Encoder-Driven Inpainting Strategy in Multiview Video Compression

Yu Gao, *Member, IEEE*, Gene Cheung, *Senior Member, IEEE*, Thomas Maugey, *Member, IEEE*, Pascal Frossard, *Senior Member, IEEE*, and Jie Liang, *Senior Member, IEEE*

***Abstract*— In free viewpoint video systems, a user has the freedom to select a virtual view from which an image of the 3D scene is rendered, and the scene is commonly represented by color and depth images of multiple nearby viewpoints. In such representation, there exists data redundancy across multiple dimensions: 1) a 3D voxel may be represented by pixels in multiple viewpoint images (inter-view redundancy); 2) a pixel patch may recur in a distant spatial region of the same image due to self-similarity (inter-patch redundancy); and 3) pixels in a local spatial region tend to be similar (inter-pixel redundancy). It is important to exploit these redundancies during inter-view prediction toward effective multiview video compression. In this paper, we propose an encoder-driven inpainting strategy for inter-view predictive coding, where explicit instructions are transmitted minimally, and the decoder is left to independently recover remaining missing data via inpainting, resulting in lower coding overhead. In particular, after pixels in a reference view are projected to a target view via depth-image-based rendering at the decoder, the remaining holes in the target view are filled via an inpainting process in a block-by-block manner. First, blocks are ordered in terms of difficulty-to-inpaint by the decoder. Then, explicit instructions are only sent for the reconstruction of the most difficult blocks. In particular, the missing pixels are explicitly coded via a graph Fourier transform or a sparsification procedure using discrete cosine transform, leading to low coding cost. For blocks that are easy to inpaint, the decoder independently completes missing pixels via template-based inpainting. We apply our proposed scheme to frames in a prediction structure defined by JCT-3V where inter-view prediction is dominant, and experimentally we show that our scheme achieves up to 3-dB gain in peak-signal-to-noise-ratio in reconstructed image quality over a comparable 3D-High Efficiency Video Coding implementation using fixed 16 × 16 block size.**

***Index Terms*— Video compression, predictive encoding, transform coding.**

Manuscript received July 17, 2014; revised January 27, 2015 and September 20, 2015; accepted October 18, 2015. Date of publication November 5, 2015; date of current version December 3, 2015. This work was supported by the Natural Sciences and Engineering Research Council of Canada under Grant RGPIN312262 and Grant STPGP447223. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Chang-Su Kim.

Y. Gao was with the School of Engineering Science, Simon Fraser University, Burnaby, BC, Canada. He is now with Evertz Microsystems, Burlington, ON, Canada (e-mail: gaoyug@sfu.ca).

G. Cheung is with the National Institute of Informatics, Tokyo, Japan (e-mail: cheung@nii.ac.jp).

T. Maugey was with the Signal Processing Laboratory, Ecole Polytechnique Fédérale de Lausanne, Lausanne CH-1015, Switzerland. He is now with the French Institute for Research in Computer Science and Automation, Rennes, France (e-mail: thomas.maugey@inria.fr).

P. Frossard is with the Signal Processing Laboratory, Ecole Polytechnique Fédérale de Lausanne, Lausanne CH-1015, Switzerland (e-mail: pascal.frossard@epfl.ch).

J. Liang is with the School of Engineering Science, Simon Fraser University, Burnaby, BC, Canada (e-mail: jiel@sfu.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2015.2498400

I. Introduction

In FREE viewpoint video systems [1], color maps (RGB images) and depth maps[1] (per-pixel distance between physical objects and capturing cameras) of the 3D scene as observed from multiple closely spaced cameras are captured at the encoder into a *color-plus-depth* representation [4]. Armed with color and depth images of multiple views, images of intermediate virtual viewpoints can be synthesized at decoder via *depth-image-based rendering* (DIBR) [5], enabling new applications such as free viewpoint TV [1], immersive video conferencing [6], etc.

It is apparent that this multiview color-plus-depth representation contains various types of data redundancy spatially and across views. First, a voxel of an object in the 3D scene that is visible from multiple camera-captured views will be represented as pixels in multiple viewpoint images (*inter-view redundancy*). Assuming that the 3D scene is Lambertian,[2] a 3D voxel reflects the same color value to different viewpoints, and recording the same value across multiple viewpoint images leads to redundancy. Second, it is well understood that values of neighboring pixels of the same object in a viewpoint image tend to be correlated statistically (*inter-pixel redundancy*). Finally, it is observed that natural images tend to be *self-similar*: similar image patches tend to recur in different spatial regions throughout the same image (*inter-patch redundancy*). Previous computer vision research efforts have successfully exploited this nonlocal self-similarity characteristic of images for denoising [7] and inpainting [8] (completion of missing pixels in a spatial region).

While temporal redundancy is crucial for video coding, for frames where temporal prediction is either not possible (*e.g.*, random access I-frames) or not effective (*e.g.*, P-frames with distant temporal predictor frames in a pre-determined frame structure), it is critical to exploit these spatial and inter-view redundancies inherent in the color-plus-depth representation for efficient data compression. The vast majority of conventional inter-view prediction schemes [9]–[11] attempt to eliminate this redundancy following the traditional video coding paradigm of hybrid signal prediction/residual coding, *e.g.*, video coding standards like H.264 [12] and HEVC [13]. Specifically, to reconstruct a given target block, a sender transmits explicit instructions like *motion vector* (MV) to guide the receiver to the location of the most similar

[1]Depth images can be acquired directly using depth sensors [2], or computed from neighboring color images using stereo-matching algorithms [3].

[2]Reflective surfaces such as wine glasses and mirrors are not Lambertian. However, for closely spaced capturing cameras, the Lambertian surface assumption is nonetheless a good approximation.

block, which serves as a predictor signal. Then, the difference between the predictor and target block—the *prediction residual*—is transform coded and transmitted to the receiver to improve the reconstruction quality. This paradigm has a long legacy in video coding research, dating back to the first ISO video coding standard MPEG1 and ITU-T standard H.261 in the late 1980's, where one of the crucial design criteria was a computationally inexpensive video decoder. In that light, the hybrid signal prediction/residual coding paradigm where the encoder dictates exactly how each code block should be reconstructed is a suitable design choice that results in today's many practical codecs across many platforms.

Given that the cost of computation has drastically decreased, the strict requirement that the video decoder must be computationally simple is no longer necessary in many practical cases. In this paper, we argue that one can leverage on the computation power of a decoder to recover a desired signal and lower the overall transmission cost. In particular, we propose an encoder-driven inpainting strategy for inter-view predictive coding, where explicit directions are transmitted minimally from the encoder, and the decoder is left to independently recover missing pixels via inpainting, resulting in lower coding overhead. Our strategy efficiently exploits the aforementioned inter-patch, inter-pixel and inter-view redundancies inherent in color-plus-depth representation, and can complement existing 3D coding tools such as those in the 3D-HEVC standard by coding frames that depend heavily on inter-view prediction for coding efficiency.

In details, first we directly project pixels from one (or more) reference view(s) to a target view via DIBR, thus eliminating inter-view redundancy. This projection results in a number of *disocclusion holes*—spatial areas that are occluded in the reference view by foreground objects but become exposed after the view change. To complete these holes, we first order the blocks with missing pixels in terms of decreasing difficulty for inpainting. For the most difficult blocks, we transmit explicit instructions called *Auxiliary information* (AI) to guide the decoder in the reconstruction process. AI typically consists of the location information of the best predictor block for inpainting, and color values for missing pixels. The incomplete blocks typically contain known pixels projected from neighboring view via DIBR as well as missing pixels, but only the missing pixels are explicitly coded via a *graph Fourier transform* (GFT) [14]–[17] or a sparsification procedure using the *discrete cosine transform* (DCT) [18], [19], in order to achieve low coding cost. Finally, the decoder can independently complete missing pixels in the blocks that are easy to inpaint via a template-matching algorithm such as [8]. We apply our proposed inter-view prediction strategy to selected frames in a prediction structure defined by JCT-3V where inter-view prediction is dominant, and experimentally we show that our strategy can outperform a comparable implementation of 3D-HEVC using fixed block size 16×16 by up to 3 dB in PSNR in reconstructed image quality, demonstrating the potential of our inter-view predictive coding strategy for 3D video compression.

The outline of the paper is as follows. We first discuss related works in Section II. We then overview our encoder-driven inpainting based coding system in Section III. We describe our design of AI used to guide the inpainting of disocclusion holes at decoder in Section IV. Methods for sparsification of DCT and GFT of code blocks are described in Section V. The order in which the missing holes are completed is crucial in our proposal; we show that finding the optimal filling order is an NP-hard problem and present a heuristic "hard-to-easy" order in Section VI. Finally, we discuss our experimentation and conclude in Section VII and VIII, respectively.

II. Related Work

We divide our discussion of related works into two sections. We first discuss previous works in multiview image and video coding in the literature. We then discuss existing art in employing decoder-side inpainting techniques for compression of 2D image and video.

A. Multiview Image and Video Coding

Multiview image & video coding refers to the compression of multiple color images of the same 3D scene captured using an array of closely spaced cameras. Many papers on this topic focus on the efficient coding of the entire set of images by exploiting the inter-view data redundancy inherent in the camera setup. A straightforward way to achieve it is to use *disparity-compensated prediction*. Similar to motion-compensated prediction in single-view video coding, for each block in the target view, disparity compensation finds the best matching block in a reference view, then encodes and transmits the *disparity vector* (DV) and the prediction residual for reconstruction at the decoder. In [20], motion compensation and disparity compensation are combined to encode stereo sequences. The concept of *Group of Group of Pictures* (GoGOP) for inter-view prediction is introduced in [21], where a picture can refer to decoded pictures in other views even at different time instants. In [9] and [22], various modified hierarchical bidirectionally predicted structures are developed for inter-view prediction. In [23], the problem of optimal Group of Pictures (GOP) prediction structure for multiview video coding is studied. However, simple 2D translational inter-view motion assumed by disparity compensation cannot accurately represent the geometry transformation in a view-switch from one camera viewpoint image to another; hence, disparity compensation is not always efficient.

Color-plus-depth format is an alternative data representation that is particularly useful for free viewpoint video [4]. Color and depth images from multiple camera viewpoints are encoded together, and at the decoder one can synthesize novel intermediate viewpoint images via DIBR [5]. In this kind of representation, an alternative to translational disparity prediction is view-synthesis-based prediction [24]–[26], where a synthesized version of a target view is used for predictive coding. Our proposal is an example of this class of view-synthesis-based methods for inter-view predictive coding, but combines inpainting and advanced transform coding in a "hard-to-easy" block order for improved compression performance.

Layered depth video (LDV) [27] is another alternative representation of multiview video data, where in addition to

the color and depth maps of the main view, residual layers of other views not visible from the main view are constructed for coding. Because our proposal is also view-synthesis based, it is similar to LDV, but we focus on the efficient coding of the disocclusion hole pixels in a synthesized image (akin to the residual layers in LDV) through an intricate system of techniques described earlier.

As depth images are used only for view synthesis and are not themselves directly viewed, different rate-distortion (RD) optimization procedures have been designed for depth map coding to optimize the synthesized view quality [18], [19], [28], [29]. In particular, since depth images possess unique signal characteristics such as *piecewise smoothness* (PWS), new coding tools such as *graph Fourier transform* (GFT) designed specifically for depth signals have been proposed [14]–[17], [30].

Recently, HEVC has been extended to support encoding of 3D video, namely multiview video and associated depth data [31], [32], similar to the MVC extension of H.264/AVC [33]. There are mainly two types of tools employed: disparity compensation and view synthesis prediction. As discussed earlier, this is the hybrid signal prediction/residual coding paradigm used in conventional video coding standards, where the encoder dictates exactly how a target signal should be constructed at the decoder.

The key differentiator for our proposal is that we leverage on the self-discovery power of the decoder, so that the decoder can recover remaining missing pixels in the reconstructed viewpoint image via inpainting procedures. Instead of disparity compensation, our proposal exploits inter-view redundancy by mapping pixels from a reference view to a target view to create a starting image, and then transmits auxiliary information to assist the decoder in the completion of the rest of the image in a RD optimal manner, thus avoiding the aforementioned shortcomings of disparity compensation.

B. Inpainting for Image and Video Coding

Employing inpainting at the decoder to aid compression was first proposed in [34] for 2D images. In a nutshell, the work in [34] essentially advertises an advanced *intra-prediction* scheme based on (local) inpainting. This inpainting is more sophisticated than uni-directional pixel copy employed in video coding standards like H.264 intra [12], where texture in the target block is predicted using observed pixels from adjacent causal blocks that have already been reconstructed. To further improve inpainting quality, edge information (called *assistant information*) for the target block is optionally sent, so that sharp edges in the reconstructed block can be preserved. The success of [34] has inspired a set of follow-up works that also employ inpainting for 2D image and video coding [35]–[39]. For example, the authors in [37] generalize the notion of assistant information to a set of parameters for different model distributions; *i.e.*, instead of simple edges, other assistant information can be transmitted to aid inpainting. Blocks with statistics that do not fit with model distributions are classified as non-featured blocks and coded using traditional DCT-based methods. The authors in [38] propose an inpainting procedure based on Laplace partial differential equation (PDE) and total variation (TV) for HEVC intra coding, and later for depth map coding as well [39].

Though also employing inpainting at the decoder, our work differs from these works fundamentally in the following regard: inpainting for intra-prediction as described above exploits *local* inter-pixel data redundancy in images, while our proposed encoder-driven inpainting strategy exploits also data redundancy in *non-local* pixel patches due to self-similarity in images. Specifically, our inpainting scheme is derived from inpainting schemes based on template-matching such as [8] that identify and copy non-local pixel patches from distant spatial regions to the target patch. This concept is similar to our previous work [40], which is significantly extended here in many aspects. In particular, we introduce a new order of encoding/decoding in our proposal, where our goal is to first transmit the hard-to-inpaint blocks, so that the remaining blocks can be filled independently by the decoder using non-local template-matching. Second, in order to code prediction residuals or intra blocks, we introduce DCT sparsification and GFT as additional coding modes to code only unknown pixels in a target block and improve coding performance. Finally, because the "hard-to-easy" order entails non-stationary statistics over time, we design a statistical context to efficiently encode AI modes, resulting in non-negligible coding gain at low bitrates when the coding of modes accounts for a significant portion of the bit budget.

We note that template-matching-based intra prediction has been proposed previously in [41]–[44] for video coding. Specifically, DIP (displacement intra prediction) and MTP (Markovian texture prediction) in [41] share similar ideas with our proposed coding modes. However, the coding strategies are fundamentally different. These works employ the traditional top-down left-to-right block order during the encoding process. In contrast, we propose a block order so that explicit information is transmitted only for difficult-to-inpaint blocks, after which decoder must complete the remaining missing pixels via inpainting. Our unique block filling order also means that the blocks requiring completion have varying numbers of missing pixels, leading to novel transform coding techniques (graph Fourier transform and DCT spasification) that are not considered in [41]–[44].

III. Coding System Overview

A. System Overview

We propose a coding strategy based on encoder-driven inpainting for color-plus-depth representation of multiview video data. Specifically, the objective of our coding strategy is to code color and depth image pairs of multiple views in a RD optimal manner. For the sake of simplicity, we describe our coding strategy for two neighboring viewpoint images (two pairs of color and depth images). The application of our strategy to multiple views with more complex frame prediction structure is straightforward, and experimental results will be reported in Section VII.

We first independently code color and depth images of one view with the help of a video codec; this view is called the

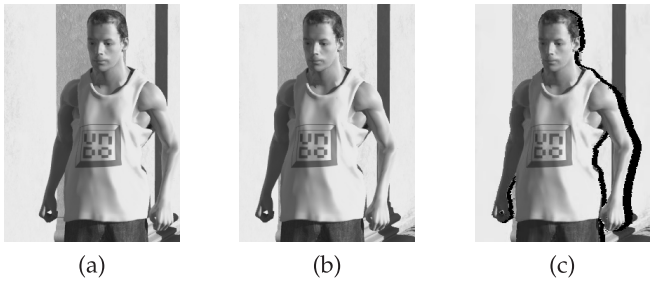


Fig. 1. Illustration of disocclusion holes left after DIBR for sequence `Undo_Dancer`. The forward warping via DIBR is performed from view 1 to view 5. The black regions close to the right side of the foreground (the man) show the missing pixels that have to be filled after projection. (a) view 1. (b) original view 5. (c) rendered view 5.

independent view. Then we propose to code the second view (the *dependent view*) as follows. We project the decoded color pixels of the independent view to the dependent view image via DIBR, where the geometric information provided by the depth map of the independent view and the camera calibration parameters are used to identify pixel correspondences between the two views. We assume in this work that the view-to-view projections via DIBR are of sufficient quality and do not require further refinements at the decoder.[3] After the projection step, there are two kinds of holes in the DIBR-synthesized image that require filling in order to have a good reconstruction of the dependent view; these are: i) *rounding holes*, and ii) *disocclusion holes*. First, the disparity values of the pixels in the independent view likely have fractional values; when projected to the dependent view they are rounded to the nearest integers to land on the 2D grid of the rendered image. This rounding operation can result in rounding holes. By assuming a rectified camera setup where the relative camera positions correspond to pure horizontal shifts, the pixel disparities have only horizontal components. Thus we can identify the rounding holes simply as follows. For a given hole in the projected image, we check if the nearest available pixels to the left and right of the hole have almost identical depth values (namely, less than a pre-defined threshold δ). If so, we identify the hole as a rounding hole and perform simple horizontal linear interpolation to fill in the missing pixel(s).

After identification and filling of the rounding holes in the projected image, disocclusion and out-of-view holes remain. Out-of-view holes are spatial regions that newly enter the visible image real estate due to the change of camera position. Our coding schemes can efficiently treat them in the same way as the disocclusion holes.[4] Disocclusion holes represent spatial regions in the dependent view that are not visible in the independent view due to occlusion by foreground objects. An example of such disocclusion holes is shown in Fig. 1.[5] Unlike rounding holes, disocclusion holes may

[3]If projections are poor due to illumination differences in color maps or estimation errors in depth maps, they should be corrected in a pre-processing step prior to the start of encoding for better coding performance.

[4]We thus use disocclusion holes to denote the two types of holes hereafter.

[5]Video sequence `Undo_Dancer` can be found from ftp://mpeg3dv.research.nokia.com

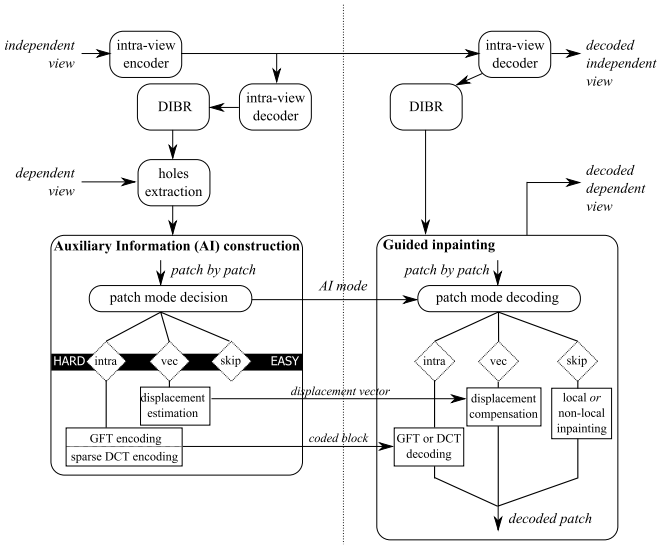


Fig. 2. The block diagram of the proposed strategy. Dependent views are obtained by DIBR estimation from the decoded independent view, along with encoder-driven inpainting of the disocclusion holes.

contain novel information that cannot be easily extrapolated from available neighboring pixels. Hence, the encoder has to provide information to the decoder so that it can properly reconstruct the dependent view. In this paper, we assume that the decoder has the computational resources to execute inpainting procedures. Thus the encoder only provides carefully chosen *auxiliary information* (AI) to guide the decoder through the reconstruction of difficult spatial regions, so that the decoder can self-discover missing pixels in the remaining holes in the dependent view via inpainting. The construction of this AI data is described in the next section.

The depth pixels of the independent view are also projected to the dependent view, and rounding holes are identified and filled in the same manner as in the color image.[6] However, the disocclusion holes are simply extrapolated using adjacent background depth pixels. This is because depth images are known to be piecewise smooth [14]–[16]. We further find empirically that adjacent background depth pixels are good predictors for signal extrapolation into the disocclusion holes. The overall procedure of the proposed coding strategy is summarized in Fig. 2.

B. Encoder-Driven Disocclusion Hole Filling

We provide now more details about our coding strategy that relies on inpainting to fill in disocclusion holes. In the computer vision literature, there exist many inpainting algorithms [8], [46]–[49] to complete holes in a given image. The key difference between our work and inpainting schemes like [8] is that, in our multiview coding scenario, the target image to be inpainted is known at the encoder. Hence, the encoder can provide additional information to guide the image completion process at the decoder. Inspired by the patch-based template-matching inpainting algorithm in [8], we will employ

[6]The corresponding depth values for the two views are the same if we assume the two viewpoints are rectified [45].

a similar inpainting framework and complete the rendered image on a per-patch basis. In a nutshell, our patch-based inpainting framework performs the following operations. We first select a pixel on the boundary of disocclusion holes in the DIBR projection of the dependent view; the selected pixel is the center of a target patch that will be inpainted. Missing pixels in the target patch are then filled using known pixels in the reconstructed dependent view via template-matching, possibly with help of AI provided by the encoder. Then, another target patch is selected for filling, and the process continues until all missing pixels are completed. The order in which the patches are selected for filling is called the *patch order*. Given this patch-based inpainting framework, there are two key questions to solve for effective coding performance: i) for a given target patch, how to best complete it, possibly with the aid of AI? ii) what is the optimal patch order to complete the rendered image?

We first observe that, given a local target patch to be completed, the level of difficulty in inpainting it—called *local hardness* in the sequel—depends on the degree of self-similarity exhibited between the target patch and already completed spatial regions in the predicted image. If the missing pixels in the target patch can be found in the completed spatial regions, then the encoder only needs to convey a simple message (called the *skip* mode) to the decoder to signal that the missing information can be self-discovered using an unassisted inpainting procedure such as template-matching [8]. If the missing pixels are available in the completed spatial regions but it is difficult for the decoder to discover them unassisted, then the encoder can provide lightweight search instructions (called the *vec* mode) to guide the discovery process. Finally, if the missing pixels are entirely innovative, then the encoder has no choice but to explicitly encode the missing pixels (called the *intra* mode).

Note that *only the subset of missing pixels in a target block requires intra coding*. This observation can be exploited for novel patch-based transform coding that can outperform conventional schemes like DCT that typically code the entire pixel block, regardless of whether pixels in the block are missing or not. These different kinds of information compose the set of AI that the encoder convey to the decoder on a patch-by-patch basis for encoder-guided inpainting. Finally, we remark that the different kinds of AI have different coding costs, and the choice of AI is determined by a RD criterion. The details of the AI design is discussed in Section IV and the patch-based transform coding is described in Section V.

The second question is related to the order of patches in the inpainting process. Clearly, a left-to-right top-to-bottom raster scanning order employed in conventional block-based image / video coding algorithms is not appropriate. A key innovation in Criminisi's inpainting algorithm [8] is the order in which target patches should be selected for inpainting: the main idea is to select easy-to-inpaint patches first, so that propagation of likely errors in hard-to-inpaint patches to other regions will be minimized. This patch ordering problem is called the *global hardness* of patches in the inpainting process. In stark contrast to the "easy-to-hard" patch order in Criminisi's algorithm, we propose a "hard-to-easy" patch order for our encoder-assisted inpainting algorithm. The basic idea is that, once the hardest patches are filled in (with ample assistance from the encoder), the remaining patches are all easy-to-inpaint. They can be completed by the decoder unassisted, and hence the encoder can directly save bits from reduction in AI signaling cost. Note that the problem of error propagation from hard-to-inpaint patches to other spatial regions can be easily contained in our setting, since the encoder-guided inpainting process can implicitly control the inpainting quality at the decoder. The details of our proposed "hard-to-easy" patch order is presented in Section VI.

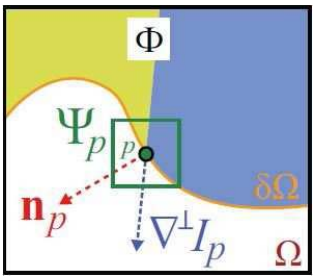


Fig. 3. Notation diagram of Criminisi's inpainting algorithm [8]. The regions Φ and Ω respectively denote the known and unknown region in the inpainting process.

Note that we focus on the development of new coding tools for inter-view predictive coding, rather than the optimization of the inter-view prediction frame structure itself, which is outside the scope of this paper.

IV. Inpainting Based Coding and Auxiliary Information

Towards a solution to address local hardness in encoder-driven image inpainting, we first overview a well-known template-matching inpainting algorithm [8]—we use a similar variant in our system. We then discuss the design and implementation of auxiliary information (AI).

A. Overview of Criminisi's Inpainting Algorithm

Criminisi's inpainting algorithm [8] is a well-known method to propagate texture patterns from known pixel regions to spatial regions with missing pixels, assuming that self-similarity typically exists in natural images. While there are more recent inpainting algorithms designed specifically for DIBR-synthesized images [50]–[53], we construct our algorithm based on [8] for its simplicity. For convenience and consistency, we reuse some notations from Criminisi's algorithm. We denote the image by I, and denote by Φ and $\Omega = I \setminus \Phi$ respectively the source and target regions in the inpainting process. As illustrated in Fig. 3, the pixels in Ω that are neighbors of Φ form the boundary, denoted by $\delta\Omega$. The region Φ includes the rendered and decoded pixels, while Ω represents the remaining holes. A square-shaped target patch Ψ_p centered at pixel p on the boundary $\delta\Omega$ is chosen by a given patch selection procedure, namely the easy-to-hard order in Criminisi's algorithm. The patch Ψ_p has two non-overlapping parts: the known region $\Psi_p \cap \Phi$ (also called *template* in the sequel) and the target unknown region $\Psi_p \cap \Omega$.

For a given a target patch Ψ_p, *template matching* is performed in [8] to find the patch Ψ_q in Φ that is the most similar to known pixels in $\Psi_p \cap \Phi$:

$$\Psi_q^* = \arg\min_{\Psi_q \in \Phi} d(\Psi_p, \Psi_q) \tag{1}$$

where the distortion term $d()$ is computed using only known pixels in Ψ_p and their corresponding pixels in Ψ_q. After the optimal Ψ_q^* is found, the pixels in Ψ_q^* that correspond to missing pixel locations in $\Psi_p \cap \Omega$ will be copied over for completion of Ψ_p.

B. Encoder-Driven Patch Completion

Given a target patch Ψ_p, we now discuss our encoder-driven patch completion procedure using AI to fill in the missing pixels in Ψ_p. In a nutshell, we seek to design a set of AI modes $\{\varphi\}$ for the completion of missing pixels in the target region and to eventually choose for each target patch Ψ_p the AI mode that minimizes the RD cost, *i.e.*,

$$\varphi^* = \arg\min_{\varphi} d(p, \varphi) + \lambda \cdot r(p, \varphi), \tag{2}$$

In (2), $r(p, \varphi)$ is the coding rate of the mode φ for patch Ψ_p centered at p, $d(p, \varphi)$ is the distortion between missing pixels in $\Psi_p \cap \Omega$ and the reconstructed pixels using mode φ, and λ is a pre-defined weighting parameter that trades off the importance of distortion and rate. We use sum of squared differences as distortion in this paper. The index of the coding mode in (2) is compressed via a context-based arithmetic coder.

In this paper, we design three AI modes with different coding costs $r(p, \varphi)$ and different degrees of influence on patch reconstruction. The AI "skip" mode results in zero rate beyond the signaling cost of the mode itself. AI "vec" mode encodes a motion or disparity vector to inform the decoder the location of the best matching patch in the known region. The AI "intra" mode encodes the intensity of missing pixels in the target patch, and thus usually results in the highest rate. The encoder chooses among these three modes for a given target patch Ψ_p in order to solve the optimization problem in (2). We describe in details the three coding modes in the rest of this section.

C. AI Modes

1) AI "Skip": The AI "skip" mode instructs the decoder to self-discover missing pixels in Ψ_p using only information in source region Φ. This can be done either *locally* or *nonlocally*. *Local skip* means that, given the strong inter-pixel redundancy exhibited in the local patch, the missing pixels in $\Psi_p \cap \Omega$ can be deduced from neighboring known pixels via simple signal extrapolation schemes such as [46]. In contrast, *nonlocal skip* instructs the decoder to perform template matching to complete missing pixels in the target patch Ψ_p using its template $\Psi_p \cap \Phi$. This is similar to the template matching in [8], except that the search region includes not only the known region Φ in the same image, but also designated decoded pictures in the decoder's buffer. Designated pictures could include L_T previous decoded pictures in time from the same view and L_V decoded pictures of the same time instant but from neighboring views. For simplicity, only the decoded picture of previous time instant from the same view is used as reference in our paper. Local skip and nonlocal skip finally translate to different AI mode indices for arithmetic encoding.

The AI "skip" mode is expected to be a dominant mode at low rate when coding rate is of higher priority than distortion. At high rate, however, because of the lack of direct control in the pixel completion process, the patch quality reconstruction is limited. We present two other modes with more direct control on the inpainting results (reconstruction quality).

2) AI "Vec": When the template matching of "nonlocal skip" fails to identify a good match for $\Psi_p \cap \Omega$, we offer the AI "vec" mode as an alternative to improve the reconstruction quality by directly locating the pixels in the known region that are the most similar to the missing pixels in the target patch. We stress here the difference between "nonlocal skip" and "vec". "Nonlocal skip" relies on the known pixels in the target region of Ψ_p for template-matching with known patches, which may not always lead to the best completion results. The "vec" mode, on the other hand, simply informs the decoder about the location of the pixels in the known region that are the most similar to the missing pixels in the target patch; it does not rely on template-matching at all.

To leverage on both self-similarity of still image and temporal redundancy of video sequences, we propose two kinds of "vec" mode, namely intra-frame AI "vec" and inter-frame AI "vec". For intra-frame "vec", a *similarity vector* (SV) pointing to the known region in the same image is signaled to the decoder. On the other hand, the inter-frame "vec" is akin to motion estimation in differential coding for single-view video: a *motion vector* (MV) is used to represent the displacement of the current block to the most similar one in a reference picture (*i.e.*, the previous frame in time in the same view).

3) AI "Intra": When no pixels similar to $\Psi_p \cap \Omega$ are found in the search space of nonlocal "skip" and "vec" modes, *e.g.*, in the case of disocclusion of novel objects, the AI "intra" mode is used to directly code the missing pixels in the target patch Ψ_p. In this mode, the block is first predicted, then the prediction residual signal is transformed, quantized, and entropy-coded. Since the shapes of known pixels $\Psi_p \cap \Phi$ and causal neighbors of Ψ_p are arbitrary, the directional intra prediction used in conventional block-based video coding such as HEVC [13] is not suitable here. Instead, we propose to use the signal extrapolation scheme in AI "local skip" as the prediction, and to code only the resulting prediction residual. Another noticeable deviation from conventional block-based video coding is that, in our scenario, only missing pixels in a block requires coding and not the full block of pixels. We discuss in Section V two methods to encode only the missing pixels in a square target block for AI "intra".

4) Arithmetic Coding of AI Mode Indices: The coding modes are compressed with arithmetic coding. Let us consider a set of AI mode decisions $\Upsilon = \{\varphi_i\}_{i=1..N}$ for N consecutive patches. In order to transmit this vector Υ, we use an arithmetic coder, which needs the mode probabilities as input, both at the encoder side and decoder side. As mentioned above,

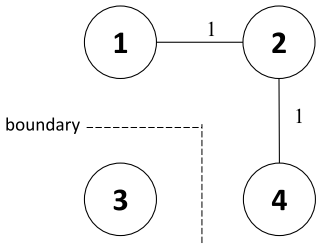


Fig. 4. An example graph from a 2×2 patch.

we have designed an encoder-driven inpainting algorithm that adopts the "hard-to-easy" order. It means that "hard" modes such as AI "intra" or "vec" are chosen more frequently at the beginning, while "easy" mode AI "skip" is likely chosen at the end of the coding process. In order to take into account this evolution in the coding of Υ, we adapt the input probabilities of the arithmetic coder. For a given mode flag φ_i to code, we evaluate the probabilities $p_{i,l}$ of each coding mode $l = 1, \ldots, L$ over the W last mode decisions

$$p_{i,l} = \frac{\sum_{j=1}^{W} b_{i-j,l}}{W}, \tag{3}$$

where

$$b_{i-j,l} = \begin{cases} 1 & \text{if } \varphi_{i-j} = l \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

The L probabilities, available at the decoder side also, are the contexts of our arithmetic coding of mode indices. Note that we code the mode index directly and do not have a binarization process.

V. Transform Coding of Missing Patch Pixels

In our proposed patch-based coding scheme, the center p of a $K \times K$ target square patch Ψ_p is always on the boundary of known and unknown regions as shown in Fig. 3. Hence, the patch Ψ_p contains known pixels in $\Psi_p \cap \Phi$ as well as missing pixels in $\Psi_p \cap \Omega$. If one naïvely use regular block-based DCT to encode the patch (or the prediction residual of the patch), then the resulting $K \times K$ transform coefficients will contain information belonging to both known and unknown pixels, resulting in undesirable representation redundancy. In this section, we propose two block-based coding procedures to encode only the missing pixels in a patch, namely i) the graph Fourier transform (GFT), and ii) the sparsification of DCT.

A. Graph Fourier Transform

GFT has recently been proposed for transform coding of a block in a piecewise smooth image like a depth map that straddles a sharp boundary, so that filtering across discontinuities is avoided. This results in a sparser signal representation in transform domain than DCT [14]–[17]. The key idea is to represent pixels in the block as nodes in a graph $\mathcal{G}$, and connect each pixel with each of its four neighbors with an edge of weight 1 only if both pixels reside on the same side of a boundary. In essence, a block of pixels is divided into connected sub-graphs, as illustrated in Fig. 4.

The graph $\mathcal{G}$ is described by a few matrices. First, an *adjacency matrix* $\mathbf{A}$ describes the connectivity of the graph $\mathcal{G}$, where $A_{i,j} = 1$ if nodes i and j are connected via an edge and 0 otherwise. For the graph in Fig. 4, the adjacency matrix is

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}. \tag{5}$$

A *degree matrix* $\mathbf{D}$ is a diagonal matrix, where $D_{i,i} = \sum_j A_{i,j}$. For the graph in Fig. 4, the degree matrix is

$$D = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6}$$

Finally, the *graph Laplacian* matrix $\mathbf{L}$ is defined as the difference between the degree matrix and the adjacency matrix [54]:

$$\mathbf{L} = \mathbf{D} - \mathbf{A}. \tag{7}$$

For the graph in Fig. 4, the graph Laplacian matrix is

$$L = \begin{bmatrix} 1 & -1 & 0 & 0 \\ -1 & 2 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix}. \tag{8}$$

GFT is then defined as follows. It is a linear transform matrix Φ composed of the eigenvectors of $\mathbf{L}$, *i.e.*, $\mathbf{L}\phi_i = \lambda_i \phi_i$, where ϕ_i is the i-th row of Φ written as a column vector, and λ_i is the i-th eigenvalue of Φ, which could be seen as the i-th *graph frequency* of the graph $\mathcal{G}$. A given pixel block $\mathbf{x}$ is then interpreted as a graph-signal on graph $\mathcal{G}$. After computing GFT coefficients $\alpha = \Phi\mathbf{x}$, the coefficients α are quantized and entropy-coded. Unlike block transforms such as DCT where the same transform is applied for every pixel block, GFT is an *adaptive* transform; *i.e.*, different signal-dependent transforms Φ are used for different input graph-signals $\mathbf{x}$, since the graph construction is dependent on the signals. Previous works [14]–[17] have shown that this overhead of encoding side information to describe GFT is not expensive for depth maps, and there is overall substantial coding gain for GFT over fixed transforms like DCT for coding depth maps.

Based on the success of GFT to code depth maps, we propose here to use GFT to encode *only* the missing pixels $\Psi_p \cap \Omega$ in a given patch Ψ_p. We first construct a graph $\mathcal{G}$ only for these missing pixels: each missing pixel in $\Psi_p \cap \Omega$ is denoted by a node, and there is an edge of weight 1 connecting two nodes if they represent two missing pixels that are neighbors. See Fig. 5 for an illustration. In this way, the graph is composed only of nodes that represent the n missing pixels.

Given this graph of missing pixels, one can compute the adjacency and degree matrices $\mathbf{A}$ and $\mathbf{D}$ accordingly. The graph Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{A}$ can also be computed, and its eigen-decomposition can be performed to obtain the GFT matrix Φ. To encode missing pixels $\mathbf{x}$ (stacked together

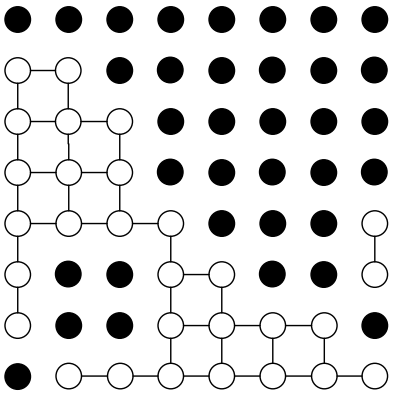


Fig. 5. An example of graph construction for an 8×8 patch. White circles denote unknown pixels. Black circles denote known pixels. Edges connect neighboring unknown pixels. The weights assigned to every edges are unity.

as a vector), we simply compute the GFT coefficients $\alpha = \Phi\mathbf{x}$, and quantize and entropy encode them into bits. Note that, unlike the classical DCT that has $K \times K$ transform coefficients for $K \times K$ pixels in the patch Ψ_p, the number of GFT coefficients is only equal to the number of missing pixels in Ψ_p. Since the locations of the missing pixels are known at decoder already, a graph can be readily constructed, and the GFT matrix Φ can be computed to permit reconstruction of the missing pixels. We will show in Section VII that usage of GFT to encode missing pixels in a patch can outperform DCT in coding performance.

B. Sparsification Procedure Using DCT

We introduce here another option to encode missing pixels in a patch by sparsification of the DCT coefficients. While the GFT leads to good coding performance, the complexity required to compute the GFT via eigen-decomposition both at the encoder and decoder can be high, especially, if the number of missing pixels is large. Compared to the GFT, the DCT sparsification procedure is less complex.

Since the values of rendered pixels in a patch are known at encoder and decoder prior to any transform encoding, the known pixels in $\Psi_p \cap \Phi$ can be viewed as degrees of freedom at encoder side: they can be manipulated in order to reduce the cost of coding the patch Ψ_p as their decoded values are simply discarded. Specifically, we propose a sparsification procedure in the DCT domain that exploits these degrees of freedom to minimize the number of non-zero DCT coefficients. The fraction of non-zero quantized transform coefficients has an approximately linear relationship with bitrate [55], hence minimization of the number of non-zero coefficients corresponds to a reduction in the coding rate.

Let $\mathbf{x}$ be pixels in a $K \times K$ patch, stacked into a column vector. Let Θ be the DCT transform; the DCT coefficients $\mathbf{y}$ can be computed simply: $\mathbf{y} = \Theta\mathbf{x}$. Let $\mathbf{V}$ be a $K^2 \times K^2$ diagonal matrix where entry $V_{i,i} = 1$ if the i-th pixel in $\mathbf{x}$ is an unknown pixel and 0 otherwise. Our objective is to minimize the rate-distortion cost of AI "intra" for the patch Ψ_p by manipulating the coefficients $\mathbf{y}$ in the transform domain

$$\min_{\mathbf{y}} \|\mathbf{V}(\Theta^{-1}\mathbf{y} - \mathbf{x})\|_2^2 \; + \; \lambda\|\mathbf{y}\|_0, \tag{9}$$

Note that $\lambda = \lambda_{\text{mode}} * B$, the product of the Lagrange multiplier used in AI mode decision (λ_{mode}) and the average bits to code a non-zero quantized coefficient (B), and hence $B\|\mathbf{y}\|_0$ is the coding rate of the patch. In Eq. (9), the l_0-norm is a rate proxy, while the l_2-norm is a measure of distortion, counting only the distortion contributed by the unknown pixels.

The minimization of the l_0-norm in (9) is in general a non-convex and NP-hard problem. For efficient computation, one can use an iterative re-weighted least squares algorithm and replace the l_0-norm with a *sparsity-promoting* weighted l_2-norm [19]:

$$\min_{\mathbf{y}} [\mathbf{V}(\Theta^{-1}\mathbf{y} - \mathbf{x})]^{\mathrm{T}} [\mathbf{V}(\Theta^{-1}\mathbf{y} - \mathbf{x})] + \mathbf{y}^{\mathrm{T}}\mathbf{W}_\lambda\mathbf{y}, \tag{10}$$

where the weight matrix is

$$\mathbf{W}_\lambda = \begin{pmatrix} \lambda w_1 & 0 & \cdots & 0 \\ 0 & \lambda w_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda w_{K^2} \end{pmatrix}, \tag{11}$$

where $\{w_1, w_2, \cdots, w_{N^2}\}$ are iteratively updated. The optimal solution to Eq. (10) can be found by solving the linear system

$$(\Theta\tilde{\mathbf{V}}\Theta^{-1} + \mathbf{W}_\lambda)\mathbf{y^o} = \Theta\tilde{\mathbf{V}}\mathbf{x}, \tag{12}$$

where $\tilde{\mathbf{V}} = \mathbf{V}^{\mathrm{T}}\mathbf{V} = \mathbf{V}$.

Iteratively updating the weights in $\mathbf{W}_\lambda$ and solving the linear system in (12) can achieve transform domain sparsity [19], [56] and minimum rate-distortion cost given λ. The detailed procedure is written in Algorithm 1. After the algorithm converges, the optimal transform coefficients are quantized and entropy coded. Finally, we note that the parameter τ in the iterative weight computation process (see Algorithm 1) can control the speed of the algorithm, *i.e.* the complexity of encoder, while at decoder side we only need to inverse-transform the received coefficients.

Algorithm 1 Iterative Re-Weighted Least Squares (IRLS) for DCT Coefficients Sparsification

1. Initialize weights: $w_i = 1/(|y_i^t| + \epsilon)^2$, where $y_i^t = (\Theta\mathbf{x})_i$.
2. Find the solution to Eq. (12): y_i^o.
3. Update weights: $w_i = (|y_i^o|^2 + \epsilon^2)^{-\frac{2-\tau}{2}}$ if $|\frac{y_i^o}{Q}| \geq 0.5$; $w_i = \epsilon^{2-\tau}$ otherwise, where Q is quantization step.
4. Repeat Step 2 to 3 until convergence.

VI. "Hard-to-Easy" Order for Target Patch Selection

In this section, we address the following problem: how target patches Ψ_p in the target region Ω should be selected for completion. The order of target patches to be filled can be denoted by a sequence of positions p on the boundary between known and target regions, *i.e.*, $p \in \delta\Omega_t$, where Ω_t denotes the target region that contains all missing pixels at

iteration t, until all missing pixels in the image are filled. Our goal is to find a patch order so that the overall RD cost to fill all missing pixels in Ω_0 is minimized. The total number of possible orders is, in the worst case, exponential in the number of missing pixels in Ω_0, so clearly an exhaustive search for the optimal order is not practical.

We discussed earlier that the Criminisi's inpainting algorithm [8] proposed an "easy-to-hard" order for patch selection to minimize the chance of error propagation from hard-to-fill patches to other spatial regions. However, in our encoder-driven inpainting framework, the encoder can transmit AI to guide the decoder in completing missing pixels. Therefore, the error propagation from hard-to-fill patches can be contained proactively, and Criminisi's order is not necessarily the optimal order in this case.[7] In this section, we first show that finding the optimal order is an NP-hard problem. We then propose a heuristic "hard-to-easy" order, which can be computed in polynomial time, and has better performance than the Criminsi's order.

A. NP-Hardness Proof for Patch Selection Order

In the most general setting, the optimal patch ordering problem can be formulated as follows. Let $\mathbf{P}_t = \{p_t, \ldots, p_1\}$ be the first t selected patch centers, and let $\Upsilon_t = \{\varphi_t, \ldots, \varphi_1\}$ be the t selected AI modes for the first t selected patches $\mathbf{P}_t$. Assuming that it requires T selected patches before all the missing pixels are filled in the initial target region Ω_0, the optimal patch order, expressed in patch centers and AI modes $\mathbf{P}_T^*$ and Υ_T^*, is defined as:

$$(\mathbf{P}_T^*, \Upsilon_T^*) = \arg\min_{\mathbf{P}_T, \Upsilon_T} \sum_{t=1}^{T} d(p_t, \varphi_t | \mathbf{P}_{t-1}, \Upsilon_{t-1}) + \lambda \cdot r(p_t, \varphi_t | \mathbf{P}_{t-1}, \Upsilon_{t-1}) \quad (13)$$

where $d(p_t, \varphi_t | \mathbf{P}_{t-1}, \Upsilon_{t-1})$ and $r(p_t, \varphi_t | \mathbf{P}_{t-1}, \Upsilon_{t-1})$ are respectively the distortion and rate of completing the patch centered at p_t using mode φ_t, given previous selected patch centers and modes $\mathbf{P}_{t-1}$ and Υ_{t-1}. The selected patch centers and modes $\mathbf{P}_T^*$ and Υ_T^* must satisfy two conditions. First, each center p_t must lie on the boundary $\delta\Omega_t$, where the target region of missing pixels Ω_t for each iteration t is updated as follows:

$$\Phi_t = \Phi_{t-1} \cup \Psi_{t-1}, \quad \Omega_t = I \backslash \Phi_t \quad (14)$$

In words, the known region Φ_t in iteration t is updated with completed pixels in patch Ψ_{t-1}, and Ω_t is the target region of remaining missing pixels.

Second, by completing all T patches, there should be no remaining pixels:

$$\Omega_{T+1} = \emptyset \quad (15)$$

The optimal patch order problem in (13) is hard for two reasons. First, the distortion term $d(p_t, \varphi_t | \mathbf{P}_{t-1}, \Upsilon_{t-1})$ depends on the history of previously selected modes Υ_{t-1}, where each mode φ_t can be selected from a discrete AI mode set $\{\varphi\}$. This means that the total number of these distortion terms $d()$ is *at least* on the order of $|\{\varphi\}|^T$, *i.e.*, exponential in the number of selected patches T. Thus, the time required just for data collection of these terms is time-consuming. This is analogous to the dependent quantization problem for video coding [57], where the distortion $d_t(Q_t | Q_{t-1}, \ldots, Q_1)$ of a differentially coded frame t depends on not only its own quantization parameter (QP) Q_t, but also QPs of all previous frames in the dependency chain as well.

Second, we have the difficulty of choosing an appropriate patch order for mode selection in our problem. This means that, in addition to the set of patch centers $\mathbf{P}_{t-1}$ selected in previous iterations, the order in which these patch centers have been selected also influences the rate term $r(p_t, \varphi_t | \mathbf{P}_{t-1}, \Upsilon_{t-1})$ and the distortion term $d(p_t, \varphi_t | \mathbf{P}_{t-1}, \Upsilon_{t-1})$. To illustrate this second difficulty, let us consider the simple case where the rate term $r(p_t, \varphi_t | \mathbf{P}_{t-1}, \Upsilon_{t-1})$ depends *only* on the current patch center and the lone previous patch center, *i.e.*, $r(p_t | p_{t-1})$. This corresponds to the case where the location of the next patch center p_t is differentially coded from the previous center p_{t-1}, while the AI mode coding cost is negligible, resulting in a rate cost $r(p_t | p_{t-1})$. We will assume that the rate cost for the first patch center $r(p_1)$ is the same for all centers, and therefore can be safely ignored in the optimization. To further simplify our complexity analysis, we also assume that the distortion cost is negligible; this will correspond to the case when λ in (13) is set so large that the rate term dominates. We now show that even in this special case, the optimal patch order problem is NP-hard via a reduction from a well-known NP-hard problem—the *traveling salesman problem* (TSP) [58].

TSP is formulated as follows. There exists a finite set of cities $\mathcal{C} = \{c_1, \ldots, c_M\}$ and a distance $l(c_i, c_j)$ between each pair of cities $c_i, c_j \in \mathcal{C}$. The question is how to find a tour of all the cities, $\pi = \{\pi(1), \ldots \pi(M)\}$, where $\pi(m) \in \mathcal{C}$, such that the total distance traveled is minimized:

$$\pi^* = \arg\min_{\pi} L(\pi) = \sum_{i=1}^{M-1} l(c_{\pi(i)}, c_{\pi(i+1)}) + l(c_{\pi(M)}, c_{\pi(1)}) \quad (16)$$

TSP remains NP-hard if we do not require a cycle and remove the last distance term $l(c_{\pi(M)}, c_{\pi(1)})$.

We now argue that the above simple case of patch selection includes TSP as a special case. First, we construct M non-overlapping patches that require separate filling in the target region Ω_0; each patch i will correspond to a city $c_i \in \mathcal{C}$ in the TSP problem. Then we set the rate cost $r(i | j)$ of selecting patch center i after previous selection of patch center j, as well as the reverse $r(j | i)$, to be $l(c_i, c_j)$ in TSP. It is now clear that the optimal patch order in our simplified problem—one that minimizes the total rate $\sum_{t=2}^{M} r(p_t | p_{t-1})$—maps to a minimum distance tour in TSP. Hence, our optimal patch order problem is at least as hard as TSP, which means that our optimal patch order problem is actually NP-hard.

B. "Hard-to-Easy" Order

Given that the optimal patch order problem in (13) is NP-hard, we propose a simple "hard-to-easy" heuristic to

[7] Interestingly, one can argue that at zero rate, Criminisi's "easy-to-hard" order is a good solution in RD performance. We will in fact show our proposed order defaults to the Criminsi's order when the rate constraint is extremely tight.

determine a good patch order. The key idea is that, if all the difficult-to-fill patches are first filled, then the missing pixels in the remaining easy-to-fill patches can be self-discovered at the decoder exploiting self-similarity in images, such that no more AI is required. Further, bundling the difficult-to-fill patches in the beginning of AI coding means that there is stronger statistical correlation among chosen modes for these patches, resulting in coding gain when the chosen modes are compressed using arithmetic coding, as described in Section IV.

In order to determine the "hard-to-easy" order, for each iteration t of the inpainting algorithm we compute a metric for each candidate target patch Ψ_p centered at $p \in \delta\Omega_t$ using known pixels in Φ_t. The metric is the distortion between candidate patch Ψ_p and the best matching block Ψ_q in Φ_t chosen via template-matching, see Eq. (1), and computed using only the known pixels in Ψ_p, *i.e.*, $\Psi_p \cap \Phi_t$. The candidate patch Ψ_p with the largest metric value will be deemed the hardest and is selected as the next target patch for filling. The intuition here is that the candidate patch Ψ_p with no good matching patch Ψ_q in the known region Φ_t likely lacks the self-similarity characteristics that are required for nonlocal template matching to recover missing pixels. Hence this patch is deemed difficult-to-fill. Note that using this method, there is no need to explicitly inform the decoder about the location of the next target patch center p, as it can perform the exact same operations as the encoder to deduce the same target patch location.

The patch selection process is first computed until there are no missing pixels left. Then, a binary search is performed to identify the best end point, at which the encoder stops all AI transmission and the decoder is left to fill the remaining holes on its own via Criminisi's algorithm in the default "easy-to-hard" order. At each candidate end point, the RD cost including both the AI-assisted patches and the decoder's self-discovered patches is computed, and the candidate point with the smallest RD cost is selected as the optimal end point. In practice, an "end-of-transmission" flag is coded to signal to the decoder the end of AI information and the start of the classical Criminisi's algorithm to fill in the remaining holes.

VII. Experimentation

A. Experimental Setup

To test the performance of our proposed encoder-driven inpainting strategy for inter-view predictive coding, we perform extensive experiments to compare our method against 3D-HEVC reference software HTM-13.0. The view synthesis software used is the VSRS-1D-Fast algorithm included in HTM-13.0.

In our codec, independent views are coded using 3D-HEVC. We only apply our strategy to code P-frames in the dependent views where inter-view prediction is dominant. The remaining B-frames in the dependent views are also coded using 3D-HEVC. Note, however, that the depth maps of all frames (P- and B-frames) in the dependent views are not explicitly coded but are reconstructed using our proposed procedure described in Section III. This amounts to a further bitrate reduction compared to 3D-HEVC. We use the same view synthesis algorithm as in 3D-HEVC. More implementation details and complexity analysis of our codec are discussed in Section VII-C.

Our tests generally follow the Common Test Conditions (CTC) [59]. We mainly use two JCT-3V standard test sequences [59], namely `Undo_Dancer` and `GT_Fly` from Nokia at 1920×1080-pixel resolution. Since they are synthetic multiview video sequences with accurate depth values, we can obtain satisfactory view synthesis quality via DIBR. In addition, we show the performance of two other JCT-3V standard test sequences, namely `Balloons` and `Kendo` from Nagoya University at 1024×768-pixel resolution. They are natural multiview sequences with imperfect depth maps, which penalizes the quality of the view synthesis via DIBR.

Views 1, 5, and 9 for `Undo_Dancer` and `GT_Fly` are tested. Each view has 250 frames with a frame rate at 25 frames per second (fps). As for `Balloons` and `Kendo`, view 1, 3, and 5 are used. There are 300 frames with a frame rate at 30 fps. The coding order of views is center-left-right. The center view is independently coded, and the left and right views are predicted by the center view. The size of group of picture (GOP) is 8. The period of I-frame for independent view and inter-view-only P-frame for dependent views is 24.

As the reconstruction quality using our proposal is bounded by the synthesized view quality via DIBR, our experimental results are under 36 dB for synthetic sequences and 31 dB for natural sequences on average in Peak-Signal-to-Noise-Ratio (PSNR). Hence, in addition to the default QP set (40, 35, 30, 25) for independent views, QP values 51, 48, 46, 43, and 37 are used for coding of the dependent views to match the DIBR-synthesized image quality stemming from the coded independent views. In general, to code dependent views both 3D-HEVC and our scheme use higher QP than the independent views. The QP pair for each RD point is specified in the following figures, in form of (QP_i, QP_d), where QP_i is for independent views and QP_d is for dependent views.

In our proposal, we try several λ's for RD optimization and QP's for our AI "intra" mode such that the operational RD convex hulls are found. Additionally, lossless network transmission is assumed in the following experiments.

The patch size in our proposed strategy is chosen to be 8×8. In theory we could also use variable patch sizes and the quadtree structure to further improve performance, but in this paper we fix the patch size to better evaluate the performance of encoder-driven inpainting. Accordingly, we fix the size of coding tree unit (CTU) in the reference 3D-HEVC to be 16×16, which is the smallest option of CTU and the closest to the patch size in our scheme. Since the 16×16 CTU in 3D-HEVC includes the 8×8 coding unit as one of its candidate modes, one might expect that the 3D-HEVC with 16×16 CTU would have better performance than our scheme with the fixed 8×8 coding patch. However, we will show that our method can actually outperform 3D-HEVC with 16×16 CTU, thanks to a more thorough exploitation of spatial and inter-view redundancies by our scheme.

Note that for sequences `Undo_Dancer` and `GT_Fly`, at the low bitrate range where our scheme operates, the

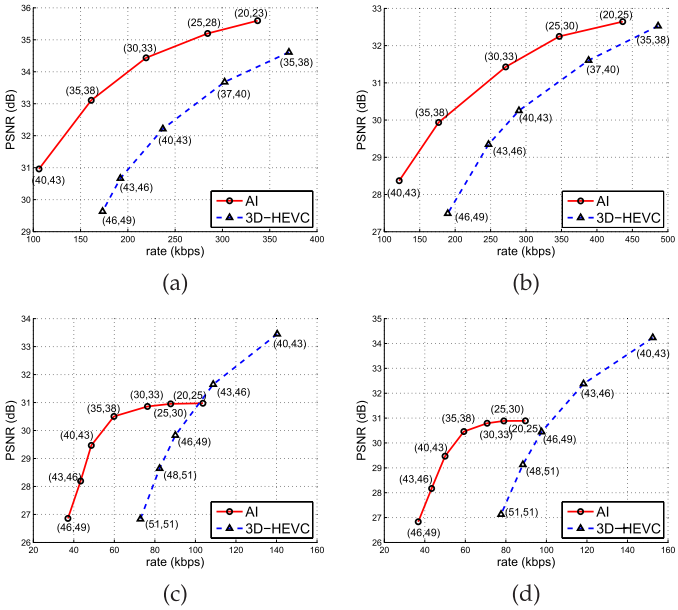


Fig. 6. Rate-distortion performance comparison between the proposed scheme (AI) and 3D-HEVC. The rate is the summation of that of two dependent views, and the distortion is the average PSNR of two dependent views. The BD gains are given in the parenthesis. (a) GT_Fly (2.7 dB). (b) Undo_Dancer (1.8 dB). (c) Balloons (1.6 dB). (d) Kendo (2.6 dB).

unrestricted 3D-HEVC (64 × 64 CTU) achieves about 3-7dB gain over the restricted 3D-HEVC with 16 × 16 CTU. We leave it as a future topic to generalize our scheme to other patch sizes, so that a direct comparison with the unrestricted 3D-HEVC could be eventually made possible.

We stress again that our proposed strategy is designed specifically for frames where temporal prediction is either not possible or not effective, and thus must rely solely on inter-view prediction for coding efficiency. As it will be demonstrated in our experiments, our new frame type in this case leads to noticeable performance gain at mid- to low-bitrate regions, and is useful as a complementary option in the situation where temporal coding is not effective.

B. Experimental Results

1) Comparison of Rate-Distortion Performance: We first compare the RD performance of the dependent views using our approach and the restricted 3D-HEVC with 16 × 16 CTU size. Here we use all the proposed coding tools, including the new transforms and "hard-to-easy" patch order. Some components will be examined individually later.

The RD performance for the two competing schemes is shown in Fig. 6 for the luma component of each sequence. The proposed method has about 3 dB gain due to the compactness of our coding strategy. Specifically, inter-view redundancy is exploited via DIBR; we do not code rendered regions whereas 3D-HEVC needs to code every block. The inter-patch redundancy is exploited via AI "skip" and "vec" while 3D-HEVC does not have efficient tools designed to exploit nonlocal image self-similarity. Further, the inter-pixel redundancy is exploited via image inpainting and novel transforms that are more suitable to our scenario than intra-prediction and DCT used in 3D-HEVC.

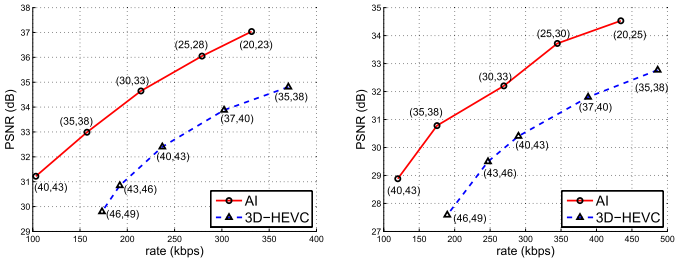


Fig. 7. Rate-distortion performance comparison including intermediate virtual views. The BD gains are given in the parenthesis. (a) GT_Fly (2.2 dB). (b) Undo_Dancer (1.7 dB).

In Fig. 6, the sequence GT_Fly has a larger gain than Undo_Dancer, because the faster movement of GT_Fly cameras (that greatly weakens temporal prediction) can be better compensated by our efficient inter-view coding. The second reason is that the disocclusion holes of GT_Fly are easier to be filled than Undo_Dancer because its background is smoother.

We observe that, for the two natural video sequences Balloons and Kendo, our scheme achieves noticeable coding gain compared to the restricted 3D-HEVC under 100 kbps. Multiview sequences with inaccurate depth maps will place a low upper-bound on the PSNR that our proposed frame type can achieve (*i.e.*, about 31 dB for Balloons and Kendo). This limits our performance, and as a result we can only achieve gain over 3D-HEVC at low bitrates. We believe that this is a transient problem as depth sensing technologies continue to improve, and therefore we will focus the following experiments on sequences GT_Fly and Undo_Dancer that have good quality depth maps.

The number of frames to which our scheme applies in each dependent view is 32 out of 250 frames for sequence GT_Fly and Undo_Dancer. The 32 frames account for up to 64% of the bitrates in 3D-HEVC, so focusing bitrate reduction for these frames is meaningful.

We next evaluate the quality of intermediate virtual views in Fig. 7. We synthesize three equally-spaced intermediate virtual views from the coded independent and dependent views. Similarly to the previous experiment, the dependent views are coded by our scheme and 3D-HEVC, respectively. The distortion is measured by the average PSNR of the dependent and intermediate views. As established in the CTC, the PSNR of an intermediate view is calculated against the synthesized view from the original color and depth map data of the reference views, not the original camera-captured views if available. The rate in Fig. 7 is that of dependent views.

Chroma components are usually smoother than luma components, but the general PSNR trends are similar. For sequence Undo_Dancer, our method can obtain up to 4 dB gain in PSNR, while for sequence GT_Fly, the gain can be up to 4.5 dB.

Compared to 3D-HEVC, our proposed scheme can achieve better visual quality at the same rate. A representative example is Fig. 8. Our scheme can reconstruct sharper boundaries between foreground (*e.g.*, the pillar) and background, as highlighted by the top ellipses in Fig. 8(c) and (d). As the

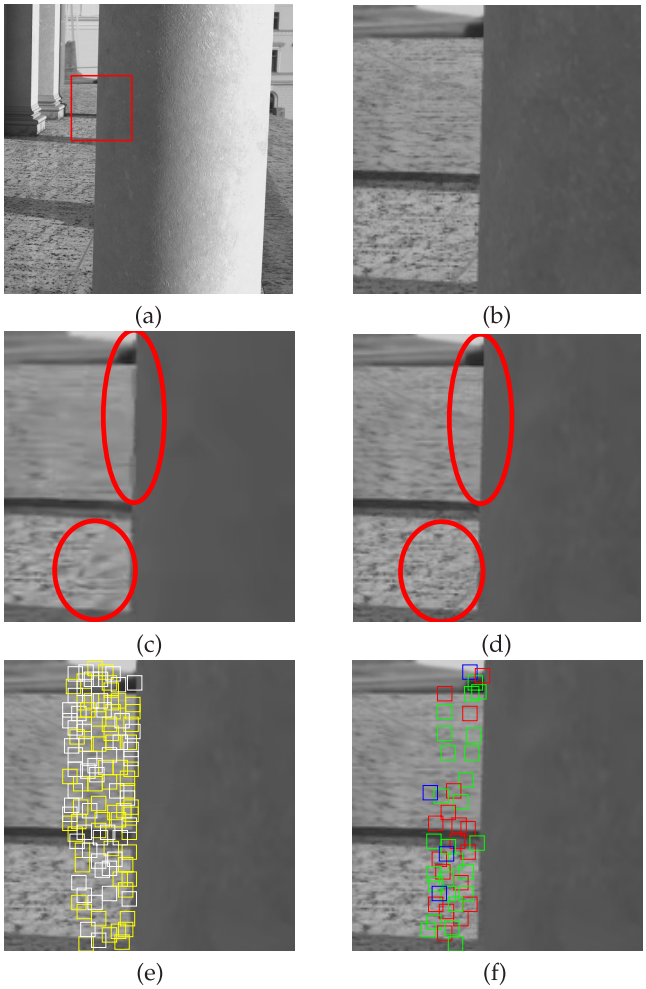


Fig. 8. An image segment of `Undo_Dancer` (view 1 frame 9). The number of bits to code the whole image is 33864 for 3D-HEVC and 31620 for the proposed method (denoted as "AI"). The coding blocks using different modes proposed in our method are shown in (e) and (f). Note that the regions that have no coding blocks marked are rendered via DIBR. (a) original `Undo_Dancer`. (b) zoom in the box in (a). (c) 3D-HEVC coded patch. (d) AI coded patch. (e) AI "skip" in white and AI "vec" in yellow. (f) AI "intra": DCT in red, GFT in blue, sDCT[8] in green.

rendered boundaries are not smoothed by the DCT-based coding in 3D-HEVC, and as the coding blocks along boundaries do not change the pixel values in the rendered region using most of our modes, the rendered boundaries are better preserved. One exception is the AI "intra" mode using regular DCT, but they are rarely applied to the boundaries as shown by red rectangles in Fig. 8(f). Secondly, our scheme can preserve more image details, as highlighted by the bottom ellipses in Fig. 8(c) and (d), because the bits saved by DIBR, AI "skip" and "vec" modes, which usually appear in smoother areas as shown in Fig. 8(e), are allocated to the blocks coding more complex regions by AI "intra".

2) Comparison of Inter-View Coding Performance: We now focus closely on the inter-view coding performance. We consider an application that requires a high degree of temporal random access, where the I-frame period of independent view is 8 instead of the default 24 in the CTC, and the frames in dependent views are only inter-view predicted by the

[8]sDCT is in short for the sparsification procedure using DCT.

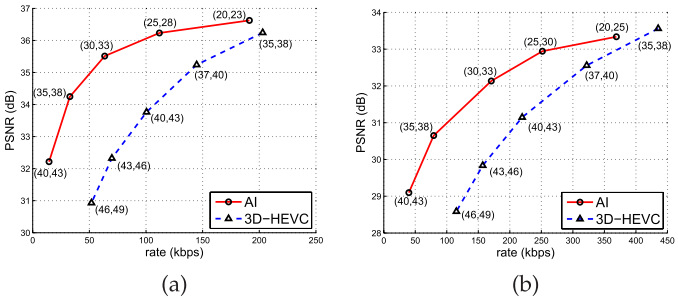


Fig. 9. Rate-distortion performance comparison of inter-view coded P-frames in dependent views. The BD gains are given in the parenthesis. (a) `GT_Fly` (3.1 dB). (b) `Undo_Dancer` (2.0 dB).

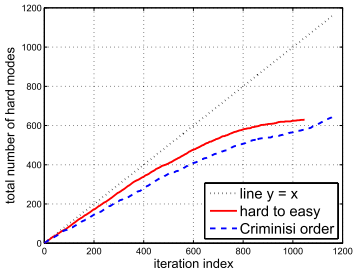


Fig. 10. Iterative accumulation of hard modes along the iteration of the coding scheme, for two different patch orders. Test on sequence `Undo_Dancer`.

I-frames in independent view at the same time instant. To avoid temporal prediction, we only measure the performance of the inter-view coded P-frames (in every 8 frames) in dependent views.[9] The results of our scheme and 3D-HEVC can be found in Fig. 9, showing that our inter-view prediction/coding tools are better than those in 3D-HEVC. In this experiment, the coding of sequence `GT_Fly` can take advantage of easy-to-inpaint holes after DIBR and leads to greater gains over 3D-HEVC than `Undo_Dancer`.

3) Comparison of Criminisi's "Easy-to-Hard" Order and the Proposed "Hard-to-Easy" Order: We next examine the effectiveness of our proposed "hard-to-easy" patch order. First, we show in Fig. 10 that the "hard-to-easy" order can be achieved by the proposed heuristic, which iteratively picks the patch that has the largest distortion with its best match in source region. We compare our patch order with Criminisi's "easy-to-hard" order. The x-axis of Fig. 10 indicates the iteration index. In each iteration, we fill a selected patch and then search the next patch to fill. The y-axis denotes the total number of hard modes selected before the current iteration. The hard modes include AI "vec" and "intra". The decoder cannot reconstruct the hard modes without any explicit instructions from the encoder. These curves depict the accumulation process of hard modes over iterations, comparing our patch order and Criminisi's order. Note that the diagonal line $y = x$ in the figure represents the situation where all modes up to current iteration are hard. The expected "hard-to-easy" behavior is to first select hard-to-fill patches (the early part of the curve should be close to $y = x$), and when most hard patches have been filled, the curve should grow as slow as possible due to the following easy-to-fill patches. In this sense,

[9]The rate should look smaller than the default setup in previous subsection.

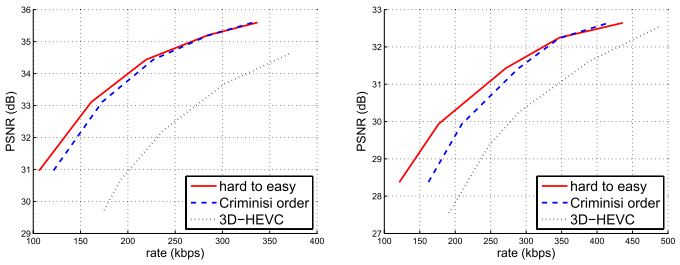


Fig. 11. Rate-distortion performance of our proposed coding strategy using two different patch orders. (a) GT_Fly. (b) Undo_Dancer.

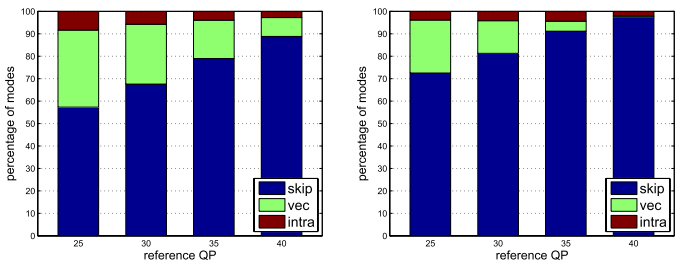


Fig. 12. Mode statistics for coding GT_Fly. Reference QP represents the one used for coding independent reference view. (a) Luma component. (b) Chroma component.

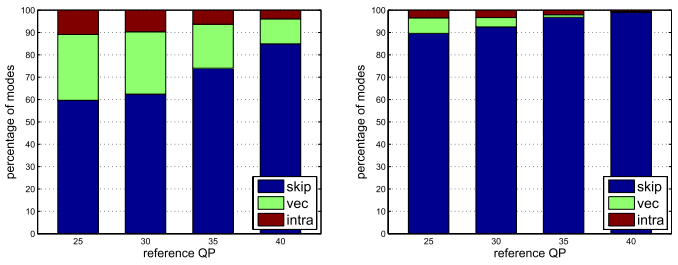


Fig. 13. Mode statistics for coding Undo_Dancer. Reference QP represents the one used for coding independent reference view. (a) Luma component. (b) Chroma component.

our heuristic for "hard-to-easy" order performs as expected. Recall that, using the proposed "hard-to-easy" patch order, we terminate the signaling of AI at an optimal RD point, which is determined by a binary search, as mentioned in Section VI. The remaining holes can be inpainted by the decoder, so that the number of iterations in our strategy are often less than the strategy using Criminisi's order as shown in Fig. 10.

The next experiment shows the performance improvement using our proposed "hard-to-easy" order. The setup is the same as that in Section VII-A, except for the competing scheme, which replaces the "hard-to-easy" order with Criminisi's order. As shown in Fig. 11, the proposed order outperforms Criminisi's order. Larger gains are observed at low rates, because more easy modes and fewer hard modes are chosen and the bitstream is likely to be truncated earlier.

Finally, we note that the arithmetic coding of the mode index can take advantage of the proposed "hard-to-easy" order: "hard" modes such as AI "intra" or "vec" are chosen more frequently at the beginning, while "easy" mode AI "skip" is likely chosen at the middle and end of the coding process. When we evaluate the probabilities of each mode over $W = 100$ last mode decisions, we observe a rate reduction of 15.51% (Undo_Dancer) for the coding of mode indices, compared to the arithmetic coding without a limited-length window whose probabilities are estimated by all previous mode decisions.

4) Evaluation of Graph Fourier Transform and Sparsification Procedure Using DCT: One key difference of our coding strategy from 3D-HEVC lies in the transform coding step. Instead of regular DCT, we propose GFT and a sparsification procedure using DCT (*i.e.*, sDCT) to take advantage of the particular feature of our coding patches, namely parts of patches are known and do not need to be coded. Given the default experimental setup used in Section VII-A, we compare the performance of AI "intra" mode using 1) DCT; 2) sDCT and DCT; 3) GFT and DCT; 4) all transforms (DCT, sDCT, and GFT). For Cases 2, 3, and 4, we pick the best transform in RD sense for each patch and signal the type of transform used to the decoder. Note that Case 4 is equivalent to the results in Fig. 6.

For sequence GT_Fly, Cases 2, 3, and 4 reduce 1.6%, 2.8%, and 3.9% BD rate [60], respectively, over Case 1. For sequence Undo_Dancer, Cases 2, 3, and 4 reduce 4.8%, 2.9%, and 6.2% BD rate, respectively, over Case 1. Since the proposed transforms are only for AI "intra", the coding gain is mostly at high rates, where more AI "intra" modes are selected. As shown in the next part, our coding strategy prefers low-cost AI "skip", and AI "vec" at low rates. Thus when the proportion of selected AI "intra" is already small, the performance improvement using our proposed transforms is limited. For sequence GT_Fly, the results shows that, using GFT can get more gain over the sparsification procedure, but the latter only needs to iteratively solve the least square problem at the encoder, while at decoder the regular inverse DCT is only performed once; hence it is less expensive in computation compared to GFT. The combination of sDCT and GFT is a better choice than using them individually.

When $\tau = 1$ and $\epsilon = 0.01 * Q_s$ (where Q_s is quantization step size), the average number of iterations of the re-weighted least square for sDCT is 2.0 for sequence Undo_Dancer.

5) Statistics of AI Modes: Each AI mode is proposed with its own purpose. To better understand why they are introduced and how they work, we show the statistics of AI modes in different circumstances. As shown in Fig. 12 and 13, we observe that 1) AI "skip" and "vec" dominate, meaning that there is a great amount of similarity to exploit from the known region and reference frame; 2) with larger QP, *i.e.*, lower rate budget, the number of the cheapest "skip" mode increases while the expensive "intra" and "vec" modes decrease; 3) the chroma component selects fewer "intra" mode due to simpler texture; 4) the proposed coding strategy can adapt to the feature of different sequences by choosing different sets of AI modes.

C. Implementation and Complexity Discussion

In our proposal, both the encoding and decoding of dependent views start from view synthesis via DIBR. In the experiments, our testing sequences are rectified and the disparities are found only along x-axis. The software package

"VSRS-1D-Fast" is employed to compute disparities for each image pixel based on coordinate conversions. Since there are no expensive operations, the rendering is much faster than coding the same image patches.

After obtaining an incomplete image via DIBR, we first fill in rounding holes. The complexity of detection is linear with the number of holes. The filling is a simple averaging of neighboring known pixels and performed only on the detected holes. Both encoder and decoder need to do the same detection and averaging.

We next fill the disocclusion holes patch-by-patch. In order to quantify the difficulty of inpainting individual patches, before the first iteration, the patches whose centers are on the boundary of the known and missing pixels are checked by template matching. Since we record the results of matching differences, we only need to perform additional template matching around the neighborhood of previously inpainted patch in each iteration. This process is also required to be performed at the encoder and decoder, because we do not explicitly encode and signal the patch orders to the decoder.

As described in Section IV, we have three AI modes as candidates to code a patch: AI "skip", AI "vec", and AI "intra". AI "skip" includes local and non-local inpainting. The local inpainting involves the solving of a linear system, whose complexity depends on the number of unknown pixels. The non-local inpainting performs template matching, whose complexity depends on the size of template and the search range, similar to template-based algorithms in the literature [8], [51]–[53]. In our proposal, the shape of template varies for each target patch and it is not larger than 8×8. The search range of template matching is 64×64. For simplicity of implementation, the matching has integer-pel accuracy, and full search is applied. Our scheme can be improved by fractional accuracies and fast search algorithms. AI "skip" only transmits the mode index to the decoder, which greatly reduces the transmission rate, but requires the decoder to perform the same operations as done at the encoder.

Similar to AI "skip", AI "vec" is also based on matching (integer-pel and 64×64 full search). Different from AI "skip", however, the decoder can directly use the received motion or similarity vectors to locate the matched patch in the search region, so the complexity is much lower than AI "skip".

AI "intra" has three transforms to choose from. The encoder will run all of them and select the one with minimum RD cost, and transmit the index of transform to the decoder, together with the coded residual. At the encoder, the sparsification procedure with DCT as introduced in Section V iterates several times to sparsify DCT coefficients and each iteration involves a matrix inverse. At the decoder, however, same as the regular inverse-DCT, only one pass (no iterations) is performed to reconstruct the prediction residual. GFT can achieve better sparsification, but to generate the adaptive transform matrix eigen-decomposition is required at both encoder and decoder. For the three transform options, CABAC in 3D-HEVC is used to code the quantized transform coefficients.

As discussed in Section VI, the patch selection and coding process iterate until there are no missing pixels left. Then, a binary search is performed to identify the best end point, at which the encoder stops all AI transmission and the decoder is left to fill the remaining holes on its own via Criminisi's algorithm in the default "easy-to-hard" order, where the major computation cost at the decoder is to perform template matching. To enable the binary search, we record the RD cost of each iteration. In each search point, we combine the recorded RD cost with the cost of inpainting the remaining holes. We empirically find that a coarser search on every eight iterations is good balance between RD performance and complexity.

According to the mode statistics shown in Fig. 12 and 13, AI "skip" is the dominant mode. Hence, the efficiency of our decoding can greatly benefit from an efficient implementation of template matching.

VIII. Conclusion

Compression of color and depth maps from multiple closely-spaced camera viewpoints is important for 3D imaging applications and new free viewpoint video communication. In this paper, we propose an encoder-driven inpainting strategy to complete disocclusion holes in the DIBR-synthesized image in an RD optimal manner. Missing pixel regions that are difficult-to-inpaint are first completed following instructions from the encoder in the form of auxiliary information (AI). The remaining easy-to-fill holes are then completed without encoder's help via nonlocal template matching, which is effective due to the self-similarity characteristics in natural images. Finally, we propose two patch-based transform coding techniques (graph Fourier transform and DCT sparsification), so that only missing pixels in a target patch are encoded, avoiding representation redundancy. In doing so, our coding strategy successfully exploits the three kinds of redundancy inherent in the color-plus-depth representation for coding gain: i) inter-view redundancy via DIBR-based 3D warping; ii) inter-pixel redundancy via patch-based transform coding; and iii) inter-patch redundancy via nonlocal template matching. Experimental results show that our proposed encoder-driven inpainting strategy is more effective than a restricted implementation of 3D-HEVC in RD performance. Inter-view, inter-patch, and inter-pixel redundancies are greatly reduced by efficient hole filling using the well-designed AI modes. Our proposed novel transforms boost RD performance at high rates, and the "hard-to-easy" patch order improves RD performance mainly at low rates such that our overall scheme can reap noticeable overall gains.

References

[1] M. Tanimoto, M. P. Tehrani, T. Fujii, and T. Yendo, "Free-viewpoint TV," *IEEE Signal Process. Mag.*, vol. 28, no. 1, pp. 67–76, Jan. 2011.

[2] S. B. Gokturk, H. Yalcin, and C. Bamji, "A time-of-flight depth sensor—System description, issues and solutions," in *Proc. Conf. Comput. Vis. Pattern Recognit. Workshop (CVPRW)*, Washington, DC, USA, Jun. 2004, p. 35.

[3] J. Sun, H.-Y. Shum, and N.-N. Zheng, "Stereo matching using belief propagation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 7, pp. 787–800, Jul. 2003.

[4] P. Merkle, A. Smolic, K. Müller, and T. Wiegand, "Multi-view video plus depth representation and coding," in *Proc. IEEE Int. Conf. Image Process.*, San Antonio, TX, USA, Sep./Oct. 2007, pp. I-201–I-204.

[5] D. Tian, P.-L. Lai, P. Lopez, and C. Gomila, "View synthesis techniques for 3D video," *Proc. SPIE*, vol. 7443, pp. 74430T-1–74430T-11, Sep. 2009.

[6] C. Zhang, Z. Yin, and D. Florencio, "Improving depth perception with motion parallax and its application in teleconferencing," in *Proc. IEEE Int. Workshop Multimedia Signal Process.*, Rio de Janeiro, Brazil, Oct. 2009, pp. 1–6.

[7] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, San Diego, CA, USA, Jun. 2005, pp. 60–65.

[8] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Trans. Image Process.*, vol. 13, no. 9, pp. 1200–1212, Sep. 2004.

[9] P. Merkle, A. Smolic, K. Müller, and T. Wiegand, "Efficient prediction structures for multiview video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 11, pp. 1461–1473, Nov. 2007.

[10] M. Flierl, A. Mavlankar, and B. Girod, "Motion and disparity compensated coding for multiview video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 11, pp. 1474–1484, Nov. 2007.

[11] S. Shimizu, M. Kitahara, H. Kimata, K. Kamikura, and Y. Yashima, "View scalable multiview video coding using 3D warping with depth map," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 11, pp. 1485–1495, Nov. 2007.

[12] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[13] G. J. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[14] G. Shen, W.-S. Kim, S. Narang, A. Ortega, J. Lee, and H. Wey, "Edge-adaptive transforms for efficient depth map coding," in *Proc. Picture Coding Symp.*, Nagoya, Japan, Dec. 2010, pp. 566–569.

[15] W. Hu, G. Cheung, X. Li, and O. Au, "Depth map compression using multi-resolution graph-based transform for depth-image-based rendering," in *Proc. IEEE Int. Conf. Image Process.*, Orlando, FL, USA, Sep./Oct. 2012, pp. 1297–1300.

[16] W. Hu, G. Cheung, A. Ortega, and O. Au, "Multiresolution graph Fourier transform for compression of piecewise smooth images," *IEEE Trans. Image Process.*, vol. 24, no. 1, pp. 419–433, Jan. 2015.

[17] W. Hu, G. Cheung, and A. Ortega, "Intra-prediction and generalized graph Fourier transform for image coding," *IEEE Signal Process. Lett.*, vol. 22, no. 11, pp. 1913–1917, Nov. 2015.

[18] G. Cheung, A. Kubota, and A. Ortega, "Sparse representation of depth maps for efficient transform coding," in *Proc. Picture Coding Symp.*, Nagoya, Japan, Dec. 2010, pp. 298–301.

[19] G. Cheung, J. Ishida, A. Kubota, and A. Ortega, "Transform domain sparsification of depth maps using iterative quadratic programming," in *Proc. 18th IEEE Int. Conf. Image Process.*, Brussels, Belgium, Sep. 2011, pp. 129–132.

[20] L. Yan, Z. Zhaoyang, and A. Ping, "Stereo video coding based on frame estimation and interpolation," *IEEE Trans. Broadcast.*, vol. 49, no. 1, pp. 14–21, Mar. 2003.

[21] K. Hideaki, K. Masaki, K. Kazuto, and Y. Yoshiyuki, "Multiview video coding using reference picture selection for free-viewpoint video communication," in *Proc. Picture Coding Symp.* 2004, pp. 15–17.

[22] K. Müller *et al.*, "Multi-view video coding based on H.264/AVC using hierarchical B-frames," in *Proc. 25th Picture Coding Symp.*, Beijing, China, Apr. 2006, pp. 385–390.

[23] H. S. Hussein, M. El-Khamy, and M. El-Sharkawy, "Blind configuration of multi-view video coder prediction structure," *IEEE Trans. Consum. Electron.*, vol. 59, no. 1, pp. 191–199, Feb. 2013.

[24] E. Martinian, A. Behrens, J. Xin, and A. Vetro, "View synthesis for multiview video compression," in *Proc. Picture Coding Symp.*, 2006, pp. 1–8.

[25] Y. Morvan, D. Farin, and P. H. N. de With, "Multiview depth-image compression using an extended H.264 encoder," in *Proc. 9th Int. Conf. Adv. Concepts Intell. Vis. Syst. (ACIVS)*, 2007, pp. 675–686.

[26] S. Yea and A. Vetro, "View synthesis prediction for multiview video coding," *Signal Process., Image Commun.*, vol. 24, nos. 1–2, pp. 89–100, Jan. 2009.

[27] K. Müller, A. Smolic, K. Dix, P. Merkle, P. Kauff, and T. Wiegand, "Reliability-based generation and view synthesis in layered depth video," in *Proc. IEEE 10th Workshop Multimedia Signal Process.*, Oct. 2008, pp. 34–39.

[28] W.-S. Kim, A. Ortega, P. Lai, D. Tian, and C. Gomila, "Depth map coding with distortion estimation of rendered view," *Proc. SPIE*, vol. 7543, p. 75430B, Jan. 2010.

[29] G. Cheung, W.-S. Kim, A. Ortega, J. Ishida, and A. Kubota, "Depth map coding using graph based transform and transform domain sparsification," in *Proc. IEEE 13th Int. Workshop Multimedia Signal Process.*, Hangzhou, China, Oct. 2011, pp. 1–6.

[30] Y. Yuan, G. Cheung, P. Frossard, P. L. Callet, and V. H. Zhao, "Piecewise smooth depth image approximate & coding for virtual view synthesis," in *Proc. IEEE Int. Workshop Multimedia Signal Process.*, Xiamen, China, Oct. 2015.

[31] K. Müller *et al.*, "3D high-efficiency video coding for multi-view video and depth data," *IEEE Trans. Image Process.*, vol. 22, no. 9, pp. 3366–3378, Sep. 2013.

[32] G. J. Sullivan, J. M. Boyce, Y. Chen, J.-R. Ohm, C. A. Segall, and A. Vetro, "Standardized extensions of high efficient video coding (HEVC)," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 6, pp. 1001–1016, Dec. 2013.

[33] Y. Chen, Y.-K. Wang, K. Ugur, M. M. Hannuksela, J. Lainema, and M. Gabbouj, "The emerging MVC standard for 3D video services," *EURASIP J. Adv. Signal Process.*, vol. 2009, Jan. 2009, Art. ID 8.

[34] D. Liu, X. Sun, F. Wu, S. Li, and Y.-Q. Zhang, "Image compression with edge-based inpainting," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 10, pp. 1273–1287, Oct. 2007.

[35] D. Liu, X. Sun, and F. Wu, "Intra prediction via edge-based inpainting," in *Proc. Data Compress. Conf.*, Snowbird, UT, USA, Mar. 2008, pp. 282–291.

[36] Y. Xu and H. Xiong, "Advanced inpainting-based macroblock prediction with regularized structure propagation in video compression," in *Proc. Picture Coding Symp.*, Nagoya, Japan, Dec. 2010, pp. 94–97.

[37] Z. Xiong, X. Sun, and F. Wu, "Block-based image compression with parameter-assistant inpainting," *IEEE Trans. Image Process.*, vol. 19, no. 6, pp. 1651–1657, Jun. 2010.

[38] X. Qi, T. Zhang, F. Ye, A. Men, and B. Yang, "Intra prediction with enhanced inpainting method and vector predictor for HEVC," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Kyoto, Japan, Mar. 2012, pp. 1217–1220.

[39] J. Cheng, F. Ye, J. Di, C. Liu, and A. Men, "Depth map compression via edge-based inpainting," in *Proc. Picture Coding Symp.*, Krakow, Poland, May 2012, pp. 57–60.

[40] I. Daribo, G. Cheung, T. Maugey, and P. Frossard, "R-D optimized auxiliary information for inpainting-based view synthesis," in *Proc. 3DTV-Conf.*, Zurich, Switzerland, Oct. 2012, pp. 1–4.

[41] J. Balle and M. Wien, "Extended texture prediction for H.264/AVC intra coding," in *Proc. IEEE Int. Conf. Image Process.*, Sep./Oct. 2007, pp. VI-93–VI-96.

[42] L.-Y. Wei and M. Levoy, "Fast texture synthesis using tree-structured vector quantization," in *Proc. 27th Annu. Conf. Comput. Graph. Interact. Techn. (SIGGRAPH)*, Jul. 2000, pp. 479–488.

[43] T. K. Tan, C. S. Boon, and Y. Suzuki, "Intra prediction by template matching," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 2006, pp. 1693–1696.

[44] T. K. Tan, C. S. Boon, and Y. Suzuki, "Intra prediction by averaged template matching predictors," in *Proc. 4th IEEE Consum. Commun. Netw. Conf.*, Jan. 2007, pp. 405–409.

[45] R. I. Hartley, "Theory and practice of projective rectification," *Int. J. Comput. Vis.*, vol. 35, no. 2, pp. 115–127, 1999.

[46] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proc. SIGGRAPH*, New Orleans, LA, USA, 2000, pp. 417–424.

[47] T. F. Chan and J. Shen, "Nontexture inpainting by curvature-driven diffusions," *J. Vis. Commun. Image Represent.*, vol. 12, no. 4, pp. 436–449, 2001.

[48] M. Bertalmio, A. L. Bertozzi, and G. Sapiro, "Navier-stokes, fluid dynamics, and image and video inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1. 2001, pp. I-355–I-362.

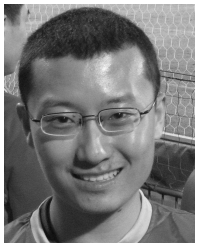
[49] K. A. Patwardhan, G. Sapiro, and M. Bertalmío, "Video inpainting under constrained camera motion," *IEEE Trans. Image Process.*, vol. 16, no. 2, pp. 545–553, Feb. 2007.

[50] P. Ndjiki-Nya *et al.*, "Depth image-based rendering with advanced texture synthesis for 3D video," *IEEE Trans. Multimedia*, vol. 13, no. 3, pp. 453–465, Jun. 2011.

[51] I. Daribo and B. Pesquet-Popescu, "Depth-aided image inpainting for novel view synthesis," in *Proc. IEEE Multimedia Signal Process. Workshop*, Saint-Malo, France, Oct. 2010, pp. 167–170.

[52] S. Reel, G. Cheung, P. Wong, and L. S. Dooley, "Joint texture-depth pixel inpainting of disocclusion holes in virtual view synthesis," in *Proc. APSIPA Annu. Summit Conf.*, Kaohsiung, Taiwan, Oct./Nov. 2013, pp. 1–7.

[53] S. Reel, K. C. P. Wong, G. Cheung, and L. S. Dooley, "Disocclusion hole-filling in DIBR-synthesized images using multi-scale template matching," in *Proc. IEEE Vis. Commun. Image Process.*, Valletta, Malta, Dec. 2014, pp. 494–497.
[54] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, May 2013.
[55] Z. He, Y. K. Kim, and S. Mitra, "Low-delay rate control for DCT video coding via ρ-domain source modeling," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 8, pp. 928–940, Aug. 2002.
[56] I. Daubechies, R. DeVore, M. Fornasier, and C. S. Güntürk, "Iteratively reweighted least squares minimization for sparse recovery," *Commun. Pure Appl. Math*, vol. 63, no. 1, pp. 1–38, 2010.
[57] K. Ramchandran, A. Ortega, and M. Vetterli, "Bit allocation for dependent quantization with applications to multiresolution and MPEG video coders," *IEEE Trans. Image Process.*, vol. 3, no. 5, pp. 533–545, Sep. 1994.
[58] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco, CA, USA: Freeman, 1999.
[59] K. Müller and A. Vetro, *Common Test Conditions of 3DV Core Experiments*, document JCT3V-G1100, San Jose, CA, USA, 2014.
[60] G. Bjøntegaard, *Calculation Average PSNR Differences Between RD-Curves*, document VCEG-M33, Mar. 2001.



Yu Gao (S'09–M'15) received the B.E. and M.E. degrees from the Beijing University of Technology, China, and the Ph.D. degree from Simon Fraser University, Burnaby, Canada, in 2005, 2008, and 2014, respectively. From 2013 to 2014, he was an Intern with the National Institute of Informatics, Tokyo, Japan. Since 2015, he has been with Evertz Microsystem, Burlington, Canada, where he is currently an Algorithm Design Engineer.

Gene Cheung (M'00–SM'07) received the B.S. degree in electrical engineering from Cornell University, in 1995, and the M.S. and Ph.D. degrees in electrical engineering and computer science from the University of California at Berkeley, in 1998 and 2000, respectively.

He was a Senior Researcher with Hewlett-Packard Laboratories Japan, Tokyo, Japan, from 2000 to 2009. He is currently an Associate Professor with the National Institute of Informatics, Tokyo. He is an Adjunct Associate Professor with the Hong Kong University of Science and Technology.

His research interests include image and video representation, immersive visual communication, and graph signal processing. He served as an Associate Editor of the IEEE TRANSACTIONS ON MULTIMEDIA (2007–2011) and DSP Applications Column in the *IEEE Signal Processing Magazine* (2010–2014). He currently serves as an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING, *SPIE Journal of Electronic Imaging* (2014-present), and the *APSIPA Journal on Signal & Information Processing* (2011-present), and an Area Editor of *EURASIP Signal Processing: Image Communication* (2011-present). He served as a member of the Multimedia Signal Processing Technical Committee in the IEEE Signal Processing Society (2012–2014), and a member of the Image, Video, and Multidimensional Signal Processing Technical Committee (2015–2017). He also served as the Technical Program Co-Chair of the International Packet Video Workshop 2010 and the IEEE International Workshop on Multimedia Signal Processing 2015, and the Symposium Co-Chair of the CSSMA Symposium in the IEEE GLOBECOM 2012. He is a co-author of the Best Student Paper Award in the IEEE Workshop on Streaming and Media Communications 2011 [in conjunction with the IEEE International Conference on Multimedia and Expo (ICME) 2011], best paper finalists in ICME 2011, the IEEE International Conference on Image Processing 2011, and ICME 2015, the Best Paper Runner-Up Award in ICME 2012, and the Best Student Paper Award in ICIP 2013.

Thomas Maugey (S'09–M'11) received the degree from the École Supérieure d'Electricité, Supélec, Gif-sur-Yvette, France, in 2007, the M.Sc. degree in fundamental and applied mathematics from Suplec and Universit Paul Verlaine, Metz, France, in 2007, and the Ph.D. degree in image and signal processing from TELECOM ParisTech, Paris, France, in 2010. From 2010 to 2014, he was a Post-Doctoral Researcher with the Signal Processing Laboratory, Swiss Federal Institute of Technology, Lausanne, Switzerland. He is a Research Scientist with INRIA in the team-project SIROCCO, Rennes, France. His research interests include monoview and multiview distributed video coding, 3D video communication, data representation, video compression, network coding, and view synthesis.

Pascal Frossard (S'96–M'01–SM'04) received the M.S. and Ph.D. degrees in electrical engineering from the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, in 1997 and 2000, respectively. From 2001 and 2003, he was a member of the Research Staff with the IBM T. J. Watson Research Center, Yorktown Heights, NY, where he worked on media coding and streaming technologies. Since 2003, he has been a Faculty Member with EPFL, where he heads the Signal Processing Laboratory. His research interests include graph signal processing, image representation and coding, visual information analysis, and distributed signal processing and communications.

Dr. Frossard has been the General Chair of the IEEE ICME 2002 and Packet Video 2007. He has been the Technical Program Chair of the IEEE ICIP 2014 and EUSIPCO 2008, and a member of the organizing or technical program committees of numerous conferences. He has been an Associate Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING (2015-), the IEEE TRANSACTIONS ON BIG DATA (2015-), the IEEE TRANSACTIONS ON IMAGE PROCESSING (2010–2013), the IEEE TRANSACTIONS ON MULTIMEDIA (2004–2012), and the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (2006–2011). He is the Chair of the IEEE Image, Video and Multidimensional Signal Processing Technical Committee (2014–2015), and an Elected Member of the IEEE Visual Signal Processing and Communications Technical Committee (2006-) and the IEEE Multimedia Systems and Applications Technical Committee (2005-). He has served as the Steering Committee Chair (2012–2014) and the Vice Chair (2004–2006) of the IEEE Multimedia Communications Technical Committee and a member of the IEEE Multimedia Signal Processing Technical Committee (2004–2007). He received the Swiss NSF Professorship Award in 2003, the IBM Faculty Award in 2005, the IBM Exploratory Stream Analytics Innovation Award in 2008, and the IEEE TRANSACTIONS ON MULTIMEDIA BEST PAPER AWARD in 2011.

Jie Liang (S'99–M'04–SM'11) received the B.E. and M.E. degrees from Xi'an Jiaotong University, China, in 1992 and 1995, respectivley, the M.E. degree from the National University of Singapore (NUS), in 1998, and the Ph.D. degree from Johns Hopkins University, Baltimore, MD, USA, in 2003. Since 2004, he has been with the School of Engineering Science, Simon Fraser University, Canada, where he is currently a Professor and the Associate Director. In 2012, he visited the University of Erlangen-Nuremberg, Germany, as an Alexander von Humboldt Research Fellow. From 2003 to 2004, he was with the Video Codec Group, Microsoft Digital Media Division. From 1997 to 1999, he was with Hewlett-Packard Singapore and the Center for Wireless Communications (now part of the Institute for Infocomm Research), NUS.

His research interests include image and video coding, multimedia communications, sparse signal processing, computer vision, and machine learning. He is currently an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, the IEEE SIGNAL PROCESSING LETTERS, *Signal Processing: Image Communication*, and the *EURASIP Journal on Image and Video Processing*. He is a member of the IEEE Multimedia Systems and Applications Technical Committee and Multimedia Signal Processing Technical Committee, and is a Professional Engineer in British Columbia. He received the 2014 IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY Best Associate Editor Award, the 2014 SFU Dean of Graduate Studies Award for Excellence in Leadership, and the 2015 Canada NSERC Discovery Accelerator Supplements Award.