymbol{L}_{\beta}(l,1,K) & \boldsymbol{0} \\ \boldsymbol{0} & 0 \end{bmatrix} \boldsymbol{R}_{\frac{1}{\beta}}(K,\alpha,K+1) = \begin{bmatrix} \boldsymbol{I} & (\boldsymbol{S}^{\rightarrow}_{1,K})^T \\ \boldsymbol{0} & 0 \end{bmatrix} \tag{5.30}$$

$$\alpha \boldsymbol{\Lambda} \boldsymbol{R}_{\frac{1}{\beta}}(K,\alpha,K+1) = \begin{bmatrix} \boldsymbol{0} & -(\boldsymbol{S}^{\rightarrow}_{1,K})^T \\ \boldsymbol{0} & 1 \end{bmatrix} \tag{5.31}$$

Consequently, we have $\boldsymbol{L}_{\beta}(K,\alpha,K+1)\boldsymbol{R}_{\frac{1}{\beta}}(K,\alpha,K+1) = \boldsymbol{I}$ and the proof is completed.

□

5.A.3 Proof of the inverse of the 2-D graph in Prop. 2

Proof. The proof is based on induction. First, we need to show that Prop. 2 is true for $M = 1$. In other words, we need to show that $\boldsymbol{L}_{\beta}(\boldsymbol{l},\boldsymbol{\alpha})^{-1} = \boldsymbol{R}_{\frac{1}{\beta}}(\boldsymbol{l},\boldsymbol{\alpha})$. This has already been proven in Prop. 1. Hence, the base case is true.

Next, for the inductive step, we assume that Prop. 2 is true for $M = K$ and we need to demonstrate that the proposition holds for $M = K+1$. Consequently, we can construct a 2-D graph Laplacian with $K+1$ lines by adding an arbitrary line graph with Laplacian equal to $\boldsymbol{L}_{\beta+1}(l_1,\alpha_1)$ to the existing K line graph Laplacian $\boldsymbol{L}_{K,1}(\boldsymbol{l}',\boldsymbol{\alpha}')$ as follows.

$$\boldsymbol{L}_{K+1,\beta}(\boldsymbol{l},\boldsymbol{\alpha}) = \begin{bmatrix} \boldsymbol{L}_{\beta+1}(l_1,\alpha_1) & -\boldsymbol{S}^T_{KN,N} \\ -\boldsymbol{S}_{KN,N} & \boldsymbol{L}_{K,1}(\boldsymbol{l}',\boldsymbol{\alpha}') \end{bmatrix} \tag{5.32}$$

It should be noted that the bias values for the sub-block Laplacians are chosen such that the augmented matrix is a valid biased graph Laplacian matrix by our definition.

Similarly, the covariance matrix for the $K+1$ line graph is claimed as follows according to Eq. (5.8).

$$\boldsymbol{R}_{K+1,\frac{1}{\beta}}(\boldsymbol{l},\boldsymbol{\alpha}) = \begin{bmatrix} \boldsymbol{R}_{\frac{1}{\beta}}(l_1,\alpha_1) & \boldsymbol{C}_{KN,N}(\frac{1}{\beta})^T \\ \boldsymbol{C}_{KN,N}(\frac{1}{\beta}) & \boldsymbol{R}_{K,\frac{1}{\beta}+1}(\boldsymbol{l}',\boldsymbol{\alpha}') \end{bmatrix} \tag{5.33}$$

The goal of the proof is to show that the Laplacian and the covariance matrices are inverse of each other at $M = K+1$. Their product is denoted by a matrix with 4 submatrices $\boldsymbol{X}$, $\boldsymbol{Y}$, $\boldsymbol{Z}$ and $\boldsymbol{U}$.

$$\boldsymbol{L}_{K+1,\beta}(\boldsymbol{l},\boldsymbol{\alpha})\boldsymbol{R}_{K+1,\frac{1}{\beta}}(\boldsymbol{l},\boldsymbol{\alpha}) = \begin{bmatrix} \boldsymbol{X} & \boldsymbol{Y} \\ \boldsymbol{Z} & \boldsymbol{U} \end{bmatrix} \tag{5.34}$$

The block matrix $\boldsymbol{X}$ is found as follows.

$$\begin{aligned}\boldsymbol{X} &= \boldsymbol{L}_{\beta+1}(l_1,\alpha_1)\boldsymbol{R}_{\frac{1}{\beta}}(l_1,\alpha_1) - \boldsymbol{S}^T_{KN,N}\boldsymbol{C}_{KN,N}(\frac{1}{\beta}) \\ &= \left(\boldsymbol{S}_{N,N} + \boldsymbol{L}_\beta(l_1,\alpha_1)\right)\boldsymbol{R}_{\frac{1}{\beta}}(l_1,\alpha_1) - \frac{1}{\beta}\boldsymbol{O}^T_{N,N}\end{aligned} \tag{5.35}$$

By Prop. 1, we have $\boldsymbol{L}_\beta(l_1,\alpha_1)\boldsymbol{R}_{\frac{1}{\beta}}(l_1,\alpha_1) = \boldsymbol{I}$. In addition, Prop. 1 tells us that the first row of $\boldsymbol{R}_{\frac{1}{\beta}}(l_1,\alpha_1)$ is $\frac{1}{\beta}$. Therefore, we have

$$\boldsymbol{S}_{N,N}\boldsymbol{R}_{\frac{1}{\beta}}(l_1,\alpha_1) = \frac{1}{\beta}\boldsymbol{O}^T_{N,N} \tag{5.36}$$

Consequently, $\boldsymbol{X} = \boldsymbol{I}$.

Similarly, the block matrix $\boldsymbol{U}$ is found as follows.

$$\begin{aligned}\boldsymbol{U} &= -\boldsymbol{S}_{KN,N}\boldsymbol{C}_{KN,N}(\frac{1}{\beta})^T + \boldsymbol{L}_{K,1}(\boldsymbol{l}',\boldsymbol{\alpha}')\boldsymbol{R}_{K,\frac{1}{\beta}+1}(\boldsymbol{l}',\boldsymbol{\alpha}') \\ &= -\frac{1}{\beta}\boldsymbol{O}^T_{KN,KN} \\ &+ \left(\boldsymbol{L}_{K,\frac{\beta}{1+\beta}}(\boldsymbol{l}',\boldsymbol{\alpha}') + \frac{1}{1+\beta}\boldsymbol{S}_{KN,KN}\right)\boldsymbol{R}_{K,\frac{1}{\beta}+1}(\boldsymbol{l}',\boldsymbol{\alpha}')\end{aligned} \tag{5.37}$$

By the assumption at the inductive step, we have $\boldsymbol{L}_{K,\frac{\beta}{1+\beta}}(\boldsymbol{l}',\boldsymbol{\alpha}')\boldsymbol{R}_{K,\frac{1}{\beta}+1}(\boldsymbol{l}',\boldsymbol{\alpha}') = \boldsymbol{I}$. Furthermore, by our definition the first row of $\boldsymbol{R}_{K,\frac{1}{\beta}+1}(\boldsymbol{l}',\boldsymbol{\alpha}')$ is $\frac{1}{\beta}+1$, we have

$$\frac{1}{1+\beta}\boldsymbol{S}_{KN,KN}\boldsymbol{R}_{K,\frac{1}{\beta}+1}(\boldsymbol{l}',\boldsymbol{\alpha}') = \frac{1}{\beta}\boldsymbol{O}^T_{KN,KN} \tag{5.38}$$

Consequently we have $\boldsymbol{U} = \boldsymbol{I}$.

Block matrix $\boldsymbol{Y}$ is found as follows.

$$\boldsymbol{Y} = \boldsymbol{L}_{\beta+1}(l_1,\alpha_1)\boldsymbol{C}_{KN,N}(\frac{1}{\beta})^T - \boldsymbol{S}^T_{KN,N}\boldsymbol{R}_{K,\frac{1}{\beta}+1}(\boldsymbol{l}',\boldsymbol{\alpha}') \tag{5.39}$$

Due to the fact that the element at the first row and first column in $\boldsymbol{L}_{\beta+1}(l_1,\alpha_1)$ is biased by $\beta+1$ and the remaining rows all sum to 0, we have

$$\boldsymbol{L}_{\beta+1}(l_1,\alpha_1)\boldsymbol{C}_{KN,N}(\frac{1}{\beta})^T = \frac{\beta+1}{\beta}\boldsymbol{O}^T_{KN,N} \tag{5.40}$$

In addition, the first row of $\boldsymbol{R}_{K,\frac{1}{\beta}+1}(\boldsymbol{l}',\boldsymbol{\alpha}')$ is $\frac{1}{\beta}+1$. We have

$$-\boldsymbol{S}^T_{KN,N}\boldsymbol{R}_{K,\frac{1}{\beta}+1}(\boldsymbol{l}',\boldsymbol{\alpha}') = -\frac{\beta+1}{\beta}\boldsymbol{O}^T_{KN,N} \tag{5.41}$$

Consequently, we have $\boldsymbol{Y} = 0$. $\boldsymbol{Z} = 0$ can be shown in a similar way.

As a consequence, $\boldsymbol{L}_{K+1,\beta}(\boldsymbol{l},\boldsymbol{\alpha})\boldsymbol{R}_{K+1,\frac{1}{\beta}}(\boldsymbol{l},\boldsymbol{\alpha}) = \boldsymbol{I}$. The inductive step is completed. $\square$

5.A.4 Proof of the Covariance of the 2-D AR1 model in Prop. 3

Proof. For this proof, let us define two functions $f_a(\boldsymbol{X}, N)$ and $g_a(\boldsymbol{X}_i, N)$. $f_a(\boldsymbol{X}, N)$ horizontally augments N identical matrices $\boldsymbol{X}$. $g_a(\boldsymbol{X}_i, N)$ horizontally augments N distinct matrices $\boldsymbol{X}_i$ for $i = 1 \dots N$ with $\boldsymbol{X}_1$ being the left most block matrix and $\boldsymbol{X}_N$ being the right most block matrix.

The proof is based on induction. First, it should be noted that the base case where $M = 1$ is shown to be true by Lemma. 1.

Next, for the induction step, we assume Prop. 3 is true for $M = K$ and we need to prove that it is true for $M = K + 1$. The covariance matrix with $K + 1$ rows can be written as follows.

$$\boldsymbol{R}_{K+1} = \boldsymbol{F}_{K+1}^{-1} E\left\{\bar{\boldsymbol{b}}_{K+1}\bar{\boldsymbol{b}}_{K+1}^T\right\}\left(\boldsymbol{F}_{K+1}^{-1}\right)^T - \bar{\boldsymbol{u}}_{K+1}\bar{\boldsymbol{u}}_{K+1}^T \tag{5.42}$$

By observing the structure of the covariance matrix we can express the three unique terms $\boldsymbol{F}_{K+1}^{-1}$, $E\left\{\bar{\boldsymbol{b}}_{K+1}\bar{\boldsymbol{b}}_{K+1}^T\right\}$ and $\bar{\boldsymbol{u}}_{K+1}\bar{\boldsymbol{u}}_{K+1}^T$ in $\boldsymbol{R}_{K+1}$ using sub-matrices.

$$\boldsymbol{F}_{K+1}^{-1} = \begin{bmatrix} \boldsymbol{F}_K^{-1} & 0 \\ f_a(\boldsymbol{O}_{N,N}, K) & \boldsymbol{F}^{-1} \end{bmatrix} \tag{5.43}$$

$$E\left\{\bar{\boldsymbol{b}}_{K+1}\bar{\boldsymbol{b}}_{K+1}^T\right\} = \begin{bmatrix} E\left\{\bar{\boldsymbol{b}}_K\bar{\boldsymbol{b}}_K^T\right\} & g_a(\boldsymbol{V}_i, K)^T \\ g_a(\boldsymbol{V}_i, K) & E\left\{\boldsymbol{b}\boldsymbol{b}^T\right\} \end{bmatrix} \tag{5.44}$$

where $\boldsymbol{V}_i$ is a $N \times N$ matrix with $m_{g_{K+1}} m_{g_i}$ at the $(l_{K+1}+1)$-th row and (l_i+1)-th column, the remaining entries are 0.

$$\bar{\boldsymbol{u}}_{K+1}\bar{\boldsymbol{u}}_{K+1}^T = \begin{bmatrix} \bar{\boldsymbol{u}}_K\bar{\boldsymbol{u}}_K^T & \bar{\boldsymbol{u}}_K\boldsymbol{u}^T \\ \boldsymbol{u}\bar{\boldsymbol{u}}_K^T & \boldsymbol{u}\boldsymbol{u}^T \end{bmatrix} \tag{5.45}$$

The matrices $\boldsymbol{u}$ and $\boldsymbol{b}$ belong to the 1D signal added in the induction step to the 2-D signal with K rows and are the 1D counterparts to $\bar{\boldsymbol{u}}_K$ and $\bar{\boldsymbol{b}}_K$.

After multiplying the sub-matrices, the covariance matrix $\boldsymbol{R}_{K+1}$ is given as follow.

$$\boldsymbol{R}_{K+1} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{bmatrix} \tag{5.46}$$

Sub-matrix $\boldsymbol{A}$ is given as follows.

$$\boldsymbol{A} = \boldsymbol{F}_K^{-1} E\left\{\bar{\boldsymbol{b}}_K\bar{\boldsymbol{b}}_K^T\right\}\left(\boldsymbol{F}_K^{-1}\right)^T - \bar{\boldsymbol{u}}_K\bar{\boldsymbol{u}}_K^T \tag{5.47}$$

By the assumption for the inductive step $\boldsymbol{A} = \boldsymbol{R}_{K,\sigma_{1,1}^2}(\boldsymbol{l},\boldsymbol{\alpha})$

Sub-matrices $\boldsymbol{B}$ and $\boldsymbol{C}$ are transpose of each other.

$$\begin{aligned}
&\boldsymbol{B}^T = \boldsymbol{C} = f_a(\boldsymbol{O}_{N,N},K)E\left\{\bar{\boldsymbol{b}}_K\bar{\boldsymbol{b}}_K^T\right\}\left(\boldsymbol{F}_K^{-1}\right)^T + \\
&\boldsymbol{F}^{-1}g_a(\boldsymbol{V}_i,K)\left(\boldsymbol{F}_K^{-1}\right)^T - \boldsymbol{u}\bar{\boldsymbol{u}}_K^T \\
&= g_a(\boldsymbol{\Omega}_i,K) + g_a(\boldsymbol{\Psi}_i,K) - g_a(\boldsymbol{\Psi}_i,K) = g_a(\boldsymbol{\Omega}_i,K)
\end{aligned} \tag{5.48}$$

where $\boldsymbol{\Omega}_i$ is a constant matrix of size $N \times N$ equal to $\sum_{j=1}^{i}\sigma_{j,1}^2 = \sigma_{1,1}^2 + i - 1$ and $\boldsymbol{\Psi}_i$ is a $N \times N$ matrix with a constant sub-matrix at $(l_{K+1}+1)$-th row and (l_i+1)-th column (for $i = 1,\ldots,K$). The constant sub-matrix is equal to $m_{g_{k+1}}m_{g_i}$ and remaining entries in $\boldsymbol{\Psi}_i$ are equal to 0.

Sub-matrix $\boldsymbol{D}$ is found as follows.

$$\begin{aligned}
\boldsymbol{D} &= f_a(\boldsymbol{O}_{N,N},K)E\left\{\bar{\boldsymbol{b}}_K\bar{\boldsymbol{b}}_K^T\right\}f_a(\boldsymbol{O}_{N,N},K)^T \\
&+ \boldsymbol{F}^{-1}g_a(\boldsymbol{V}_i,K)f_a(\boldsymbol{O}_{N,N},K)^T \\
&+ f_a(\boldsymbol{O}_{N,N},K)g_a(\boldsymbol{V}_i,K)^T\left(\boldsymbol{F}^{-1}\right)^T \\
&+ \boldsymbol{F}^{-1}E\left\{\boldsymbol{b}\boldsymbol{b}^T\right\}\left(\boldsymbol{F}^{-1}\right)^T - \boldsymbol{u}\boldsymbol{u}^T
\end{aligned} \tag{5.49}$$

By multiplying the sub-matrices, we arrive at the following results.

$$\begin{aligned}
&f_a(\boldsymbol{O}_{N,N},K)E\left\{\bar{\boldsymbol{b}}_K\bar{\boldsymbol{b}}_K^T\right\}f_a(\boldsymbol{O}_{N,N},K)^T = \boldsymbol{\Omega}_K \\
&\boldsymbol{F}^{-1}g_a(\boldsymbol{V}_i,K)f_a(\boldsymbol{O}_{N,N},K)^T = 0 \\
&f_a(\boldsymbol{O}_{N,N},K)g_a(\boldsymbol{V}_i,K)^T\left(\boldsymbol{F}^{-1}\right)^T = 0
\end{aligned} \tag{5.50}$$

Consequently, sub-matrix $\boldsymbol{D}$ can be written as follows.

$$\boldsymbol{D} = \boldsymbol{\Omega}_K + \boldsymbol{F}^{-1}E\left\{\boldsymbol{b}\boldsymbol{b}^T\right\}\left(\boldsymbol{F}^{-1}\right)^T - \boldsymbol{u}\boldsymbol{u}^T \tag{5.51}$$

Furthermore, according to Lemma. 1, $\boldsymbol{F}^{-1}E\left\{\boldsymbol{b}\boldsymbol{b}^T\right\}\left(\boldsymbol{F}^{-1}\right)^T - \boldsymbol{u}\boldsymbol{u}^T = \boldsymbol{R}_1(l_{K+1},\alpha_{K+1})$ so $\boldsymbol{D} = \boldsymbol{\Omega}_K + \boldsymbol{R}_1(l_{K+1},\alpha_{K+1}) = \boldsymbol{R}_{\sigma_{1,1}^2+K}(l_{K+1},\alpha_{K+1})$

As a result we have

$$\boldsymbol{R}_{K+1} = \begin{bmatrix} \boldsymbol{R}_{K,\sigma_{1,1}^2}(\boldsymbol{l},\boldsymbol{\alpha}) & g_a(\boldsymbol{\Omega}_i,K)^T \\ g_a(\boldsymbol{\Omega}_i,K) & \boldsymbol{R}_{\sigma_{1,1}^2+K}(l_{K+1},\alpha_{K+1}) \end{bmatrix} \tag{5.52}$$

which corresponds to the definition of the covariance matrix in Eq. (5.8) when $M = K+1$. Consequently, the proof is completed. $\square$

5.A.5 Proof of the inverse of the 1-D random graph in Prop. 4

First we prove two special cases in Prop. 4.

Special case 1: There is only a single possible transition location bounded by the confinement.

In this special case, $P = 1$. Therefore, Prop. 4 is reduced to Prop. 1, which is previously proven.

Special case 2: The confinement region includes all the edges.

In this special case, if there are N vertices in the graph, then there are $N-1$ possible edge locations, therefore $P = N-1$ and $w = \frac{(N-1)\alpha}{(N-2)\alpha+1}$. The corresponding graph Laplacian $\tilde{\boldsymbol{L}}_\beta(1,N,w)$ and its inverse can be written as follows.

$$\begin{aligned}\tilde{\boldsymbol{L}}_\beta(1,N,w) &= \frac{(N-1)\alpha}{(N-2)\alpha+1}\boldsymbol{L}_{\beta(\frac{(N-2)\alpha+1}{(N-1)\alpha})}(l,1)\\ \tilde{\boldsymbol{L}}_\beta(1,N,w)^{-1} &= \frac{(N-2)\alpha+1}{(N-1)\alpha}\boldsymbol{L}_{\beta(\frac{(N-2)\alpha+1}{(N-1)\alpha})}(l,1)^{-1}\end{aligned} \tag{5.53}$$

where $\boldsymbol{L}_{\beta(\frac{(N-2)\alpha+1}{(N-1)\alpha})}(l,1)$ is the biased line graph Laplacian in Prop. 1 with l taking an arbitrary value in the range of $[1\ldots N-1]$. It should be noted that due to the scaling factor $\frac{(N-1)\alpha}{(N-2)\alpha+1}$, the bias value becomes $\beta(\frac{(N-2)\alpha+1}{(N-1)\alpha})$.

According to Prop. 1, $\boldsymbol{L}_{\beta(\frac{(N-2)\alpha+1}{(N-1)\alpha})}(l,1)^{-1}$ can be expressed as a layered matrix. Consequently, $\tilde{\boldsymbol{L}}_\beta(1,N,w)^{-1}$ is also a layered matrix which can be easily verified to be equal to $\tilde{\boldsymbol{R}}_{\frac{1}{\beta}}(I, I+P)$.

To finish the rest of the proof, we use the results from the two special cases and induction. Let $\tilde{\boldsymbol{L}}_\beta(I, I+P, w, N)$ denote a 1-D random graph Laplacian with N vertices and $\tilde{\boldsymbol{R}}_{\frac{1}{\beta}}(I, I+P, N)$ its corresponding inverse.

For the base case, we consider a graph with 3 vertices. When $N = 3$, the two special cases cover all the possible confinement patterns. Therefore, the base case is true.

For the inductive step, we assume Prop. 4 is true for $N = K$. At $N = K+1$, we need to consider 3 cases.

Inductive case 1: The confinement pattern includes the edges in the first K vertices.

We have the following expressions for $\tilde{\boldsymbol{L}}_\beta(I, I+P, w, K+1)$ and $\tilde{\boldsymbol{R}}_{\frac{1}{\beta}}(I, I+P, K+1)$.

$$\tilde{\boldsymbol{L}}_\beta(I, I+P, w, K+1) = \boldsymbol{\Lambda} + \begin{bmatrix} \tilde{\boldsymbol{L}}_\beta(I, I+P, w, K) & \boldsymbol{0} \\ \boldsymbol{0} & 0 \end{bmatrix} \tag{5.54}$$

where $\boldsymbol{\Lambda}$ is defined in the proof for Prop. 1 in Appendix 5.A.2

$$\tilde{\boldsymbol{R}}_{\frac{1}{\beta}}(I, I+P, K+1) = \begin{bmatrix} \tilde{\boldsymbol{R}}_{\frac{1}{\beta}}(I, I+P, K) & \boldsymbol{\Delta}^T \\ \boldsymbol{\Delta} & \frac{1}{\alpha}+\frac{1}{\beta}+K-1 \end{bmatrix} \tag{5.55}$$

where $\boldsymbol{\Delta}$ is equal to the last row of $\tilde{\boldsymbol{R}}_{\frac{1}{\beta}}(I, I+P, K)$. Similar to the proof for Prop. 1, it can be shown that by multiplying the submatrices and utilizing the inductive assumption $\tilde{\boldsymbol{L}}_{\beta}(I, I+P, w, K+1)\tilde{\boldsymbol{R}}_{\frac{1}{\beta}}(I, I+P, K+1) = \boldsymbol{I}$.

Inductive case 2: The confinement pattern includes the edges in the last K vertices.

In this case, we have the following expressions.

$$\tilde{\boldsymbol{L}}_{\beta}(I, I+P, w, K+1) = \left[\begin{array}{cc} \beta+1 & -\boldsymbol{S}_{1,K} \\ -\boldsymbol{S}_{K,1} & \tilde{\boldsymbol{L}}_{1}(I, I+P, w, K) \end{array}\right] \tag{5.56}$$

$$\begin{aligned} \tilde{\boldsymbol{R}}_{\frac{1}{\beta}}(I, I+P, K+1) = & \left[\begin{array}{cc} 0 & \boldsymbol{0} \\ \boldsymbol{0} & \tilde{\boldsymbol{R}}_{1}(I, I+P, K) \end{array}\right] \\ & + \boldsymbol{C}_{K+1,K+1}(\frac{1}{\beta}) \end{aligned} \tag{5.57}$$

Similarly, in this case by direct multiplication and the inductive assumption we have $\tilde{\boldsymbol{L}}_{\beta}(I, I+P, w, K+1)\tilde{\boldsymbol{R}}_{\frac{1}{\beta}}(I, I+P, K+1) = \boldsymbol{I}$.

Inductive case 3: The confinement pattern includes all the edges.

This is the same as special case 2 and has already been proven.

All three inductive cases cover all the possible confinement patterns for the 1-D random graph with $K+1$ vertices. Therefore, Prop. 4 is true at $N = K+1$ and the proof is completed.

5.A.6 Proof of Theorem 2

Proof. The proof is also based on induction. Prop. 4 has shown that the base case where $M = 1$ is true.

Assume Theorem 2 is true for $M = K$. Consequently, the following is true by this assumption.

$$\begin{aligned} \tilde{\boldsymbol{L}}_{K,\beta}(\boldsymbol{I}, \boldsymbol{I}+\boldsymbol{P}, \boldsymbol{w}) = & \left(\prod_{i=1}^{K} P_i\right) \\ & \left(\sum_{l_1=I_1}^{I_1+P_1-1} \sum_{l_2=I_2}^{I_2+P_2-1} \cdots \sum_{l_K=I_K}^{I_K+P_K-1} \boldsymbol{L}_{K,\beta}(\boldsymbol{l}, \boldsymbol{\alpha})^{-1}\right)^{-1} \end{aligned} \tag{5.58}$$

where $\boldsymbol{I} = \{I_i | i = 1, \ldots, K\}$ and $\boldsymbol{P} = \{P_i | i = 1, \ldots, K\}$ bound the location of the transition in each row. $\boldsymbol{w} = \{w_i | i = 1, \ldots, K\}$ are the graph edge weights in the confinement region in each row. For each transition geometry within the confinement, $\boldsymbol{l} = \{l_i | i = 1, \ldots, K\}$ and $\boldsymbol{\alpha} = \{\alpha_i | i = 1, \ldots, K\}$ define the exact location of the transition and corresponding graph edge weight in each row. We will show that this relationship is also true for $M = K+1$ to complete the induction.

First we note that for a particular transition geometry, the biased graph Laplacian with $K+1$ rows can be constructed using the biased graph Laplacian with K rows and a biased line graph Laplacian.

$$\boldsymbol{L}_{K+1,\beta}(\boldsymbol{l},\boldsymbol{\alpha}) = \left[\begin{array}{cc} \boldsymbol{L}_{\beta+1}(l_1,\alpha_1) & -\boldsymbol{S}_{KN,N}^T \\ -\boldsymbol{S}_{KN,N} & \boldsymbol{L}_{K,1}(\boldsymbol{l}',\boldsymbol{\alpha}') \end{array}\right] \tag{5.59}$$

where $\boldsymbol{l}' = \{l_i | i = 2,\dots,K+1\}$ and $\boldsymbol{\alpha}' = \{\alpha_i | i = 2,\dots,K+1\}$. For this construction, the new line is added to the top of the 2-D graph. The bias values are chosen such that the new matrix is a valid biased graph Laplacian.

Suppose in each row the transition is bounded by pixels I_i and I_i+P_i(for $i = 1,\dots,M$). Accordin to Eq. (5.3), the biased graph Laplacian for the signal with $K+1$ rows is given by the following expression.

$$\begin{aligned} \tilde{\boldsymbol{L}} = &\left(\prod_{i=1}^{K+1} P_i\right) \\ &\left(\sum_{l_1=I_1}^{I_1+P_1-1}\sum_{l_2=I_2}^{I_2+P_2-1}\cdots\sum_{l_{K+1}=I_{K+1}}^{I_{K+1}+P_{K+1}-1} \boldsymbol{L}_{K+1,\beta}(\boldsymbol{l},\boldsymbol{\alpha})^{-1}\right)^{-1} \end{aligned} \tag{5.60}$$

Our goal is to show that $\tilde{\boldsymbol{L}} = \tilde{\boldsymbol{L}}_{K+1,\beta}(\boldsymbol{I},\boldsymbol{I}+\boldsymbol{P},\boldsymbol{w})$.

If we recall Prop. 3, $\boldsymbol{L}_{K+1,\beta}(\boldsymbol{l},\boldsymbol{\alpha})^{-1}$ has the following strucutre in terms of the sub-block matrices. Furthermore, by utilizing Prop. 1 and 2, $\boldsymbol{L}_{K+1,\beta}(\boldsymbol{l},\boldsymbol{\alpha})^{-1}$ can be expressed in terms of biased Laplacian matrices.

$$\begin{aligned} \boldsymbol{L}_{K+1,\beta}(\boldsymbol{l},\boldsymbol{\alpha})^{-1} &= \left[\begin{array}{cc} \boldsymbol{R}_{\frac{1}{\beta}}(l_1,\alpha_1) & \boldsymbol{C}_{KN,N}(\frac{1}{\beta})^T \\ \boldsymbol{C}_{KN,N}(\frac{1}{\beta}) & \boldsymbol{R}_{K,\frac{1}{\beta}+1}(\boldsymbol{l}',\boldsymbol{\alpha}') \end{array}\right] \\ &= \left[\begin{array}{cc} \boldsymbol{L}_{\beta}(l_1,\alpha_1)^{-1} & \boldsymbol{C}_{KN,N}(\frac{1}{\beta})^T \\ \boldsymbol{C}_{KN,N}(\frac{1}{\beta}) & \boldsymbol{L}_{K,\frac{\beta}{\beta+1}}(\boldsymbol{l}',\boldsymbol{\alpha}')^{-1} \end{array}\right] \end{aligned} \tag{5.61}$$

If we put Eq. (5.61) into Eq. (5.60), we have the following relationship.

$$\tilde{\boldsymbol{L}} = \left(\prod_{i=1}^{K+1} P_i\right) \left[\begin{array}{cc} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{C} & \boldsymbol{D} \end{array}\right]^{-1} \tag{5.62}$$

Sub-matrix A is

$$\boldsymbol{A} = \left(\prod_{i=2}^{K+1} P_i\right) \sum_{l_1=I_1}^{I_1+P_1-1} \boldsymbol{L}_{\beta}(l_1,\alpha_1)^{-1} \tag{5.63}$$

By Prop. 4, $\boldsymbol{A}$ can also be written as

$$\boldsymbol{A} = \left(\prod_{i=1}^{K+1} P_i\right) \tilde{\boldsymbol{L}}_{\beta}(I_1, I_1 + P_1, w_1)^{-1} \tag{5.64}$$

where $w_1 = \frac{P_1\alpha_1}{(P_1-1)\alpha_1+1}$.

Sub-matrix $\boldsymbol{D}$ is

$$\boldsymbol{D} = P_1 \sum_{l_2=I_2}^{I_2+P_2-1} \sum_{l_3=I_3}^{I_3+P_3-1} \cdots \sum_{l_{K+1}=I_{K+1}}^{I_{K+1}+P_{K+1}-1} \boldsymbol{L}_{K,\frac{\beta}{\beta+1}}(\boldsymbol{l}', \boldsymbol{\alpha}')^{-1} \tag{5.65}$$

By the assumption made at the beginning of the inductive step, the following is true.

$$\boldsymbol{D} = \left(\prod_{i=1}^{K+1} P_i\right) \tilde{\boldsymbol{L}}_{K,\frac{\beta}{\beta+1}}(\boldsymbol{I}', \boldsymbol{I}' + \boldsymbol{P}', \boldsymbol{w}')^{-1} \tag{5.66}$$

where $\boldsymbol{I}' = \{I_i | i = 2, \dots, K+1\}$, $\boldsymbol{P}' = \{P_i | i = 2, \dots, K+1\}$ and $\boldsymbol{w}' = \left\{w_i = \frac{P_i\alpha_i}{(P_i-1)\alpha_i+1} | i = 2, \dots, K+1\right\}$.

Sub-matrices $\boldsymbol{B}$ and $\boldsymbol{C}$ are

$$\boldsymbol{B}^T = \boldsymbol{C} = \left(\prod_{i=1}^{K+1} P_i\right) \boldsymbol{C}_{KN,N}(\frac{1}{\beta}) \tag{5.67}$$

Consequently,$\tilde{L}$ is given as follows in terms of the sub-matrices X, Y, Z and U.

$$\begin{aligned} \tilde{\boldsymbol{L}} &= \begin{bmatrix} \boldsymbol{X} & \boldsymbol{Y} \\ \boldsymbol{Z} & \boldsymbol{U} \end{bmatrix} \\ &= \begin{bmatrix} \tilde{\boldsymbol{L}}_{\beta}(I_1, I_1 + P_1, w_1)^{-1} & \boldsymbol{C}_{KN,N}(\frac{1}{\beta})^T \\ \boldsymbol{C}_{KN,N}(\frac{1}{\beta}) & \tilde{\boldsymbol{L}}_{K,\frac{\beta}{\beta+1}}(\boldsymbol{I}', \boldsymbol{I}' + \boldsymbol{P}', \boldsymbol{w}')^{-1} \end{bmatrix}^{-1} \end{aligned} \tag{5.68}$$

Next, we use block matrix inversion to find $\tilde{\boldsymbol{L}}$.

Sub-matrix $\boldsymbol{X}$ is given as follows.

$$\begin{aligned} \boldsymbol{X} = \Big(&\tilde{\boldsymbol{L}}_{\beta}(I_1, I_1 + P_1, w_1)^{-1} - \\ &\boldsymbol{C}_{KN,N}(\frac{1}{\beta})^T \tilde{\boldsymbol{L}}_{K,\frac{\beta}{\beta+1}}(\boldsymbol{I}', \boldsymbol{I}' + \boldsymbol{P}', \boldsymbol{w}') \boldsymbol{C}_{KN,N}(\frac{1}{\beta})\Big)^{-1} \end{aligned} \tag{5.69}$$

Since $\boldsymbol{C}_{KN,N}(\frac{1}{\beta})^T$ is a constant matrix and only the first column sum of $\tilde{\boldsymbol{L}}_{K,\frac{\beta}{\beta+1}}(\boldsymbol{I}', \boldsymbol{I}' + \boldsymbol{P}', \boldsymbol{w}')$ is nonzero and equal to $\frac{\beta}{\beta+1}$ we have the following.

$$\boldsymbol{C}_{KN,N}(\frac{1}{\beta})^T \tilde{\boldsymbol{L}}_{K,\frac{\beta}{\beta+1}}(\boldsymbol{I}', \boldsymbol{I}' + \boldsymbol{P}', \boldsymbol{w}') = \frac{1}{\beta+1} \boldsymbol{O}_{N,KN} \tag{5.70}$$

$$\boldsymbol{U} = \tilde{\boldsymbol{L}}_{K,\frac{\beta}{\beta+1}}(\boldsymbol{I}',\boldsymbol{I}'+\boldsymbol{P}',\boldsymbol{w}') - \frac{1}{1+g}\tilde{\boldsymbol{L}}_{K,\frac{\beta}{\beta+1}}(\boldsymbol{I}',\boldsymbol{I}'+\boldsymbol{P}',\boldsymbol{w}')(-\boldsymbol{C}_{KN,KN}(\frac{1}{\beta}))\tilde{\boldsymbol{L}}_{K,\frac{\beta}{\beta+1}}(\boldsymbol{I}',\boldsymbol{I}'+\boldsymbol{P}',\boldsymbol{w}') \tag{5.75}$$

Therefore, $\boldsymbol{X}$ is reduced to

$$\boldsymbol{X} = \Big(\tilde{\boldsymbol{L}}_{\beta}(I_1, I_1 + P_1, w_1)^{-1} - \boldsymbol{C}_{N,N}(\frac{1}{(\beta+1)\beta})\Big)^{-1} \tag{5.71}$$

In addition, Prop. 4 shows that $\tilde{\boldsymbol{L}}_{\beta}(I_1, I_1 + P_1, w_1)^{-1}$ contains a constant value $\frac{1}{\beta}$ at every entry. By changing the constant value, $\boldsymbol{X}$ is further reduced to a Laplacian matrix with a different bias

$$\boldsymbol{X} = \tilde{\boldsymbol{L}}_{\beta+1}(I_1, I_1 + P_1, w_1) \tag{5.72}$$

Sub-matrix $\boldsymbol{U}$ by the block inversion algorithm is given as follows.

$$\begin{aligned}\boldsymbol{U} = \Big(&\tilde{\boldsymbol{L}}_{K,\frac{\beta}{\beta+1}}(\boldsymbol{I}',\boldsymbol{I}'+\boldsymbol{P}',\boldsymbol{w}')^{-1} - \\ &\boldsymbol{C}_{KN,N}(\frac{1}{\beta})\tilde{\boldsymbol{L}}_{\beta}(I_1, I_1 + P_1, w_1)\boldsymbol{C}_{KN,N}(\frac{1}{\beta})^T\Big)^{-1}\end{aligned} \tag{5.73}$$

Similar to the steps in finding $\boldsymbol{X}$, we can reduce $\boldsymbol{U}$ to the following expression.

$$\boldsymbol{U} = \Big(\tilde{\boldsymbol{L}}_{K,\frac{\beta}{\beta+1}}(\boldsymbol{I}',\boldsymbol{I}'+\boldsymbol{P}',\boldsymbol{w}')^{-1} - \boldsymbol{C}_{KN,KN}(\frac{1}{\beta})\Big)^{-1} \tag{5.74}$$

The stucture of $\tilde{\boldsymbol{L}}_{K,\frac{\beta}{\beta+1}}(\boldsymbol{I}',\boldsymbol{I}'+\boldsymbol{P}',\boldsymbol{w}')^{-1}$ has not been established in our previous discussions. To circumvent this issue, we use Eq. (5.22) in Appendix 5.A.1. The constant matrix with value $\frac{1}{\beta}$ is a rank-one matrix. Using the Woodbury matrix identity from Eq. (5.22) in Appendix 5.A.1, we can rewrite $\boldsymbol{U}$ as in Eq. (5.75).

We reuse the same trick on multiplying a biased graph Laplacian matrix with constant matrices. Therefore, $g = -\frac{1}{\beta+1}$ and $\frac{1}{g+1} = \frac{1+\beta}{\beta}$. Furthermore, we rewrite sub-matrix $\boldsymbol{U}$ as follows.

$$\begin{aligned}\boldsymbol{U} &= \tilde{\boldsymbol{L}}_{K,\frac{\beta}{\beta+1}}(\boldsymbol{I}',\boldsymbol{I}'+\boldsymbol{P}',\boldsymbol{w}') - \frac{1+\beta}{\beta}(-\frac{\beta}{(\beta+1)^2})\boldsymbol{S}_{KN,KN} \\ &= \tilde{\boldsymbol{L}}_{K,1}(\boldsymbol{I}',\boldsymbol{I}'+\boldsymbol{P}',\boldsymbol{w}')\end{aligned} \tag{5.76}$$

As a result, by employing Eq. (5.22), $\boldsymbol{U}$ can be found by changing the single bias value in $\tilde{\boldsymbol{L}}_{K,\frac{\beta}{\beta+1}}(\boldsymbol{I}',\boldsymbol{I}'+\boldsymbol{P}',\boldsymbol{w}')$.

Sub-matrix $\boldsymbol{Y}$ is expressed as follows.

$$\begin{aligned}\boldsymbol{Y} &= -\tilde{\boldsymbol{L}}_{\beta}(I_1, I_1 + P_1, w_1)\boldsymbol{C}(\frac{1}{\beta})^T\boldsymbol{U} \\ &= -\boldsymbol{O}^T_{KN,N}\boldsymbol{U} = -\boldsymbol{S}^T_{KN,N}\end{aligned} \tag{5.77}$$

Sub-matrix $\boldsymbol{Z}$ is found as follows.

$$\begin{aligned}\boldsymbol{Z} &= -\tilde{\boldsymbol{L}}_{K,\frac{\beta}{\beta+1}}(\boldsymbol{I}', \boldsymbol{I}' + \boldsymbol{P}', \boldsymbol{w}')\boldsymbol{C}(\frac{1}{\beta})\boldsymbol{X} \\ &= -\frac{1}{\beta+1}\boldsymbol{O}^T_{N,KN}\boldsymbol{X} = -\boldsymbol{S}_{KN,N}\end{aligned} \tag{5.78}$$

In summary, $\tilde{\boldsymbol{L}}$ can be expressed as follows.

$$\begin{aligned}\tilde{\boldsymbol{L}} &= \begin{bmatrix} \tilde{\boldsymbol{L}}_{\beta+1}(I_1, I_1 + P_1, w_1) & -\boldsymbol{S}^T_{KN,N} \\ -\boldsymbol{S}_{KN,N} & \tilde{\boldsymbol{L}}_{K,1}(\boldsymbol{I}', \boldsymbol{I}' + \boldsymbol{P}', \boldsymbol{w}') \end{bmatrix} \\ &= \tilde{\boldsymbol{L}}_{K+1,\beta}(\boldsymbol{I}, \boldsymbol{I} + \boldsymbol{P}, \boldsymbol{w})\end{aligned} \tag{5.79}$$

Eq.(5.79) is a valid biased graph Laplacian with $K+1$ rows and satisfies the structure of $\tilde{\boldsymbol{L}}_{K+1,\beta}(\boldsymbol{I}, \boldsymbol{I} + \boldsymbol{P}, \boldsymbol{w})$. Hence we have shown that if Theorem 2 is true at $M = K$ then it also holds at $M = K+1$. The proof is completed. $\square$

5.B Rate Distortion Comparisons

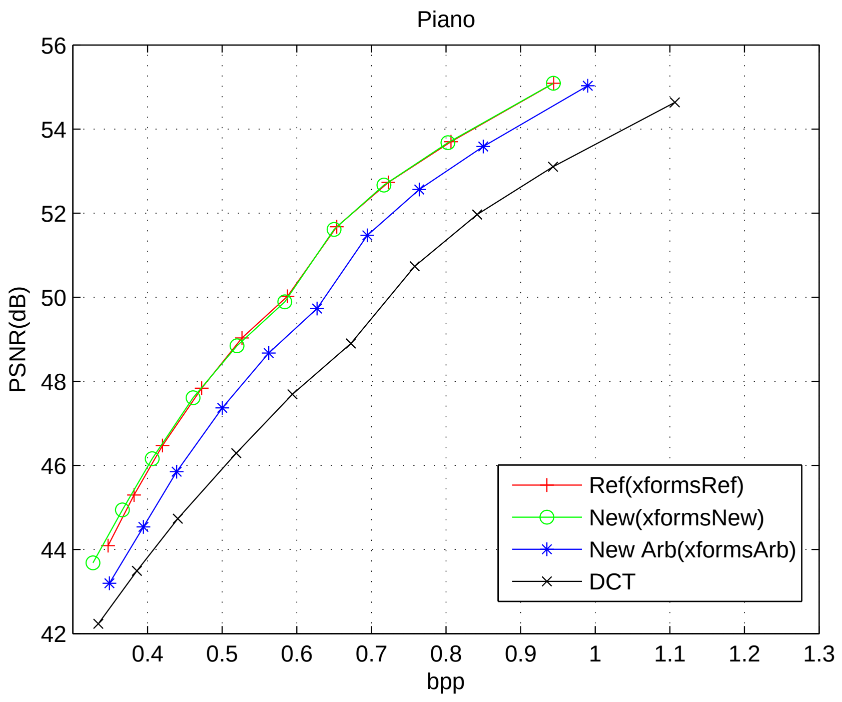


Figure 5.5: Rate distortion comparison for Piano.

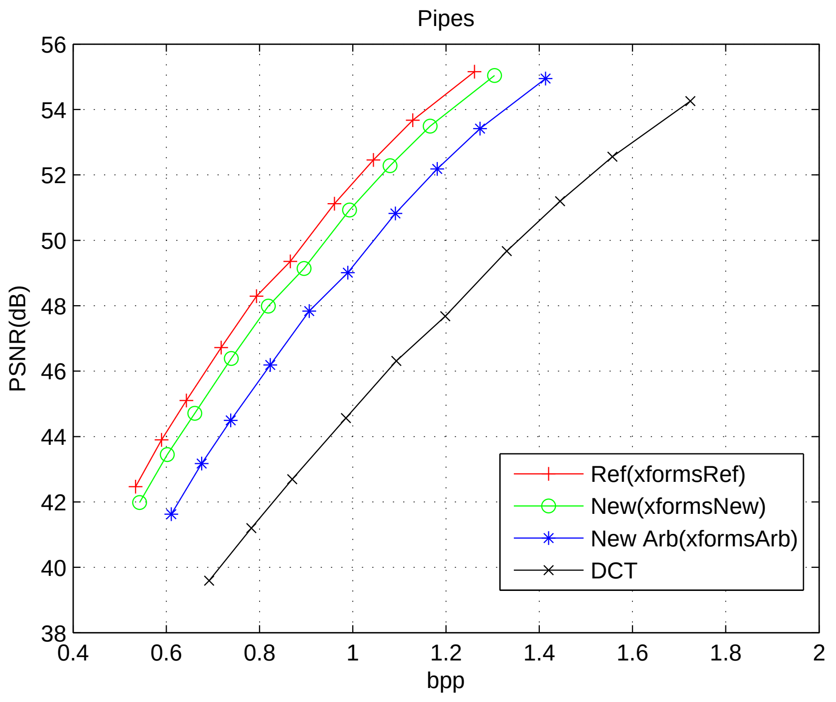


Figure 5.6: Rate distortion comparison for Pipes.

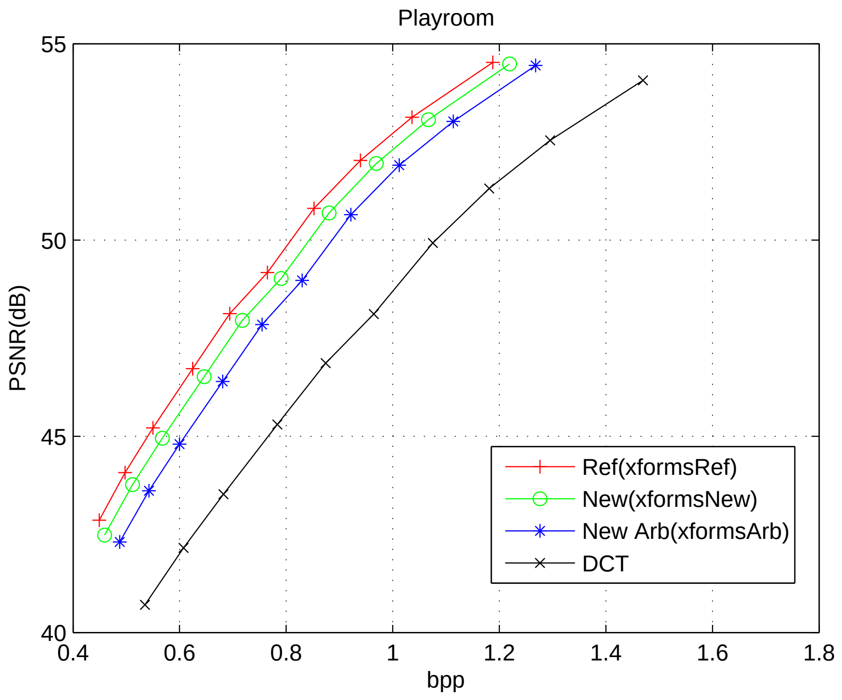


Figure 5.7: Rate distortion comparison for Playroom.

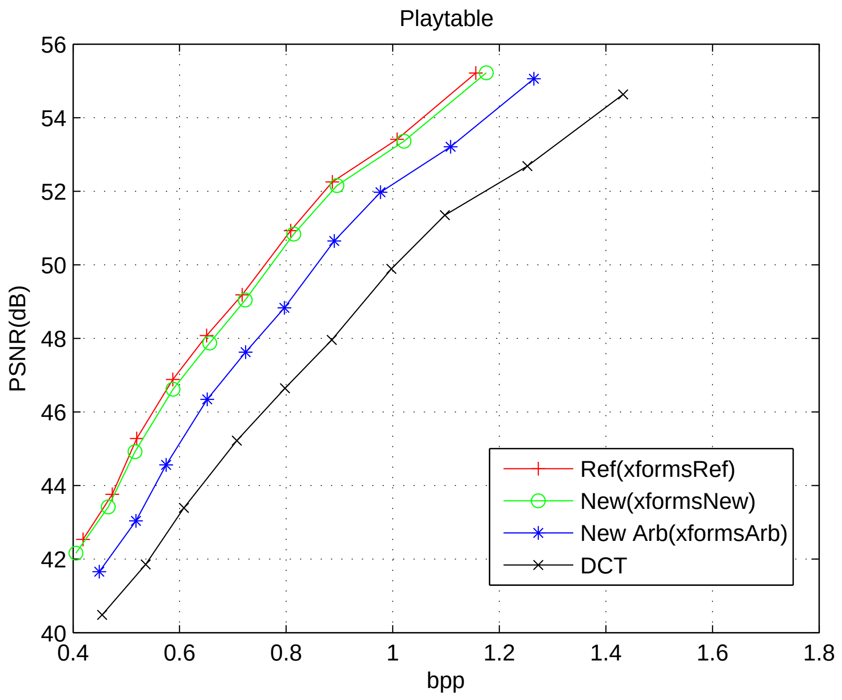


Figure 5.8: Rate distortion comparison for Playtable.

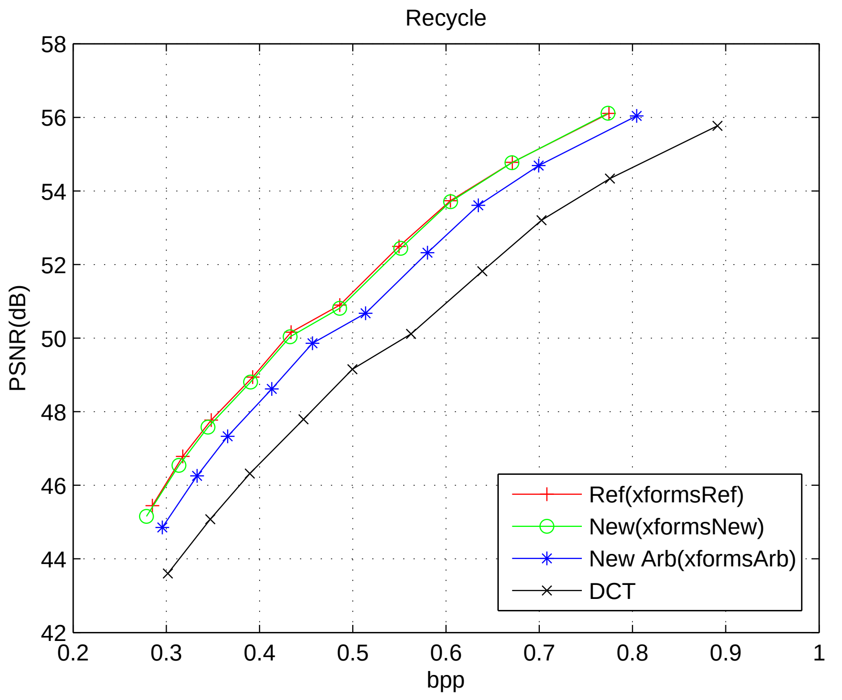


Figure 5.9: Rate distortion comparison for Recycle.

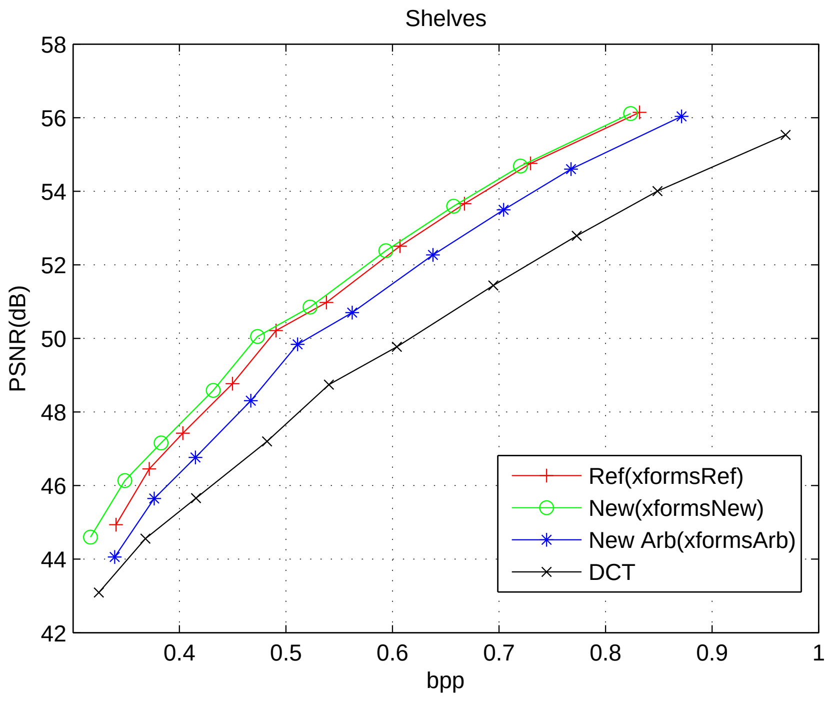


Figure 5.10: Rate distortion comparison for Shelves.

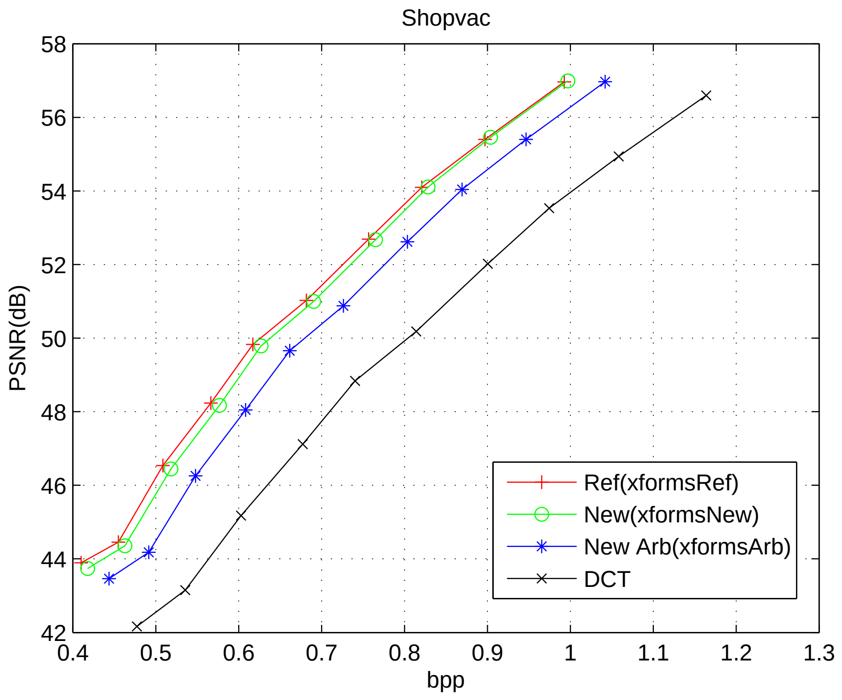


Figure 5.11: Rate distortion comparison for Shopvac.

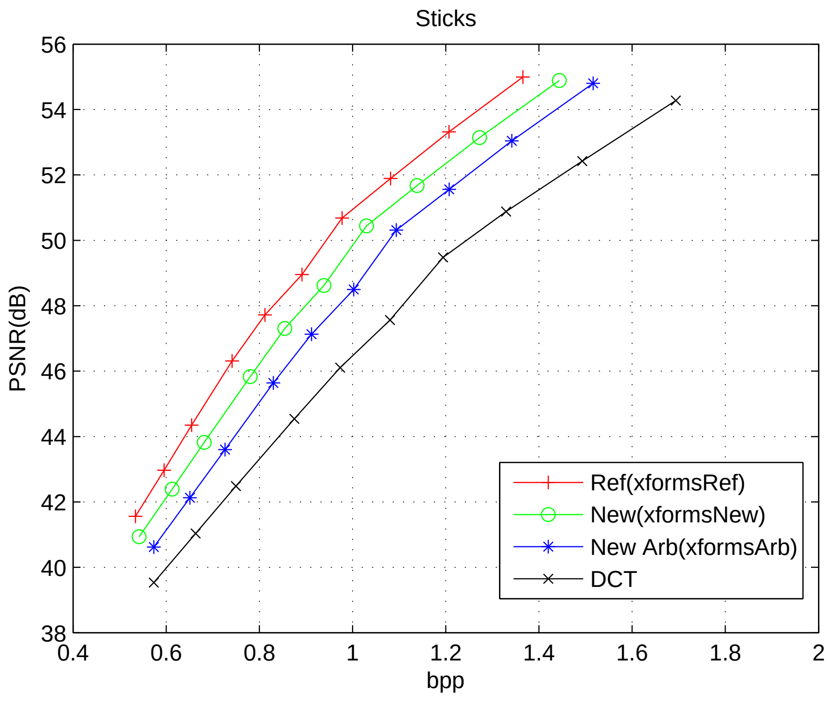


Figure 5.12: Rate distortion comparison for Sticks.

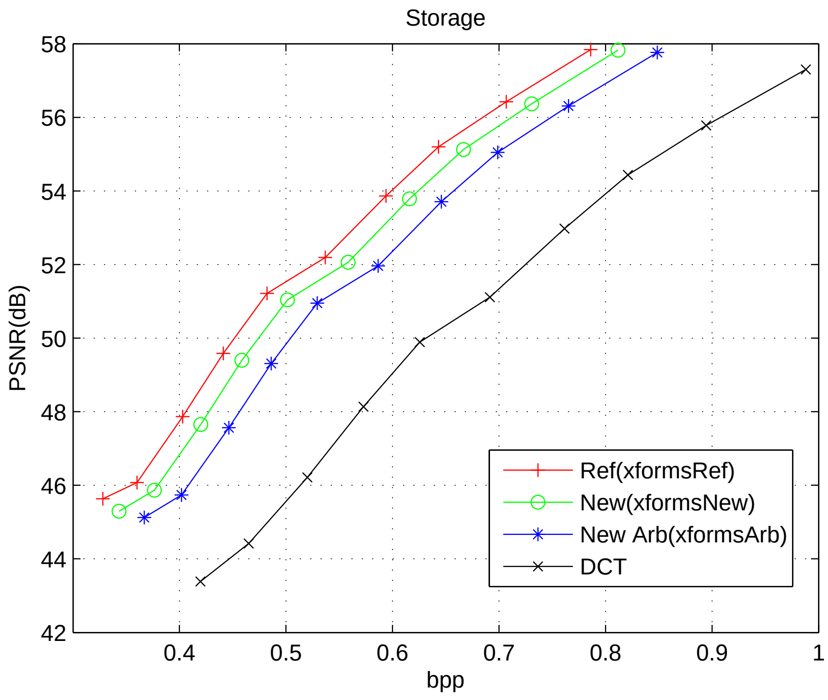


Figure 5.13: Rate distortion comparison for Storage.

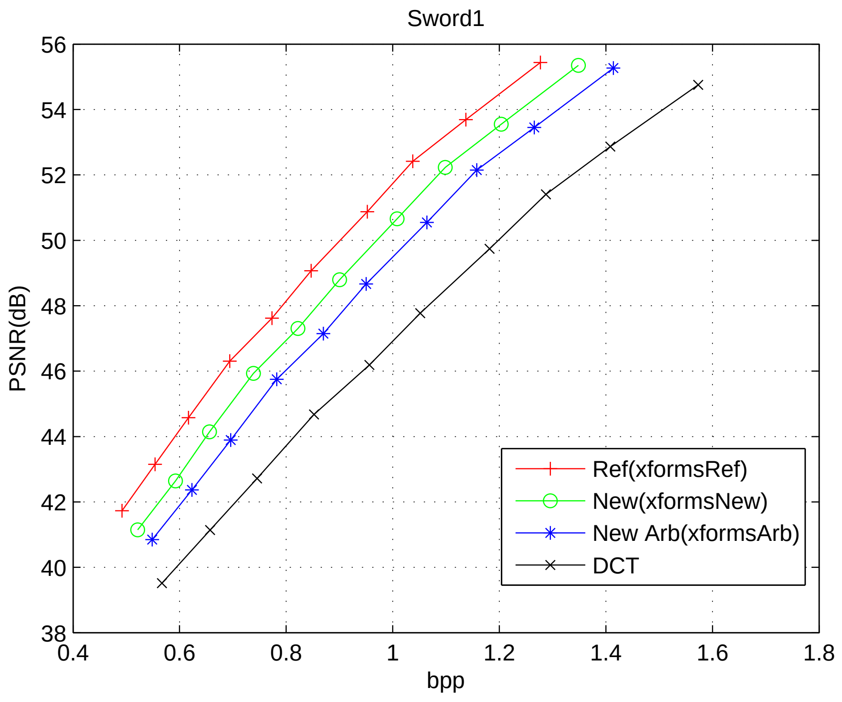


Figure 5.14: Rate distortion comparison for Sword1.

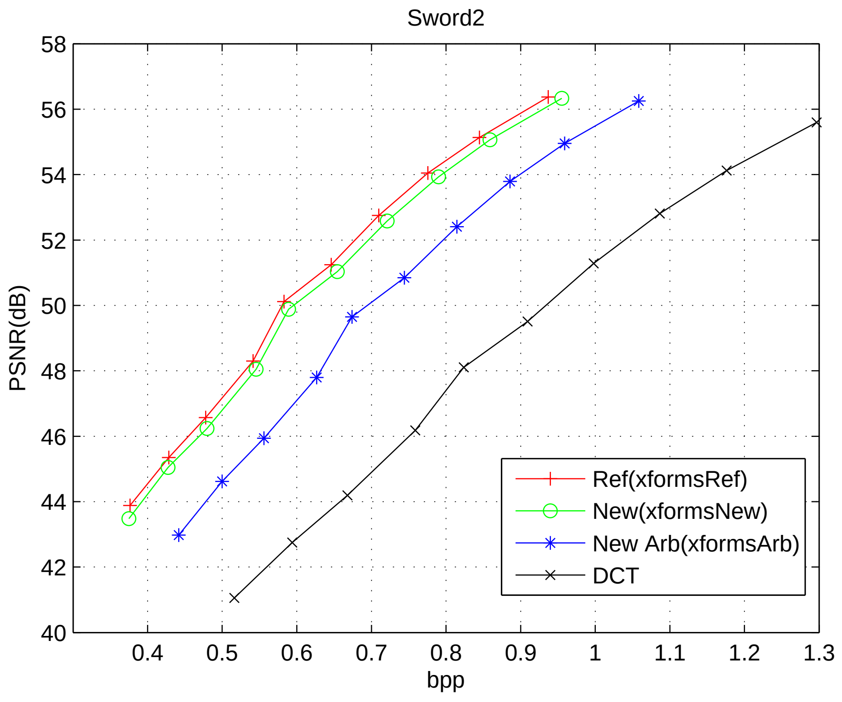


Figure 5.15: Rate distortion comparison for Sword2.

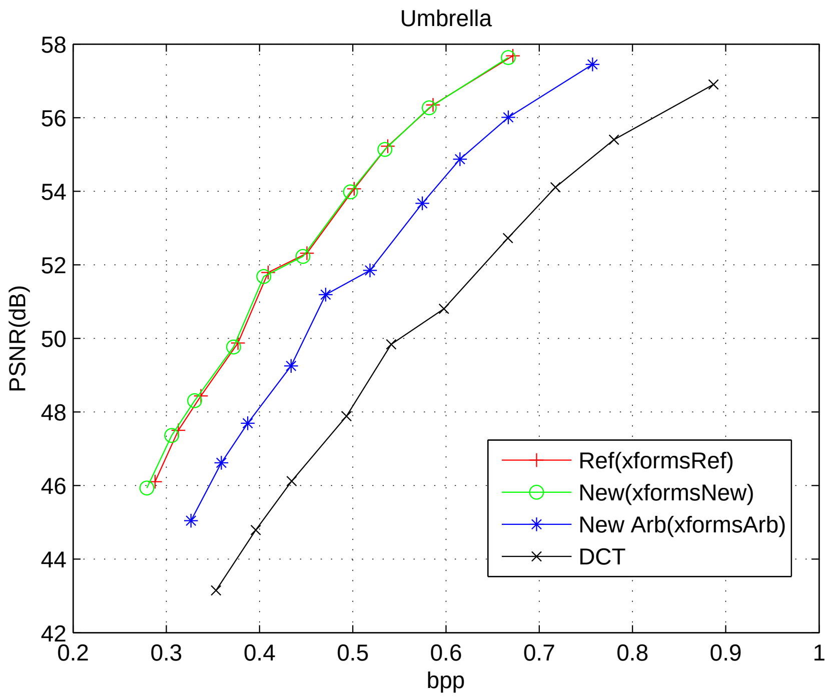


Figure 5.16: Rate distortion comparison for Umbrella.

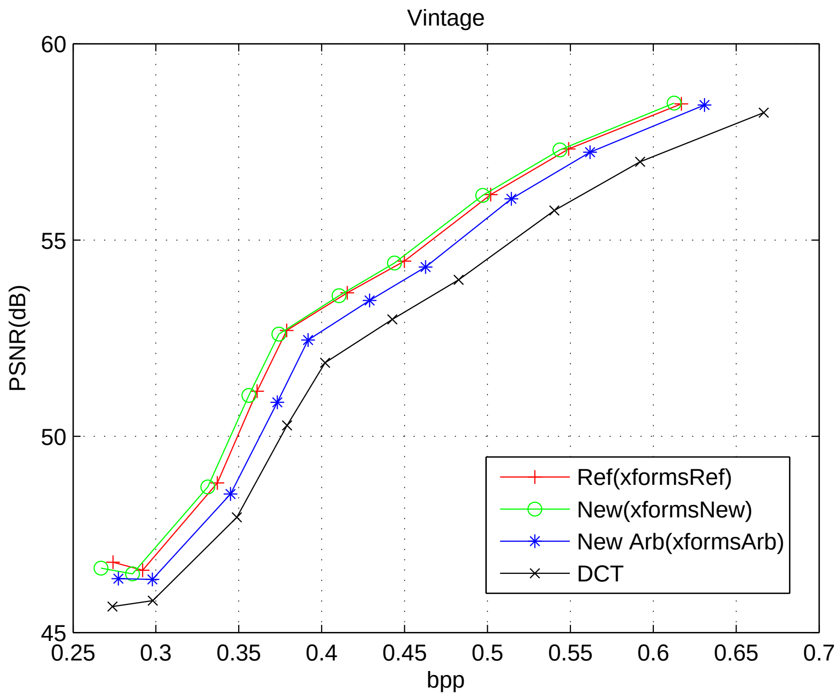


Figure 5.17: Rate distortion comparison for Vintage.

Chapter 6

Conclusions

6.1 Conclusions

In this thesis, we study the problems of joint source channel coding for single view video transmission, transmission distortion estimation for multiview video coding and depth video coding for multiview video applications.

In Chapter 3, we present the design of a real-time embedded H.264/AVC-based video conferencing system, which utilizes a fast MB-level end-to-end distortion estimation algorithm, a family of very short FEC codes, and a fast adaptive FEC rate allocation scheme. The experimental results demonstrate that the system design outperforms conventional non-adaptive FEC rate allocation scheme.

In Chapter 4, we intend to design a similar joint source channel coding framework for FTV systems. However, the current transmission distortion estimation algorithms for the synthesized virtual view suffer from accuracy issues. As an initial step toward this goal, we develop a theoretical framework which accurately models the virtual view distortion when the reference texture and depth videos are affected by random errors like packet loss. A graphical model-based method is first formulated which is capable of modeling the pixel level interactions when reference depth images are assumed to be affected by random errors with known distributions. Then a recursive optimal distribution estimation (RODE) method is introduced such that pixel level distribution due to transmission error can be precisely calculated. Finally, the RODE method is integrated into the graphical model-based method and synthesized view distortion due to transmission error can be estimated. Experimental results demonstrate that the accuracy of the proposed scheme outperforms the existing state-of-the-art approaches.

An efficient compression algorithm for depth images is crucial to multiview video based applications like FTV. In Chapter 5, we aim to derive a single optimal transform for a class of depth image blocks that contain distinct but similar edge geometries. In our derivation, we model the collection of depth image blocks as a 2-D piecewise smooth signal class in which the location of the discontinuity is random but bounded within a region. A theory is proposed to derive the optimal graph transform for this type of signals. Depth image coding results reveal that our new transform designs outperform DCT and have comparable performance to the state-of-the-art method in the literature with much lower complexity.

6.2 Future Works

It is mentioned in Chapter 4 that the proposed distortion estimation algorithm for the synthesized virtual view does not consider filtering operations and complex motion compensation schemes. Therefore, one possible future work can focus on accounting for the effect of median filtering, hole dilation, deblocking filter, sub-pixel motion estimation, and B frames. It is evident from our derivation that the complexity of the graphical model makes the estimation algorithm unsuitable for real-time applications. In comparison to ROPE, PMF tracking in RODE requires much more computational resources. One possible approach to reduce the complexity of RODE is to use maximum entropy principle [74] to estimate the PMF rather than tracking the exact distribution for each pixel. This makes our estimation framework more attractive to future 3D video coding and transmission optimization applications.

In Chapter 5, our discussion implies that signaling the exact location of discontinuity is expensive whereas signaling an offline transform table index requires far less bits. However, in terms of coding performance, a block transformed by the graph which exactly reflects its discontinuity is certainly sparser than if the same block is transformed by a graph that is designed for a class of signals. Based on this observation, one possible future work can focus on optimally balancing the trade-off between the two coding methods to achieve better RD performance. In addition, the image model used in Chapter 5 is based on a 2-D AR1 process. To improve the modeling accuracy, a higher-order AR process can be adopted. Consequently, finding the graph structure corresponding to a higher-order AR image model can be another interesting future work. Furthermore, the applications of graph-based transforms are not limited to image coding. Many big and complex data which are conveniently represented using graphs can be analyzed using graph related techniques. Therefore, it could be interesting as a future work to investigate how graph-based transform can be utilized in big data analysis. As a concrete example, analyzing historical flight traffic data is a crucial step in optimizing airplane routes and airport utilization. However, the vast amount of traffic data presents a challenge for data storage. Since the graph-based transform proposed in this thesis has extremely low complexity, it is possible to employ our transform design for traffic data compression while meeting real-time requirements.

Bibliography

[1] IETF RFC 6330: RaptorQ forward error correction scheme for object delivery. *IETF Proposed Standard*, Aug. 2011.

[2] Middlebury 2014 depth image datasets. [Online]. *Available: http://vision.middlebury.edu/stereo/data/scenes2014/*, Accessed: Dec. 2015.

[3] MPEG-FTV test sequence download page. [Online]. *http://www.tanimoto.nuee.nagoya-u.ac.jp/ fukushima/mpegftv/*, Accessed: Jan. 2014.

[4] Nokia. [Online]. *ftp://mpeg3dv.research.nokia.com*, Accessed: Jan. 2014.

[5] Poznan university of technology. [Online]. *ftp://multimedia.edu.pl/3DV/*, Accessed: Jan. 2014.

[6] View synthesis reference software (VSRS) version 3.5. [Online]. *http://wg11.sc29.org/svn/repos/MPEG-4/test/tags/3D/view-synthesis/VSRS-3-5/*, Accessed: Jan. 2014.

[7] B. Aksasse and L. Radouane. Two-dimensional autoregressive (2-D AR) model order estimation. *IEEE Trans. Signal Process.*, 47(7):2072 – 2077, Jul. 1999.

[8] S. Argyropoulos, A. S. Tan, N. Thomos, E. Arikan, and M. G. Strintzis. Robust transmission of multi-view video streams using flexible macro block ordering and systematic LT codes. In *Proc. 3DTV Conf. (Kos, Greece)*, pages 1–4, May. 2007, May. 2007.

[9] M. Bertalmio, A. L. Bertozzi, and G.Sapiro. Navier-stokes, fluid dynamics, and image and video inpainting. *Proc. ICCV 2001*, pages 1335–1362, 2001.

[10] J. Byers, M. Luby, and M. Mitzenmacher. A digital fountain approach to asynchronous reliable multicast. *IEEE J. Sel. Areas Commun.*, 20(8):1528–1540, Oct. 2002.

[11] Y. H. Chao, A. Ortega, W. Hu, and G. Cheung. Edge-adaptive depth map coding with lifting transform on graphs. *Picture Coding Symposium (PCS)*, pages 60–64, Los Angeles, CA. May 2015.

[12] Z. Chen, P. V. Pahalawatta, A. M. Tourapis, and D. Wu. Improved estimation of transmission distortion for error-resilient video coding. *IEEE Trans. Circuits Syst. Video Technol.*, 22(4):636–647, Apr. 2012.

[13] G. Cheung, J. Ishida, A. Kubota, and A. Ortega. Transform domain sparsification of depth maps using iterative quadratic programming. *Proc. IEEE Conf. on Image Process.*, pages 129–132, Brussels, Belgium. September 2011.

[14] G. Cheung, W.S. Kim, A. Ortega, J. Ishida, and A. Kubota. Depth map coding using graph based transform and transform domain sparsification. *Proc. IEEE Multimedia Signal Processing*, pages 1 – 6, Oct. 2011.

[15] G. Cheung, V. Velisavljevic, and A. Ortega. On dependent bit allocation for multi-view image coding with depth-image-based rendering. *IEEE Trans. Image Process.*, 20(11):3179–3194, November 2011.

[16] S.Y. Chung, G. D. Forney, T. J. Richardson, and R. Urbanke. On the design of low-density parity-check codes within 0.0045db of the shannon limit. *IEEE Communications Letters*, 5(2):58–60, Feb. 2001.

[17] T. Y. Chung, W. D. Jang, and C. S. Kim. Efficient depth video coding based on view synthesis distortion estimation. *IEEE Visual Communications and Image Processing*, pages 1–4, November 2012.

[18] J. De Cock, S. Notebaert, P. Lambert, and R. Van de Walle. Architectures for fast transcoding of H.264/AVC to quality-scalable SVC streams. *IEEE Trans. Multimedia*, 11(7):1209–1224, Nov. 2009.

[19] H. P. Deng, L. Yu, B. Feng, and Q. Liu. Structural similarity-based synthesized view distortion estimation for depth map coding. *IEEE Transactions on Consumer Electronics*, 58(4):1338–1344, November 2012.

[20] C. Di, I.E. Proietti, D. Telatar, T.J Richardson, and R.L Urbanke. Finite-length analysis of low-density parity-check codes on the binary erasure channel. *IEEE Transactions on Information Theory*, 48(6):1570–1579, Jun. 2002.

[21] A. W. Eckford, J. P. K. Chu, and R. S. Adve. Low-complexity cooperative coding for sensor networks using rateless and LDGM codes. In *Proc. IEEE Int. Conf. on Commun.*, volume 4, pages 1537–1542, Jun. 2006, Jun. 2006.

[22] L. Fang, N. M. Cheung, D. Tian, A. Vetro, H. Sun, and O. C. Au. An analytical model for synthesis distortion estimation in 3D video. *IEEE Trans. Image Process.*, 23(1):185–199, 2014.

[23] R.G. Gallager. *Low Density Parity-Check Codes.* MIT Press, Cambridge, MA, 1963.

[24] J. Han, A. Saxena, V. Melkote, and K. Rose. Jointly optimized spatial prediction and block transform for video and image coding. *IEEE Trans. Image Process.*, 21 (4):1874–1884, Apr. 2012.

[25] Z. He, J. Cai, and C. W. Chen. Joint source channel rate-distortion analysis for adaptive mode selection and rate control in wireless video coding. *IEEE Trans. Circuits Syst. Video Technol.*, 12:511–523, Jun. 2002.

[26] Z. He and H. Xiong. Transmission distortion analysis for real-time video encoding and streaming over wireless networks. *IEEE Trans. Circuits Syst. Video Technol.*, 16(9):1051–1062, Sept. 2006.

[27] Y. R. Horng, Y. C. Tseng, and T. S. Chang. VLSI architecture for real-time HD1080p view synthesis engine. *IEEE Trans. Circuits Syst. Video Technol.*, 21(9):1329–1340, September 2011.

[28] H. Hotelling. Some new methods in matrix calculation. *The Annals of Mathematical Statistics*, 14:1–34, 1943.

[29] S. Hu, S. Kwong, Y. Zhang, and C. C. J. Kuo. Rate-distortion optimized rate control for depth map-based 3-D video coding. *IEEE Trans. Image Process.*, 22(2):585 –594, 2012.

[30] W. Hu, G. Cheung, and A. Ortega. Intra-prediction and generalized graph Fourier transform for image coding. *IEEE Signal Process. Lett.*, 22(11):1913 – 1917, Nov. 2015.

[31] W. Hu, G. Cheung, A. Ortega, and O. C. Au. Multiresolution graph Fourier transform for compression of piecewise smooth images. *IEEE Trans. Image Process.*, 24(1):419–433, Jan. 2015.

[32] W. S. Kim, A. Ortega, P. Lai, D. Tian, and C. Gomila. Depth map distortion analysis for view rendering and depth coding. *in Proc. IEEE Int. Conf. Image Process.*, 2009.

[33] W. S. Kim, A. Ortega, P. Lai, D. Tian, and C. Gomila. Depth map coding with distortion estimation of rendered view. *in Proc. SPIE Vis. Inf. Process. Commun.*, 2010.

[34] W.S. Kim, S.K. Narang, and A. Ortega. Graph based transforms for depth video coding. *Proc. IEEE Int. Conf. Acoust., Speech, and Signal Process.*, pages 813–816, March 2012.

[35] F. Li and G. Liu. Compressed-domain-based transmission distortion modeling for precoded H.264/AVC video. *IEEE Trans. Circuits Syst. Video Technol.*, 19(12):1908–1914, Dec. 2009.

[36] J. Liang and T. D. Tran. Fast multiplierless approximation of the DCT with the lifting scheme. *IEEE Trans. Signal Process.*, 49(12):3032 –3044, Dec. 2001.

[37] S. Lin and D. J. Costello. *Error Control Coding (2nd Edition)*. Prentice Hall, 2004.

[38] Y. Liu, Q. Huang, S. Ma, D. Zhao, and W. Gao. Joint video/depth rate allocation for 3D video coding based on view synthesis distortion model. *Signal Processing: Image Communication*, 24(8):666–681, September 2009.

[39] M. Luby. LT codes. *Proc. 43rd Annual IEEE Symp. Foundations of Computer Science*, pages 271–282, Vancouver, Canada, Nov. 2002.

[40] M. Luby, M. Watson, T. Gasiba, T. Stockhammer, and W. Xu. Raptor codes for reliable download delivery in wireless broadcast systems. In *Proc. Consumer Comm. Net. Conf*, pages 192–197, Feb. 2006, Feb. 2006.

[41] B. Macchiavello, C. Dorea, E. M. Hung, G. Cheung, and W. T. Tan. Loss-resilient coding of texture and depth for free-viewpoint video conferencing. *IEEE Trans. Multimedia*, 16(3):711 –725, Apr. 2014.

[42] B. Macchiavello, C. Dorea, E. M. Hung, G. Cheung, and W. T. Tan. Reference frame selection for loss-resilient depth map coding in multiview video conferencing. *Proc. of SPIE, Visual Information Processing and Communication III*, Burlingame, CA. January 2012.

[43] B. Macchiavello, C. Dorea, E. M. Hung, G. Cheung, and W. T. Tan. Reference frame selection for loss-resilient texture & depth map coding in multiview video conferencing. *Proc. IEEE Conf. on Image Process.*, pages 1653–1656, Orlando, FL. September 2012.

[44] D.J.C MacKay. Fountain codes. *IEE Proceedings Communications*, 152(6):1062–1068, Dec. 2005.

[45] P. Merkle, Y. Morvan, A. Smolic, D. Farin, K. MÃijller, P.H.N. de With, and T. Wiegand. The effect of depth compression on multiview rendering quality. *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video*, pages 245 –248, Istanbul, Turkey. May 2008.

[46] K. S. Miller. On the inverse of the sum of matrices. *Mathematics Magazine*, 54(2):67–72, 1981.

[47] T. Mladenov, K. Kim, and S. Nooshabadi. Forward error correction with RaptorQ code on embedded system. In *Proc. IEEE Inter. Midwest Symp. Circ. Sys.*, pages 1–4, Aug. 2011, Aug. 2011.

[48] S. Mys, Y. Dhondt, D. Walle, D. Schrijver, and R. Walle. A performance evaluation of the data partitioning tool in H.264/AVC. In *Proc. the International Society for Optics and Photonics*, volume 6391, Boston, MA, USA, Oct. 2006, 2006.

[49] S. Nyamweno, R. Satyan, and F. Labeau. Exponential decay of transmission distortion in H.264. In *Multimedia Signal Processing Workshop*, pages 268–271, Oct. 2007, Oct. 2007.

[50] B. T. Oh and K. J. Oh. View synthesis distortion estimation for AVC- and HEVC-compatible 3D video coding. *IEEE Trans. Circuits Syst. Video Technol.*, to appear.

[51] E. Pavez, H. E. Egilmez, Y. Wang, and A. Ortega. GTT: Graph template transforms with applications to image coding. *Picture Coding Symposium (PCS)*, pages 199–203, Los Angeles, CA. May 2015.

[52] T. Richardson, A. Shokrollahi, and R. Urbanke. Design of capacity-approaching irregular low-density parity-check codes. *IEEE Trans. Inf. Theory*, 47(2):619–637, Feb. 2001.

[53] I. Rotondo, G. Cheung, A. Ortega, and H. Egilmez. Designing sparse graphs via structure tensor for block transform coding of images. *Asia-pacific signal and information processing association annual summit and conference*, Hong Kong, China. December, 2015.

[54] W. Saesue, J. Zhang, and C. T. Chou. Frame-recursive block-based distortion estimation model for multiple reference frames and motion copy concealment in H.264/AVC. In *Proc. Packet Video Workshop*, pages 56–63, Hong Kong, Dec. 2010, Dec. 2010.

[55] W. Saesue, J. Zhang, and C. T. Chou. Hybrid frame-recursive block-based distortion estimation model for wireless video transmission. In *Proc. IEEE Multimedia Signal Processing*, pages 774–779, Australia, Oct. 2008, Oct. 2008.

[56] A. Secker and D. Taubman. Highly scalable video compression with scalable motion coding. *IEEE Trans. Image Process.*, 13(8):1029–1041, 2004.

[57] F. Shao, G. Jiang, W. Lin, M. Yu, and Q. Dai. Joint bit allocation and rate control for coding multi-view video plus depth based 3d video. *IEEE Trans. Multimedia*, 15(8):1843–1854, 2013.

[58] G. Shen, W.S. Kim, A. Ortega, J. Lee, and H. Wey. Edge-aware intra prediction for depth-map coding. *Proc. IEEE Conf. on Image Process.*, pages 3393 – 3396, Sept. 2010.

[59] A. Shokrollahi. Raptor codes. *IEEE Transactions on Information Theory*, 52(6):2551–2567, Jun. 2006.

[60] H. Shum and S. B. Kang. Review of image-based rendering techniques. *Proc. of SPIE*, 4067:2 –13, May 2000.

[61] K. Stuhlmuller, N. Farber, M. Link, and B. Girod. Analysis of video transmission over lossy channels. *IEEE J. Sel. Areas Commun.*, 18(6):1012–1032, Jun. 2000.

[62] K. Takahashi. Theoretical analysis of view interpolation with inaccurate depth information. *IEEE Trans. Image Process.*, 21(2):718–732, February 2012.

[63] M. Tanimoto. Overview of free viewpoint television. *Signal Processing: Image Communication*, 21(6):454 –461, Jul. 2006.

[64] A. Telea. An image inpainting technique based on the fast marching method. *J. Graphics, GPU, Game Tools*, 9(1):25–36, 2004.

[65] D. Tian, P.-L. Lai, P. Lopez, and C. Gomila. View synthesis techniques for 3D video. *Proc. of SPIE*, 7443:74430T–1–11, 2009.

[66] T. D. Tran, J. Liang, and C. Tu. Lapped transform via time-domain pre- and post-filtering. *IEEE Trans. Signal Process.*, 51 (6):1557–1571, Jun. 2003.

[67] Q. Wang, X. Ji, Q. Dai, and N. Zhang. Free viewpoint video coding with rate-distortion analysis. *IEEE Trans. Circuits Syst. Video Technol.*, 22(6):875–889, 2012.

[68] Y. Wang, Z. Wu, and J.M Boyce. Modeling of transmission-loss-induced distortion in decoded video. *IEEE Trans. Circuits Syst. Video Technol.*, 16(6):716–732, Jun. 2006.

[69] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4):600–612, April 2004.

[70] X. Wu, K. U. Barthel, and W. Zhang. Piecewise 2D autoregression for predictive image coding. *Proc. IEEE Conf. on Image Process.*, pages 901–904, Chicago, IL. Oct. 1998.

[71] J.-Z. Xu, F. Wu, J. Liang, and W. Zhang. Directional lapped transforms for image coding. *IEEE Trans. Image Process.*, 19 (1):85–97, Jan. 2010.

[72] H. Yang and K. Rose. Optimizing motion compensated prediction for error resilient video coding. *IEEE Trans. Image Process.*, 19(1):108–118, Jan. 2010.

[73] H. Yang and K. Rose. Advances in recursive per-pixel end-to-end distortion estimation for robust video coding in H.264/AVC. *IEEE Trans. Circuits Syst. Video Technol.*, 17(7):845–856, Jul. 2007.

[74] H. Yang and K. Rose. Advances in recursive per-pixel end-to-end distortion estimation for robust video coding in H.264/AVC. *IEEE Trans. Circuits Syst. Video Technol.*, 17(7):845–856, Jul. 2007.

[75] Y. Ye and M. Karczewicz. Improved H.264 intra coding based on bidirectional intra prediction, directional transform, and adaptive coefficient scanning. *Proc. IEEE Conf. on Image Process.*, pages 2116–2119, Oct. 2008.

[76] C. Yeo, Y. H. Tan, Z. Li, and S. Rahardja. Mode-dependent transforms for coding directional intra prediction residuals. *IEEE Trans. Circuits Syst. Video Technol.*, 22(4):545–554, April 2012.

[77] H. Yuan, Y. Chang, J. Huo, F. Yang, and Z. Lu. Model-based joint bit allocation between texture videos and depth maps for 3-D video coding. *IEEE Trans. Circuits Syst. Video Technol.*, 21(4):485–497, April 2011.

[78] H. Yuan, J. Liu, H. Xu, Z. Li, and W. Liu. Coding distortion elimination of virtual view synthesis for 3D video system: Theoretical analyses and implementation. *IEEE Trans. Broadcasting*, 58(4):558–568, December 2012.

[79] B. Zeng and J. Fu. Directional discrete cosine transforms - a new framework for image coding. *IEEE Trans. Circuits Syst. Video Technol.*, 18(3):305 – 313, March 2008.

[80] C. Zhang and D. Florencio. Analyzing the optimality of predictive transform coding using graph-based models. *IEEE Signal Process. Lett.*, 20(1):106–109, Jan. 2013.

[81] D. Zhang. Packet loss protection for H.264-based video conferencing. *Master of Applied Science Thesis, Simon Fraser University*, Dec. 2010.

[82] D. Zhang and J. Liang. Graph-based transform for 2-D piecewise smooth signals with random discontinuity locations. *in preparation for IEEE Trans. Image Process.*

[83] D. Zhang and J. Liang. View synthesis distortion estimation with graphical model and recursive calculation of probability distribution. *IEEE Trans. Circuits Syst. Video Technol.*, 25(5):827 – 840, May 2015.

[84] D. Zhang and J. Liang. Graph-based transform for 2D piecewise smooth signals with random discontinuities. *Data Compression Conference (DCC)*, Utah, USA, Mar. 2016.

[85] D. Zhang, J. Liang, and X. Gong. Distortion estimation for two-step view synthesis. *Picture Coding Symposium (PCS)*, pages 93–96, San Jose, CA. December 2013.

[86] D. Zhang, J. Liang, and I. Singh. End-to-end distortion estimation for H.264 with unconstrained intra prediction. In *Proc. IEEE Conf. on Image Process.*, pages 697–700, Orlando, FL, USA, Oct. 2012, 2012.

[87] D. Zhang, J. Liang, and I. Singh. Fast transmission distortion estimation and adaptive error protection for H.264/AVC-based embedded video conferencing systemss. *Signal Processing: Image Communication*, 28(5):417 –429, May 2013.

[88] Q. Zhang, P. An, Y. Zhang, and Z. Zhang. Efficient rendering distortion estimation for depth map compression. *Proc. IEEE Conf. on Image Process.*, pages 1105–1108, September 2011.

[89] R. Zhang, S. L. Regunathan, and K. Rose. Video coding with optimal intra/inter mode switching for packet loss resilience. *IEEE J. Sel. Areas Commun.*, 18(6):966–976, Jun. 2000.

[90] T. Zhang, X. Fan, D. Zhao, and W. Gao. New distortion model for depth coding in 3DVC. *IEEE Visual Communications and Image Processing*, pages 1–6, November 2012.

[91] X. Zhang and X. Wu. Image interpolation by adaptive 2-D autoregressive modeling and soft-decision estimation. *IEEE Trans. Image Process.*, 17(6):887 – 896, Jun. 2008.

[92] Y. Zhang, W. Gao, Y. Lu, Q. Huang, and D. Zhao. Joint source-channel rate-distortion optimization for H.264 video coding over error-prone networks. *IEEE Trans. Multimedia*, 9(3):445–454, Apr. 2007.

[93] Y. Zhang, S. Kwong, L. Xu, S. Hu, G. Jiang, and C. C. J. Kuo. Regional bit allocation and rate distortion optimization for multiview depth video coding with view synthesis distortion model. *IEEE Trans. Image Process.*, 22(9):3497–3512, 2013.

[94] Y. Zhao, C. Zhu, Z. Chen, and L. Yu. Depth no-synthesis-error model for view synthesis in 3-D video. *IEEE Trans. Image Process.*, 20(8):2221–2228, August 2011.