Compression of Multiview Images using a Sparse Layer-based Representation

by
Andriy Gelman

A Thesis submitted in fulfilment of requirements for the degree of
Doctor of Philosophy of Imperial College London

Communications & Signal Processing Group
Department of Electrical & Electronic Engineering
Imperial College London
2012

Statement of Originality

I declare that the intellectual content of this thesis is the product of my own research work under the guidance of my thesis advisor, Dr. Pier Luigi Dragotti. Any ideas or quotations from the work of other people, published or otherwise, are fully acknowledged in accordance with the standard referencing practices of the discipline. The material of this thesis has not been submitted for any degree at any other academic or professional institution.

Abstract

Multiview images are obtained by recording a scene from different viewpoints. The additional information can be used to improve the performance of various applications ranging from e-commerce to security surveillance. Many such applications process large arrays of images, and therefore it is important to consider how the information is stored and transmitted.

In this thesis we address the issue of multiview image compression. Our approach is based on the concept that a point in a 3D space maps to a constant intensity line in specific multiview image arrays. We use this property to develop a sparse representation of multiview images. To obtain the representation we segment the data into layers, where each layer is defined by an object located at a constant depth in the scene. We extract the layers by initialising the layer contours and then by iteratively evolving them in the direction which minimises an appropriate cost function. To obtain the sparse representation we reduce the redundancy of each layer by using a multi-dimensional discrete wavelet transform (DWT). We apply the DWT in a separable approach; first across the camera viewpoint dimensions, followed by a 2D DWT applied to the spatial dimensions. The camera viewpoint DWT is modified to take into account the structure of each layer, and also the occluded regions.

Based on the sparse representation, we propose two compression algorithms. The first is a centralised approach, which achieves a high compression, however requires the transmission of all the data. The second is an interactive method, which trades-off compression performance in order to facilitate random access to the multiview image dataset. In addition, we address the issue of rate allocation between encoding of the layer contours and the texture. We demonstrate that the proposed centralised and interactive methods outperform H.264/MVC and JPEG 2000, respectively.

Acknowledgement

First, I would like to express my thanks to my supervisor Pier Luigi Dragotti. I am grateful for the constant support, guidance and advice I received throughout my PhD. It has been a pleasure to work with such an enthusiastic researcher. Also, I would like to say thank you to my second supervisor Vladan Velisavljevic. I am truly grateful for having received support from two researchers who were always there for me when I needed guidance.

I want to thank my family for the constant support. My mother Svetlana - I'm extremely proud to be your son, my sister Julia - you always cheer me up when I need you most, and my father Victor - my engineering influence.

I also want to say thank you to my close friends Bes and Toni. This PhD would not have been the same without you.

ב

Contents

List of Figures

List of Tables

Abbreviations

AWGN:	Additive White Gaussian Noise
bpp:	bits per pixel
bop:	bits per object pixel
DCP:	Disparity compensated prediction
DSC:	Distributed Source Coding
DWT:	Discrete Wavelet Transform
EPI:	Epipolar-plane image
FTV:	Free Viewpoint TV
GOP:	Group of pictures
HVS:	Human Visual System
IBR:	Image-based rendering
LSB:	Least significant bit
MSB:	Most significant bit
MSE:	Mean squared error
MVC:	Multiview Coding
NLA:	Nonlinear approximation
SP:	Switching Predictive
SSD:	Sum of squared differences
SURE:	Stein's Unbiased Risk Estimate
PDE:	Partial Differential Equation
PSNR:	Peak signal to noise ratio
RGB:	Red Green Blue
RD:	Rate-distortion

List of Symbols

α:	Global α-Lipschitz regularity of a 2D signal
c_i:	RD scaling constant of an α-Lipschitz signal coded in the independent setup
$\hat{c_i}$:	RD scaling constant of an α-Lipschitz signal coded in the DSC setup
C_j:	Scaling factor used to model occlusions
Δp:	Layer's disparity gradient
d_j :	Transform coefficients in the j-th scale of a DWT
d_ϵ^j:	Wavelet coefficients in the α-Lipschitz error term
$D\left(\mathbf{R_s}, \mathbf{R_x}\right)$:	Joint rate-distortion function in the centralised coding scheme
$\mathbf{D_{DSC}}\left(\mathbf{R_t}\right)$:	Total distortion in the DSC setup
$\mathbf{D_{ind}}\left(\mathbf{R_t}\right)$:	Total distortion in the independent setup
$D_s\left(\mathbf{R_s}\right)$:	Upper bound of the distortion due to coding the layer contours
$D_x\left(\mathbf{R_x}\right)$:	Distortion due to encoding the texture
$\gamma\left(s\right)$:	2D layer contour defined on one image viewpoint
Γ:	4D/3D layer contour defined using $\gamma\left(s\right)$ and Δp
$\mathcal{H}$:	Layer defined by boundary Γ
$\mathcal{H}^{\mathcal{V}}$:	Visible regions of layer $\mathcal{H}$
K_j:	Number of subbands in the j-th layer
L:	Number of the layers in the dataset
$\mathcal{L}_e\left[n\right]$:	Low-pass subband following the inter-view DWT
$\mathcal{L}_o\left[n\right]$:	High-pass subband following the inter-view DWT

M:	Number of images in the dataset
N_{ij}:	Number of transform coefficients in the i-th subband of the j-th layer
$P_o[n]$:	Images located at odd camera viewpoint locations
$P_e[n]$:	Images located at even camera viewpoint locations
$\mathbf{R_s}$:	Bit budget allocated to encoding the layer contours
$\mathbf{R_x}$:	Bit budget allocated to encoding the texture
$\mathbf{R_t}$:	Total bit budget
σ_{ij}:	Standard deviation of the transform coefficients in the i-th subband of the j-th layer
T_j:	Size of a bounding box in the j-th layer
V_j:	Number of vertices in the j-th layer
V_x:	Camera viewpoint location (x-axis)
V_y:	Camera viewpoint location (y-axis)
$\mathcal{W}$:	Warping operator
x:	Image spatial coordinate (x-axis)
y:	Image spatial coordinate (y-axis)
ζ_j:	Maximal value of the texture in the j-th layer

Chapter 1

Introduction

1.1 Motivation

Multiview images are obtained by recording a scene from different viewpoints using an array of cameras. These datasets have become an important component in a wide range of signal processing applications. In the computer graphics community, multiview images are used to create photorealistic results with a low computational complexity. The reason for using real datasets, as opposed to an accurate 3D representation of a scene, stems from the fact that natural images contain many subtle properties which are difficult to model and reproduce. Yet these properties are necessary to create a 'realistic' perception in the rendered scene [44]. The process of creating virtual views from images is known as image-based rendering (IBR) [81].

In particular, IBR has been extensively researched due to its applications in free viewpoint TV (FTV) [84] and 3D TV [49]. The latter creates a perception of depth, whereas FTV allows the user to perceive an immersive experience by interactively choosing their viewpoint. In FTV, IBR plays an important role; it is used to synthesise novel viewpoints where no camera exists. Other examples of applications where multiview images are commonly used include: object and feature recognition, security surveillance, teleconferencing and remote education [49].

These applications have to process significantly more data than the traditional single view setup. For example, in IBR a popular approach known as *light field* rendering [52] can consist of 1000 images with an uncompressed data size of ∼1GB [96]. Therefore, in order to make these applications practical, it is important to develop methods which efficiently compress this type of data.

Multiview images are highly redundant in that they contain very similar content in each view. This property is due to the fact that in a multiview image array the cameras are commonly very closely spaced. As a consequence, a given object in one of the images will also appear shifted in each of the neighbouring images. Moreover, the shift can often be predicted and partially depends on the geometry of the array. These properties can be taken into account when developing coding methods to achieve a high compression.

When developing a multiview compression method, it is important to consider the type of application the algorithm is designed for. If the data is encoded offline and stored on a hard disk, we can design a complex encoding method with a high compression efficiency. However, in an interactive communication system [21], we must consider other aspects in addition to this property. An interactive communication setup consists of a server and remote clients. The remote clients connect to the server and request certain images from the dataset. An advantage of this method is that only the requested views are transmitted as opposed to the complete dataset. In this setup, the ability to transmit certain images without decoding the dataset is significantly more important than compression efficiency. This property is known as random access. In addition, other factors such as decoding complexity and scalability should be taken into account. Scalability is a general term which defines whether a single bit stream can be decoded multiple times according to different distortion or resolution levels (spatial or temporal) [64].

In compression, we must also carefully consider how the coding parameters are chosen to optimise the method. Typically, the parameters are selected using a rate-distortion (RD) formulation where the goal is to minimise the distortion for a given bit budget (complexity and other factors can also be taken into account [4]). For example in subband coding [89] the goal is to select a set of quantisers which minimise the distortion for a given rate constraint. The number of free parameters in a coding setup is typically large and optimisation techniques must be employed to solve these problems. A popular framework applied in image and video coding is Lagrangian minimisation [83]. Here, the constrained optimisation problem is transformed into an unconstrained one by jointly minimising the rate and distortion. This framework can be applied in an operational sense by encoding the data points or by modelling the RD using a certain class of signals, such as a Gaussian random process [25] or piecewise polynomial signals [80].

1.2 Problem Statement

In this thesis, we address the issue of multiview image compression. The input dataset is a 2D array of images. We assume that the cameras are evenly spaced on a 2D grid, and that the viewing direction of the cameras is perpendicular to the location plane. We also consider a calibrated setup, such that the camera locations are known. In addition, we assume that the scene is stationary; this allows us to capture a dataset using a conventional camera by changing its viewpoint to different locations on the array. Without a loss of generality, we encode only the monochromatic component of the images.

The coding schemes that we propose are based on a sparse representation of multiview images. A sparse representation is a decomposition, where the original signal can be approximated well using a small number of transform coefficients [12]. The representation that we develop is unique to multiview images. It is based on analysing how the data in multiview images is structured, and it is related to the idea that multiview images are highly redundant.Two of the underlying assumptions which are necessary to obtain the sparse representation are that the scene is Lambertian and that it can be analysed as a set of constant depth planes. In a Lambertian scene the luminance of an object (light ray intensity) appears the same when it is observed from different viewpoints.

Based on the sparse representation we propose two coding schemes. The first is a centralised approach and it achieves a high compression performance in comparison to the state-of-the-art. The aim of this method is to minimise the distortion for a given bit budget constraint, where the distortion is measured in terms of the mean squared error (MSE). MSE is a conventional metric for multiview images, and has the desirable convexity and differentiability properties [91]. We optimise the method in the RD sense to ensure an efficient performance. In addition, we show that for certain classes of signals there exist closed-form solutions for the rate allocation, which minimises the distortion.

However, the joint encoding method requires that all of the data is transmitted to the user. As outlined in the introduction, this type of coding scheme is not appropriate for certain applications, such as an interactive communication system. We address this issue in the second algorithm, by showing that we can trade-off compression performance with random access capabilities.

1.3 Thesis Outline

The outline of this thesis is as follows. In Chapter 2 we overview IBR and show that multiview images can be characterised using a single function known as the *plenoptic function* [3]. The main aim of this chapter is to show that multiview images are structured and redundant. Specifically, we show that an object located at a constant depth in the scene, shifts by the same number of pixels in each frame (under some conditions). Based on this analysis, we present the layer-based representation. We show that multiview images can be analysed as a set of layers, where each layer is related to an object in the scene. This chapter is concluded by presenting a number of multiview image compression methods.

In Chapter 3, we outline our approach to obtain a sparse decomposition of multiview images. The method consists of two main stages. First, we segment the multiview image array into a set of layers. To extract the layers, we pose the segmentation as an optimisation problem. The segmentation of each layer is initialised and then iteratively evolved using the level-set method [78]. In order to obtain a sparse representation, we remove the redundancy in the second stage of the method by using a multi-dimensional discrete wavelet transform (DWT). We apply the transform in a separable approach across the camera viewpoint dimensions, followed by the spatial (image) dimensions. We evaluate the sparsity of the method, and also show that the representation can be used to denoise multiview images.

In Chapter 4, we present the joint compression method based on the proposed sparse decomposition. To encode multiview images we transmit the texture and also the contours of the layer-based representation. We present our approach to encode both these features. This chapter also deals with the RD optimisation of the proposed method. We show how the total bit budget can be correctly distributed between encoding the texture and the layer contours. To solve this problem, we derive the RD behaviour by appropriately modelling the data. We model the transform coefficients as Gaussian random variables and the contours as piecewise linear signals. The RD function is then minimised given a rate constraint using the Lagrangian multiplier optimisation method.

In Chapter 5, we evaluate the proposed centralised compression algorithm. We show that our approach outperforms the state-of-the-art H.264/AVC [92] coding scheme. We use a number of real datasets with varying number of images, spatial resolution and scene complexity. In addition, we evaluate the proposed RD model and also the rate allocation strategy.

In Chapter 6, we show that the proposed joint coding algorithm can be modified to support random access by trading off compression efficiency. We call it an interactive compression method, as it is suitable for an interactive communication setup. In addition, we derive a RD model for the proposed method by assuming that the input images are globally smooth 2D α-Lipschitz signals [61]. Our analysis shows that the proposed method has the same rate of decay as an independent coding scheme (such as JPEG 2000 [24]), but with different scaling constants.

The thesis is concluded in Chapter 7. Here, we overview the presented results and outline possible directions for future work.

1.4 Original Contributions and Publications

The main contribution of this thesis is the development of the centralised and interactive compression methods and the derivation of the RD model. The design of the sparse representation is in part based on the work in [6]. More specifically, we use the layer-extraction method proposed in [8], which takes into account the structure of multiview images. The remaining work on the sparse representation in Section 3.3 has been developed by the author of this thesis.

The presented material in this thesis has resulted in the following publications:

- A. Gelman, P.L. Dragotti and V.Velisavljevic, “Multiview Image Coding using Depth Layers and An Optimized Bit Allocation”, accepted subject to minor revisions to *Transactions of Image Processing*, January 2012.
- A. Gelman, J. Berent and P.L. Dragotti, “Layer-based sparse representation of multiview images”, accepted to *EURASIP Journal on Advances in Signal Processing*, July 2011.
- A. Gelman, P.L. Dragotti and V.Velisavljevic, “Interactive Multiview Image Coding”, in *Proceedings of International Conference on Image Processing (ICIP)*, Brussels, September 2011.
- A. Gelman, P.L. Dragotti and V.Velisavljevic, “Centralized and Interactive Compression of Multiview Images”, in *Proceedings of SPIE Applications of Digital Image Processing XXXIV*, San Diego, August 2011.
- A. Gelman, P.L. Dragotti and V.Velisavljevic, “Multiview Image Compression using a Layer-Based Representation”, in *Proceedings of International Conference on Image Processing (ICIP)*, Hong Kong, September 2010.
- A. Gelman, P.L. Dragotti and V.Velisavljevic, “Layer-Based Multi-view Image Compression”, in *Proceedings of International Mobile Multimedia Conference (MobiMedia)*, Kingston, UK, September 2009.

Chapter 2

Multiview Data Structure and Compression

In this chapter we start by reviewing the structure of multiview images. Specifically, we introduce the plenoptic function and IBR. Then we show that the multiview structure is determined by the objects in the scene and their relative depth to the camera array. Based on this analysis we describe the layer-based representation [8]. The representation segments multiview images into layers, where each layer is related to a constant depth in the scene. To conclude this chapter, we also review a number of existing compression algorithms in the literature.

2.1 Plenoptic Function

In the IBR framework, multiview images form samples of a multi-dimensional signal called the *plenoptic function* [3]. Introduced by Adelson and Bergen, this function parameterises each light ray with a 3D point in space (V_x, V_y, V_z)[1] and its direction of arrival (θ, ϕ). Two further variables λ and t are used to specify the wavelength and time, respectively. In total the plenoptic function is seven dimensional:

$$I = P_7\left(V_x, V_y, V_z, \theta, \phi, \lambda, t\right), \tag{2.1}$$

[1]This can also be analysed as the location of a sampling camera.

where I corresponds to the light ray intensity.

The continuous version of the plenoptic function stores all of the information contained in a scene, and any viewpoint can be synthesised by selecting the correct light rays. In practise the plenoptic function is sampled, for example using a conventional camera. The goal of IBR is then to synthesise an image at a novel viewpoint by estimating the missing light rays from the sampled plenoptic function. This has resulted in IBR to be analysed as a sampling problem in [15], and an interesting result is that the spectrum is bandlimited by the maximum and minimum depth in the scene.

Due to the number of dimensions of the plenoptic function it is not easy to sample each of the dimensions. Therefore, in practise a number of simplifications are applied to reduce its dimensionality. Firstly, it is common to drop the λ parameter and instead deal with either the monochromatic intensity or the red, green, blue (RGB) channels separately. Secondly, many scenes are recorded in a static setting; this means that the plenoptic function can be defined at a specific moment in time, thus dropping the t parameter. The resulting object is a 5D function, and rendering using this object is known as plenoptic modelling [62].

Based on these assumptions a popular parameterisation of the plenoptic function is known as the *light field* [52] or equivalently the *lumigraph* [40]. Here the dimensionality of the plenoptic function is reduced to four by restricting the camera coordinate $V_z = 0$. In this representation, each light ray is defined by its intersection with a camera plane and a focal plane:

$$I = P_4\left(V_x, V_y, x, y\right), \tag{2.2}$$

where as illustrated in Fig. 2.1, (V_x, V_y) and (x, y) correspond to the coordinates of the camera plane and the focal plane, respectively.

The light field can also be analysed as a 2D array of images. For example, in Fig. 2.2 we illustrate a light field with 16 camera locations. The camera positions are evenly spaced on a 2D grid indexed with V_x and V_y. This demonstrates the close relationship between the plenoptic function and multiview images.

The light field can be further simplified by setting the 2D camera plane to a line.

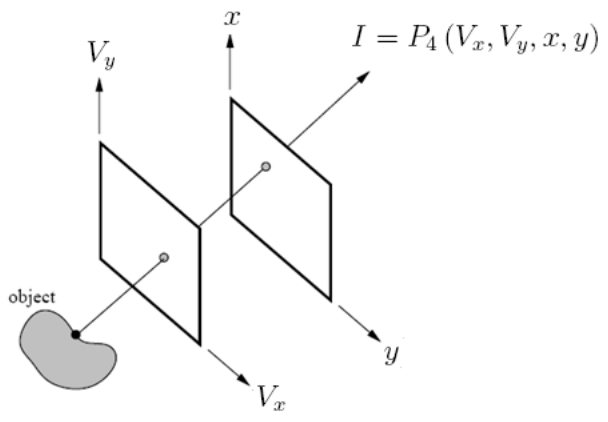


Figure 2.1: Light field parameterisation. Each light ray is defined by its intersection with a camera plane (V_x, V_y) and a focal plane (x, y) [82].

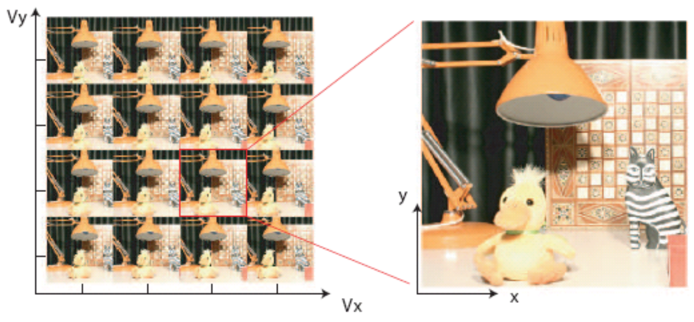


Figure 2.2: Captured light field [6]. Dataset can be analysed as a 2D array of images.

This representation is equivalent to a 1D array of images and is known as the Epipolar-plane image (EPI) volume [11]:

$$I = P_3(V_x, x, y)\,. \tag{2.3}$$

In comparison to the light field, the EPI is easier to visualise and in this thesis we use it to illustrate a number of concepts. All of the properties are however easily generalised to the light field. Next, we review the EPI and light field data structure and present the layer-based representation [8].

2.2 EPI and Light Field Structure

In this section we show that an EPI volume and a light field are structured datasets. By structure we mean that the fundamental component of multiview images are lines along which the pixel intensity is approximately constant; this concept is shown in Fig. 2.3(c). This illustration is obtained by stacking an array of images into a volume and taking a cross section through the dataset. It can be observed that pixels are redundant along lines of varying gradients. The set of pixels along which the intensity of the volume is constant is known as an *EPI line*.

In order to demonstrate why the fundamental component of multiview images are EPI lines, consider the setup in Fig. 2.3(a). Here we show a simplified version of the scene: the horizontal axis corresponds to the camera location line; the line parallel to it defines the focal plane of each camera[2]; and the vertical axis defines the depth of the scene. The curved line corresponds to the surface of the object.

Given this setup consider a point in space with coordinates (X, Y, Z). Assuming a Lambertian scene[3] this point will appear in each of the images (V_x) with coordinates

$$x = \frac{fX}{Z} - \frac{fV_x}{Z}, \quad (2.4)$$

$$y = \frac{fY}{Z}, \quad (2.5)$$

where f is the focal length. As illustrated in Fig. 2.3(b), the spatial coordinate x is linearly related to the camera location V_x. The rate of change in the pixel location, also known as the *disparity gradient* $\Delta p = \frac{f}{Z}$, is inversely related to the depth of the object. This analysis tell us that a point in space with coordinates (X, Y, Z), maps to a constant intensity line in the EPI volume. Moreover, objects closer to the focal plane (smaller Z), correspond to lines with a steeper gradient.

The EPI lines in the volume have varying gradients (due to different objects in the scene) and may intersect at a point. Clearly, when two lines intersect, the EPI line corresponding to a smaller depth (larger disparity gradient) will occlude all the EPI

[2]Each camera in the setup is modelled by the pinhole model [41].
[3]Light ray intensity is constant when an object is observed from a different angle.

lines which are related to larger depths (smaller disparity gradient) in the scene. This principle is illustrated in Fig. 2.3(c).

The EPI line concepts can also be extended to the light field, where the camera is allowed to move along two dimensions (V_x, V_y). In this case, a point (X, Y, Z) maps onto a 2D plane as

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \rightarrow \begin{pmatrix} x \\ y \\ V_x \\ V_y \end{pmatrix} = \begin{pmatrix} (X - V_x)\, f/Z \\ (Y - V_y)\, f/Z \\ V_x \\ V_y \end{pmatrix}. \tag{2.6}$$

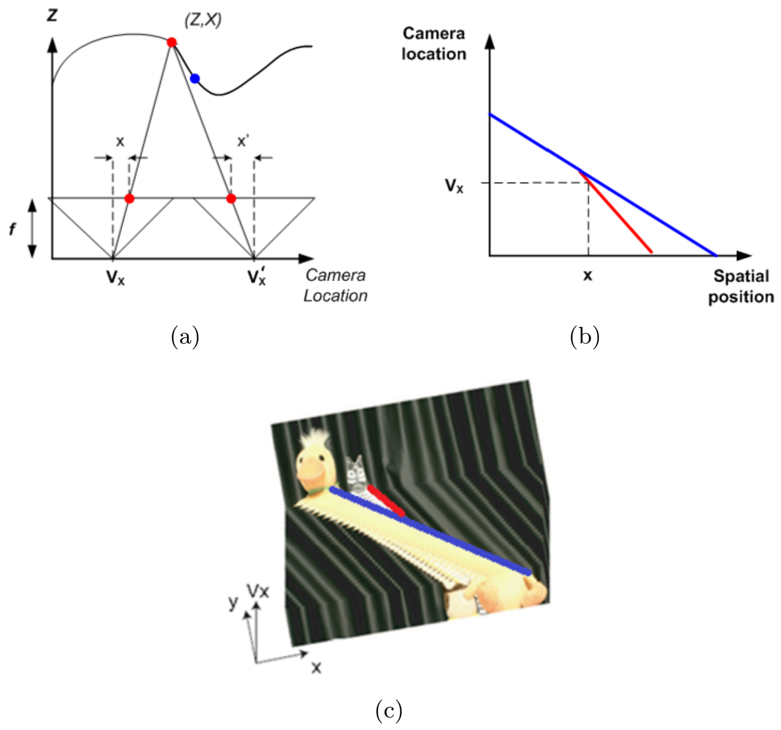


Figure 2.3: (a) Camera setup. The sampling camera moves along a straight line; the direction of the camera is perpendicular to the camera location line. (b) Each point in space maps to a line in the EPI volume. Observe that the blue object is closer to the focal plane and therefore occludes the red object. It can be shown using (2.4) and (2.5) that a data sample (x, y, V_x) can be mapped onto a different viewpoint V_x' with spatial coordinates $x' = x - \frac{f(V_x' - V_x)}{Z}$ and $y' = y$. (c) Shows a cross section of an EPI volume. This figure is obtained by stacking a 1D array of images into a volume and analysing a slice in the dataset. Two EPI lines which correspond to two points in space are illustrated.

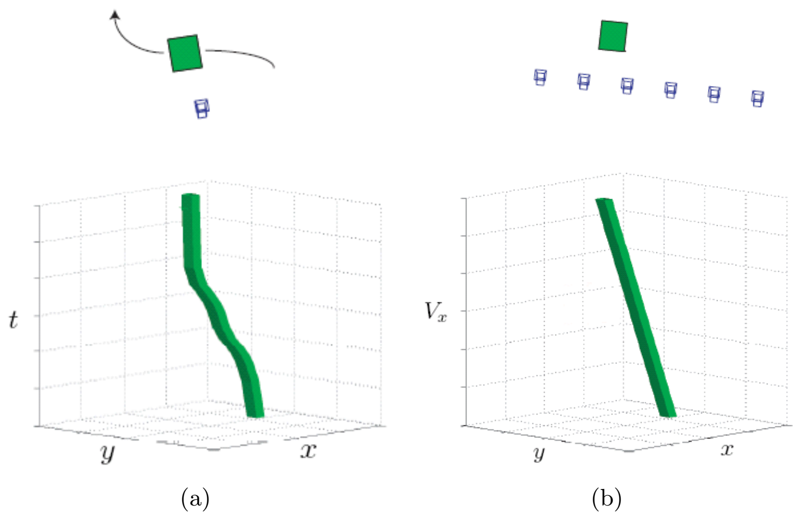


Figure 2.4: A comparison between a layer carved out in a video and an EPI volume [6].

2.3 Layer-based representation

In the previous section we discussed the structure of multiview images. It was shown that captured datasets such as the light field and the EPI volume consist of EPI lines along which the pixel intensity is constant. This concept can be further extended by grouping EPI lines with the same gradient into a single volume (we call this a layer). A multiview array can be segmented into a set of layers, where each one corresponds to a constant depth in the scene. We call this the *layer-based representation*. An example of the representation is illustrated in Fig. 2.7.

It is intuitive to introduce the concept of layers using a standard video. Consider a moving object in front of a stationary camera. If we track the object in each frame, the object will carve out a volume in time [90]. A carved out synthetic volume is shown in Fig. 2.4(a)

A carved out video volume has a close connection to a layer in a multiview image array. Recall that camera motion and object motion are relative. For example, we can obtain the same video dataset by having a stationary object and a moving camera; this corresponds to a multiview image array[4]. By contrast in a multiview setup, the

[4]This is only true when there is one moving object in the scene. It is not possible to obtain the same dataset when there are multiple objects experiencing different motion.

camera motion is constrained (the camera can only move on a 2D array.). Therefore, it is intuitive to deduce that multiview images will also consist of layers. In addition, due to the constrained camera motion, a multiview layer will be more structured; a synthetic example is shown in Fig. 2.4(b).

Next we describe in more detail the structure of the layer-based representation. We show that the layer segmentation can be efficiently described by a set of contours on an image taken from one viewpoint and the corresponding depth of each layer. In addition we discuss what happens when two or more layers overlap. In videos, it is not possible to deduce the occlusion ordering using the motion of objects. However, in multiview images, we show that this property can be correctly inferred simply by knowing the layer gradients (disparities).

2.3.1 Efficient boundary description and occlusions

To provide a more precise analysis, consider a set of EPI lines modelled by a constant disparity gradient Δp_k as shown in Fig. 2.5(a). We define the layer carved out by the EPI lines with $\mathcal{H}_k$. The layer can be separated from the 3D space by its boundary Γ_k. Here the subscript k corresponds to the k-th layer in the dataset.

Just as in the case of EPI lines, we know that the surface Γ_k will be structured. In fact, assuming there are no occlusions, the surface Γ_k can be defined by a 2D contour on one viewpoint projected[5] to the remaining frames. More specifically, if we define a contour $\gamma_k(s) = [x(s), y(s)]$ to be the boundary on the viewpoint ($V_x = 0$), we obtain the relationship

$$\Gamma_k(s, V_x) = \begin{pmatrix} x(s) - \Delta p_k V_x \\ y(s) \\ V_x \end{pmatrix}, \tag{2.7}$$

where s parameterises the contour $\gamma_k(s)$. This equation outlines that the surface is created by shifting the 2D contour onto each of the images. This concept is further illustrated in Fig. 2.6(a) which shows an unoccluded layer from the Animal Farm

[5]In this setting, a projection onto a new camera location will simply be a shift along the x-axis in spatial coordinates of the new camera location.

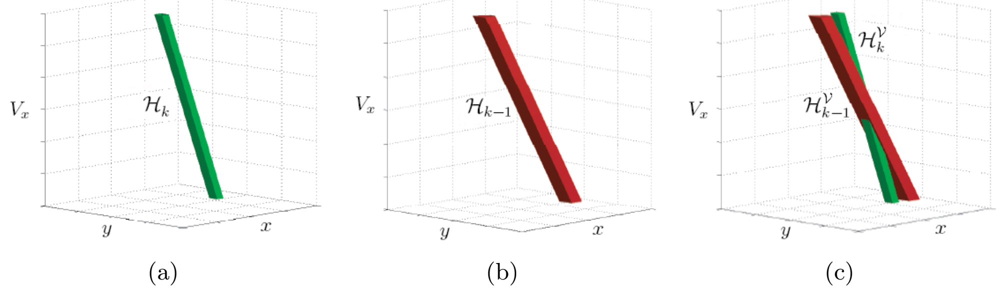


Figure 2.5: Comparison between two layers $\mathcal{H}_{k-1}$, $\mathcal{H}_k$ and their intersection [6]. The layers are ordered in terms of depth (i.e. $\mathcal{H}_{k-1}$ corresponds to a smaller depth than $\mathcal{H}_k$). (a) A set of EPI lines related to constant disparity gradient Δp_k. The collection of EPI lines carve out a layer $\mathcal{H}_k$. Observe that the complete segmentation of the layer can be defined by a boundary on one viewpoint projected to the remaining frames. (b) $\mathcal{H}_{k-1}$ modelled by a constant disparity gradient Δp_{k-1}. (c) When the two layers intersect, $\mathcal{H}_{k-1}$ will occlude $\mathcal{H}_k$ as it is modelled by a smaller depth. We define the visible volumes with $\mathcal{H}_{k-1}^{\mathcal{V}}$ and $\mathcal{H}_k^{\mathcal{V}}$.

dataset [7]. The complete segmentation is defined by the red boundary $\gamma_k(s)$ on the first image viewpoint shifted to the remaining frames.

Note that the above analysis does not take into account occluded/disoccluded regions. In order to include these properties in the formulation, we can use the same reasoning as in the case of EPI lines: a layer will only be occluded when it intersects with other layers which are related to a smaller depth (larger disparity gradient) in the scene. We illustrate this in Fig. 2.5, which shows that when two layers intersect we obtain their visible representations[6] $\mathcal{H}_{k-1}^{\mathcal{V}}$ and $\mathcal{H}_k^{\mathcal{V}}$. In this example the layers are ordered in terms of increasing depth (i.e. $\mathcal{H}_k$ corresponds to an object with a larger depth than $\mathcal{H}_{k-1}$).

To further clarify the difference between $\mathcal{H}_k$ and $\mathcal{H}_k^{\mathcal{V}}$; layer $\mathcal{H}_k$ is obtained from a contour on one viewpoint $\gamma_k(s)$, whereas its visible representation $\mathcal{H}_k^{\mathcal{V}}$ corresponds to the same layer, but with its occluded regions removed.

In general, the visible representation of a layer can be defined as

$$\mathcal{H}_k^{\mathcal{V}} = \mathcal{H}_k \bigcap \overline{\left(\bigcup_{j=1}^{k-1} \mathcal{H}_j \right)}. \tag{2.8}$$

[6]By visible regions we mean the EPI line segments which are present in the original EPI volume.

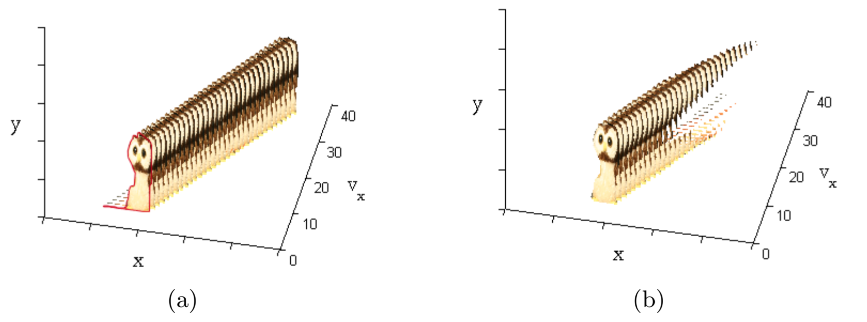


Figure 2.6: Layer from the Animal Farm dataset. (a) The unoccluded layer $\mathcal{H}_k$ can be defined using the contour $\gamma_k(s)$ on one viewpoint projected to the remaining frames. The 2D contour is denoted by the red curve on the first image. (b) Occluded layer $\mathcal{H}_k^{\mathcal{V}}$ can be inferred by removing the regions which intersect with other layers related to a smaller depth.

We illustrate the layers $\mathcal{H}_k$ and $\mathcal{H}_k^{\mathcal{V}}$ from the Animal Farm dataset in Fig. 2.6.

There are a number of advantages to segmenting a multiview dataset into layers. Firstly, each layer is highly redundant in the direction of the disparity gradient Δp. Secondly, any occluded regions are explicitly defined by the representation. These regions correspond to artificial boundaries, and their specific locations can be used to design a transform which takes them into account. Thirdly, the segmentation of each layer can be efficiently defined by a contour on one viewpoint $\gamma(s)$ and its disparity gradient Δp. This property is important in terms of data compression, where the segmentation of each layer must also be transmitted.

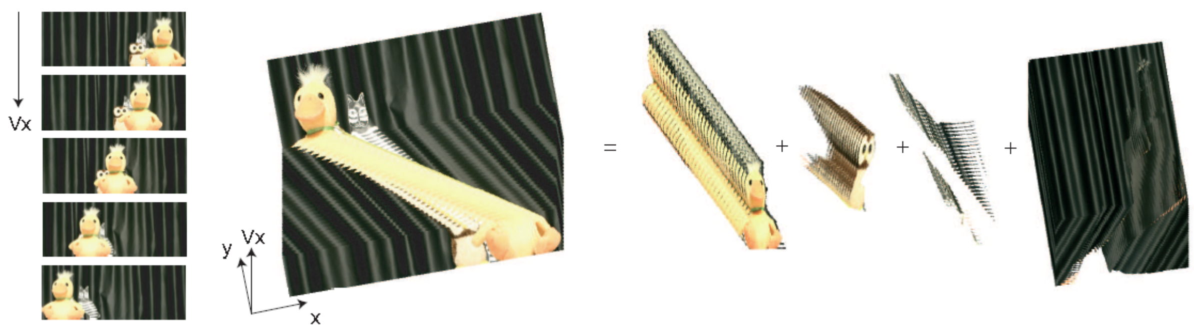


Figure 2.7: Animal Farm layer-based representation [6]. The dataset can be divided into a set of volumes, where each one is related to a constant depth in the scene. Observe that the layer contours at each viewpoint remain constant, unless there is an intersection with another layer which is modelled by a smaller depth.

2.4 Multiview Image Compression

In this section we review existing multiview image compression algorithms. A variety of methods have been proposed which trade-off encoding/decoding complexity, scalability and random access [81, 97].

We separate the analysis into the following sections. Initially, we describe a number of conventional methods for encoding a 2D array of images. These have been mainly designed to reduce the storage requirements. Then, we present a number of schemes proposed for the interactive setup. Recall that in this scenario multiview images are stored at a central server and are transmitted to the client on request. This type of setup places more emphasis on random access. We also present a number of innovative solutions that approach this problem by storing different versions of the data at the server in order to reduce the transmission rate.

2.4.1 Conventional Multiview Image Coding

Disparity Compensated Prediction

Multiview images share many similarities with traditional videos. Thus, a number of video coding tools [92] have been applied to multiview images in the literature. A popular approach is to use disparity compensated prediction (DCP). In DCP multiview images are segmented into blocks and each block is predicted from previously coded images by minimising an error metric[7]. Only the residual error (prediction error) is quantised and transmitted to reduce the bit rate. The concept of DCP has been extended to 2D multiview image arrays by applying the compensation along the two viewing dimensions. As illustrated in Fig. 2.8, prediction-based algorithms encode the data by partitioning the images into intra and inter frames. The intra frames are independently encoded using traditional image compression techniques and the inter frames are predicted from the nearest intra images [47, 96]. In [57] a hierarchical DCP coding approach was used; in this setup, the first and last images are encoded in intra modality. Then, the middle view is coded as a prediction of the two frames. The

[7]Popular metrics include mean squared error and absolute error.

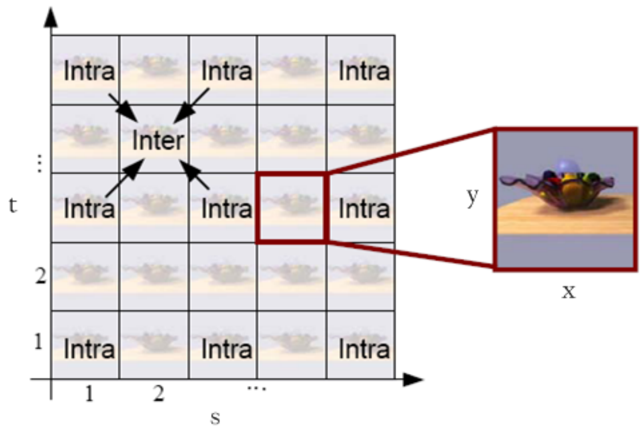


Figure 2.8: Prediction-based light field encoding. The light field images are partitioned into inter and intra frames. The intra frames are encoded independently, while the inter-frames are predicted from the intra frames [97].

hierarchical approach subsequently continues to encode the remaining views. These type of methods can be further optimised by choosing different block sizes and selection modes for each block [59] similar to H.264/AVC [92].

Although the proposed methods achieve a high compression, the prediction structures used during coding reduce random access. For example to decode one image it may be necessary to fully decode all of the reference frames.

To resolve this issue Li and Zhang [96] used the analysis that a particular set of coefficients has a fixed number of reference blocks. By creating a look-up table with a set of pointers to each block in the dataset, they facilitated random access by decoding only the necessary transform coefficients. In addition, they cached the transmitted images to reduce the decoding time of future frames (in case the frames are reused).

In a conventional approach, random access is affected due to the large number of possible reference frames. In [19], the authors proposed to modify the encoding structure to deal with this problem. In their approach, they restrict the reference frames to be the corner images. The remaining images are then predictively coded using only these views. However, in this setup, the compression performance is limited by the number of images in the dataset: when the number of images is large, the corner images may be a poor predictor and hence result in a large residual error.

DCP has also been a popular approach in stereo image coding due to its applications in 3D TV. This is a special case of multiview images, where only two input views are considered. The proposed methods [77, 94, 95] are similar to the ones outlined above, where one image is encoded in intra modality and used as a prediction for the second view.

Model-based and Wavelet Methods

A number of schemes proposed in the literature use the scene geometry (depth of the objects in the scene) to improve the compression performance. The authors in [73] used geometry information to predict the disparity (motion) vectors in order to reduce the overall bit rate. The model has also been used to infer the silhouette of an object in [18, 39]; this information was then used to improve the coding efficiency by applying a shape adaptive transform in the region of interest. In [55], the model was used to infer appearing and disappearing objects in the images to further improve performance.

In [38, 58] a different approach was used; the authors proposed to map the original multiview images onto texture maps using the 3D model. The main idea in this method is that texture maps are more redundant than the original images, and therefore higher compression can be achieved. However, the transformation that maps images onto texture maps is not invertible and this implies that it is not possible to reconstruct the original dataset. Furthermore, estimating a 3D model in a cluttered scene is a complicated and error prone process [49].

A popular approach to encode multiview images has also been to use the Discrete Wavelet Transform (DWT). The DWT has been widely used in compression due to its ability to represent natural features such as edges and smooth regions. In addition, the representation provides a framework to support both bit-rate and spatial scalability. These properties have resulted in the DWT being adopted in the JPEG 2000 [24] standard. The separable properties of the DWT [61] also mean that the method can naturally be extended to multi-dimensional datasets [20, 75, 86].

A natural approach to encode the light field is to extend the 2D wavelet coder to operate on the additional camera viewpoint dimensions. This implementation was used in [56] where a 4D DWT was applied to de-correlate the data. In this case, disparity compensation was not included and this resulted in a large number of high-pass coefficients, therefore reducing compression performance. In [37], the coding gain was increased by using a warping operator. The warping operator maps an image onto a different viewpoint and thus increases the inter view correlation in the DWT. In addition, the lifting implementation [27] was used to reduce the complexity of the algorithm and ensure the transformation is invertible (even if the warping operator is not invertible).

2.4.2 Texture and Model Rate Allocation

Many existing compression schemes rely on a 3D model of the scene to perform disparity compensation or to map images onto different viewpoints. To correctly reconstruct the data, the scene model must also be transmitted. Therefore, there is a trade-off in terms of the number of bits allocated to encoding the texture and the model. One approach is to losslessly encode the model at all bit rates [35, 55]. However, this is clearly sub-optimal since the model itself only defines the accuracy of the disparity compensation. At low rates it is intuitive to allocate significantly more bits to the texture (residual error) than the 3D scene. The bit rate allocation problem between texture and scene geometry has, for example, been studied in [60] and [26], where the geometry is defined using a per pixel disparity map. However, in our case the model is defined by a set of contours and this requires a different RD model to the one proposed in the literature.

An additional problem is that lossy coding of the model indirectly affects the reconstructed multiview image distortion. For example, errors in the model introduce errors in disparity compensation. These pixels reduce the coding efficiency and when quantised, they contribute to the distortion. Therefore, it is difficult to develop a RD model for optimisation. In our approach we deal with this problem by considering the worst case distortion scenario. Using this setup we can derive a closed-form expression for the rate allocation between the model and the transform coefficients.

2.4.3 Interactive Compression Methods

In the interactive setup a client downloads images from a central server on request. In this case, only a subset of the data may be required and this approach allows the user to significantly reduce the transmission rate. For example consider a dataset which stores all of the images for an immersive viewing experience of a museum. The total size of the data is extremely large and it is redundant to transmit all the information, when only a small proportion of the images is actually going to be viewed. Therefore, the server and client interactively communicate; as the user moves around the scene he requests the images which are needed to render a novel viewpoint.

There are a number of points which must be considered in this encoding setup. First, the server must support random access; it is not practical to decompress the data at the server each time there is a request from the client[8]. Second, the viewing trajectory is unknown before the compression[9] [21]. Therefore, the encoding method must be able to handle this uncertainty.

There are a number of ways in which these requirements can be met. One option would be to encode each image independently. It has been shown that in this setup an independent coding approach can outperform a joint encoding scheme [70,71]. This occurs because the joint scheme requires that all the dependent frames, which may not be required for rendering are also transmitted. Although an independent method can outperform a joint scheme, the approach is still inefficient since the inter-view redundancy in the transmitted images is not exploited.

A different solution is to store at the server the residual signals for all the possible combinations of the viewed images [22]. This leads to a low transmission rate, however requires a large storage requirement at the server. If a conventional closed loop predictive coding scheme is used, the storage requirement is of the order $\mathcal{O}\left(N^N\right)$, where N is the total number of images. The is due to the fact that a reconstructed image also depends on the reference frame used [21]. For example, decoding frame 2 given

[8]This is only possible when the number of clients is small.
[9]If the viewing trajectory was known, each frame could be differentially encoded in a setup similar to traditional video coding.

frame 1 as reference does not give the same result as decoding frame 2 given frame 3 as reference. Therefore, to ensure there is no error propagation (drift) each possible combination must be encoded.

To reduce the storage requirement issue, the use of Switching Predictive (SP) frames [45] from H.264/AVC was proposed in [69]. These frames reconstruct the same image given different reference views, and this reduces the storage size to be in the order of $\mathcal{O}\left(N^2\right)$.

In general there is a trade-off between the storage at the server and the transmission rate. This is determined by the number of independent frames and SP-frames in the dataset. Given a certain storage and transmission requirements, this can be analysed as a constrained optimisation problem to find the correct frame structure. A solution to this problem based on Lagrangian multipliers was proposed in [21].

In addition to SP-frames, Distributed Source Coding (DSC) [32, 67] has been used to support random access while reducing the transmission rate and storage requirements. DSC encodes each image independently and instead shifts the complexity to the client. At the client, the data is jointly decoded using side information. In this context, the side information is created using the previously reconstructed images. This approach was used in [2] where the images were partitioned into intra and Wyner-Ziv (DSC) frames. The intra frames were encoded using a conventional independent coding scheme, whereas parity bits were transmitted from the Wyner-Ziv images. The Wyner-Ziv frames were then decoded by generating side information at the client. One of the issues in interactive viewing of a scene is that there is an uncertainty as to which image will be available in the cache of the user for prediction. In [23] DSC principles were used to remove this ambiguity by encoding the Wyner-Ziv frames with respect to the worst possible error. Thus, all frames could be correctly decoded using any side information in the cache.

2.5 Summary

In this chapter we discussed multiview images. It was shown that multiview images, such as an EPI volume or a light field are structured datasets and consist of EPI lines, along which the intensity of the pixels is constant. In addition it was shown that the correct EPI line occlusion ordering can be inferred by the disparity gradient of each line.

Based on the EPI line concept, we presented the layer-based representation. The model outlines that multiview images can be analysed as a sequence of layers, where each layer is related to an object at a constant depth in the scene. The advantages of using this representation are as follows: each layer is redundant in the direction of the disparity gradient; the occluded regions of each layer are known; and the segmentation can be efficiently described using a sequence of 2D contours and a set of disparities.

In addition we outlined a number of compression algorithms which have been proposed in the literature for both the interactive and the centralised setup.

Chapter 3

Sparse Layer-based Representation of Multiview Images

The notion of sparsity is the idea that a signal can be defined with a small number of significant components which contain most of the signal's information. Sparse signal representations are at the heart of many successful signal processing applications, such as compression and denoising [12].

In compression, the idea is to find basis functions which closely match the data structure. In the transform domain, this creates a decomposition with the majority of the transform coefficients close to zero. The coefficients are then quantised and the total bit budget can be effectively allocated among the small number of non-zero coefficients. An example in image coding is JPEG 2000 [24] which uses a 2D DWT to obtain a sparse decomposition. The bit budget is then allocated among the transform coefficients using Lagrangian multipliers [85]. In this thesis, we demonstrate how a constrained optimisation problem can be solved using Lagrangian multipliers in Section 4.3.

In denoising, a model which creates a sparse representation can be used to remove significant amounts of noise while retaining salient signal features. The denoising is implemented by assuming the original signal is sparse following the transformation. If the signal is corrupted by additive white Gaussian noise (AWGN), we can obtain an approximation of the original sparse signal by setting the transform coefficients below a certain threshold to zero. The denoised version of the signal is then obtained by applying the inverse transformation.

As outlined above, a sparse decomposition is important for various signal processing applications. In the case of images, the data is composed of smooths regions separated by edges. This simple analysis has contributed to the development of a number of data driven representations such as curvelets [13], contourlets [30], ridgelets [29], directionlets [88], bandlets [50, 51] and complex wavelets transform [5, 76]. All these representations aim to achieve a sparse decomposition of images. However, when dealing with multiview images, the data model must take into account appearing (disocclusions) and disappearing (occlusions) objects. This nonlinear property means that finding a sparse representation is significantly more difficult than in the single image case.

We therefore use a hybrid method to find a sparse representation of multiview images. The fundamental component of the algorithm is the layer-based representation highlighted in Section 2.3. Recall, that the layer-based representation partitions multiview images into a set of layers each related to a constant depth in the observed scene. We review a method to extract these regions [8], which takes into account the structure of multiview data to achieve accurate results. Then, given the extracted layers, we obtain the sparse representation by applying a 4D DWT[1] to each layer.

This chapter is organised as follows. Next we present a high-level overview of the proposed method. In Section 3.2 we outline the layer extraction algorithm. The multidimensional DWT is discussed in Section 3.3. We evaluate the sparsity of the proposed representation in Section 3.4 and summarise the chapter in Section 3.5.

[1]A light field has four dimensions and therefore we use a 4D DWT. In the case of an EPI volume we use a 3D DWT.

3.1 High-level overview

A high-level overview of the proposed method is shown in Fig. 3.1. The input to the algorithm is an array of multiview images. The multiview images can either correspond to an EPI volume (1D array of images) or a light field (2D array of images).

Given the input, the first step of the method is to segment the multiview dataset into a sequence of layers, where each layer is modelled by a constant depth in the scene. To extract these layers, we define a cost functional[2], which is initialised by a set of layer boundaries. The boundary of each layer is then iteratively evolved to minimise the cost. Furthermore, we take into account the camera setup and occlusion constraints to simplify the evolution from a 4D surface to a 2D contour.

Following the layer extraction, we obtain the sparse decomposition by applying a 4D DWT in a separable fashion across the viewpoint and the spatial dimensions. We modify the viewpoint transform to include disparity compensation and also efficiently deal with occluded regions. Additionally, the transform is implemented using the lifting scheme [27] to reduce the complexity and maintain invertibility.

3.2 Layer-based segmentation

Data segmentation is the first stage of the proposed algorithm. We introduce the method by first describing a general segmentation problem. Then, we show how the solution can be adapted to extract layers from a multiview image dataset by taking into account the properties of the data.

3.2.1 General data segmentation

Consider a general segmentation problem shown in Fig. 3.2. The aim is to partition a m-dimensional dataset $\mathcal{D} \subset \mathbb{R}^m$ into subsets $\mathcal{H}$ and $\overline{\mathcal{H}}$ where the boundary which separates the two regions is defined by $\Gamma(\boldsymbol{\sigma})$ with $\boldsymbol{\sigma} \in \mathbb{R}^{m-1}$. This type of problem can be solved using an optimisation framework, where the boundary is obtained by

[2]A functional is function which takes in other functions as variables.

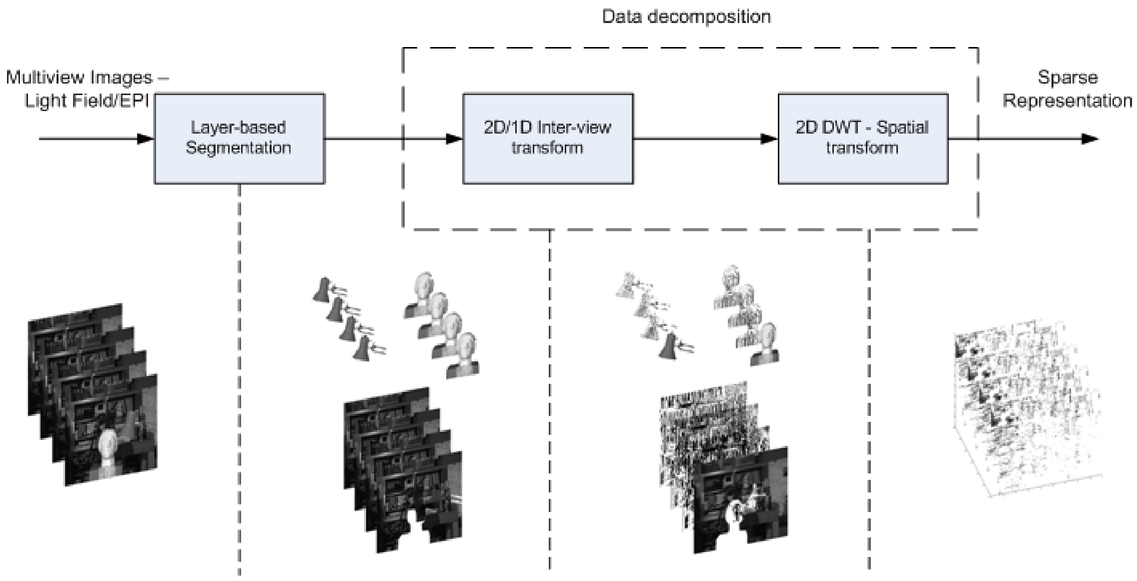

Data decomposition
Multiview Images – Light Field/EPI
Layer-based Segmentation
2D/1D Inter-view transform
2D DWT - Spatial transform
Sparse Representation

Figure 3.1: High-level algorithm block diagram. The data is initially segmented into layers where each volume is related to a constant depth in the scene. The obtained layers are then decomposed using a 4D DWT along the viewpoint and spatial dimensions. Additionally, we illustrate the obtained transform coefficients at each stage of the method.

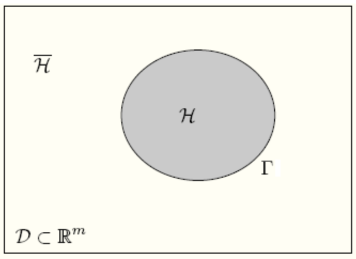


Figure 3.2: A m-dimensional dataset $\mathcal{D}$ is partitioned into $\mathcal{H}$ and $\overline{\mathcal{H}}$. The boundary is a closed curve defined by Γ [6] and is a (m-1)-dimensional object.

minimising an objective function J:

$$\Gamma_{opt} = \arg\min\{J(\Gamma)\}. \tag{3.1}$$

A popular approach in segmentation is to first initialise a boundary Γ_0. The boundary is then iteratively evolved in the direction which minimises the cost function. The boundary evolution is modeled as a partial differential equation (PDE). By making the boundary a function of a time parameter τ, the evolution is defined as

$$\frac{\partial\Gamma\left(\boldsymbol{\sigma},\tau\right)}{\partial\tau}=\mathbf{v}\left(\boldsymbol{\sigma},\tau\right)=F\left(\boldsymbol{\sigma},\tau\right)\mathbf{n}\left(\boldsymbol{\sigma},\tau\right), \tag{3.2}$$

where $\mathbf{v}$ is a velocity vector, which can be expressed in terms of a scalar force F acting at each point on the boundary. The direction of the force is in the inward normal direction to the boundary $\mathbf{n}$. Note that the force can also be negative, in which case, the evolution would be in the outward normal direction and act to expand the surface.

The velocity vector determines the evolution of the boundary with time. For example, given $\mathbf{v}$, the boundary at time τ can be estimated using the first order Taylor series as

$$\Gamma\left(\boldsymbol{\sigma},\tau\right)=\Gamma\left(\boldsymbol{\sigma},0\right)+\tau\frac{\partial\Gamma\left(\boldsymbol{\sigma},0\right)}{\partial\tau}, \tag{3.3}$$

where $\Gamma\left(\boldsymbol{\sigma},0\right)$ is the initialised boundary Γ_0. These type of boundaries are known as geodesic active contours [14] or snakes [46].

In order to evaluate a velocity vector which minimises the cost function, we first need to construct the cost function itself. Next we discuss two such techniques known as boundary-based and region-based schemes.

Boundary-based segmentation

A classical approach to define the cost in (3.1) is to sum a contribution at each point on the boundary Γ. These are known as boundary-based methods [14]:

$$J\left(\Gamma\right)=\int_{\Gamma}d^{b}\left(\mathbf{x}\right)d\boldsymbol{\sigma}. \tag{3.4}$$

The function $d^{b}\left(\mathbf{x}\right)$ with $\mathbf{x}\in\mathbb{R}^{m}$ is known as a descriptor and it determines the weight at each point on the boundary. We can choose the descriptor such that the cost is minimised when the boundary lies on a perfect edge

$$d^{b}\left(\mathbf{x}\right)=\frac{1}{1+\left|\nabla I\left(\mathbf{x}\right)\right|}, \tag{3.5}$$

where $|\nabla I(\mathbf{x})|$ is the magnitude of the gradient operator. In this case, the cost will tend to zero as the boundary approaches a perfect edge.

The velocity vector $\mathbf{v}$ can be derived in terms of the descriptor by making the boundary a function of the evolution parameter and then differentiating (3.4) with respect to τ. In this setup it has been shown [14] that the velocity vector can be derived as

$$\mathbf{v} = \left[d^b(\mathbf{x})\,\kappa(\mathbf{x}) - \nabla d^b(\mathbf{x}) \cdot \mathbf{n}\right]\mathbf{n}, \tag{3.6}$$

where $\kappa(\mathbf{x})$ is the mean curvature of the surface at $\mathbf{x}$, and $\cdot$ denotes the dot product. In this equation, the first term $\left(d^b(\mathbf{x})\,\kappa(\mathbf{x})\right)$ will create a positive inward force and this acts to reduce the overall length of the boundary. For example, assuming that $d^b(\cdot)$ is a constant, the velocity becomes proportional to the curvature. In this case, regions on the surface with a high curvature will create a large force in the inwards normal direction, and this causes regions of high curvature to contract. If this process continues, the surface would eventually become a multi-dimensional sphere and continue shrinking to an empty set. In the process, the cost would also reduce to zero.

The second term $\left(-\nabla d^b(\mathbf{x}) \cdot \mathbf{n}\right)$ in (3.6) also creates a force which acts to reduce the overall cost function. We know from standard calculus that $-\nabla d^b(\mathbf{x})$ is a vector[3] which points in the direction of the largest decrease in the value of the descriptor. Thus, intuitively, we would like to evolve each point of the boundary in this direction. However, recall that the evolution is fixed beforehand to be in the normal direction and the only free parameter we can choose is the force. Therefore, if the normal direction corresponds to the anti-gradient, the dot product will be maximised and this will result in a positive force. On the other hand, if the inward normal points in the opposite direction to the anti-gradient, the dot product will instead be minimised. This will result in a negative force and evolve the point on the boundary in the direction of the anti-gradient.

In summary, evolving the boundary by using the velocity vector in (3.6) will reduce the cost function with each iteration. One issue with this approach is that the cost

[3] $-\nabla d^b(\mathbf{x})$ is also known as the anti-gradient.

function is evaluated only on the boundary. Therefore, we cannot segment the data according to the properties inside the region. For example, instead of simply locating an edge, we may wish to segment the data by minimising the total variance inside the region. In addition, boundary based methods by definition only evaluate the local features and can thus be affected by noise. Next, we describe a region-based approach, which can be designed to segment the data according to global features.

Region-based segmentation

In region-based segmentation, the cost function is evaluated by summing the contribution in each of the regions. A typical function [43] can be defined as

$$J(\Gamma) = \int_{\mathcal{H}} d(\mathbf{x}, \mathcal{H})\, d\mathbf{x} + \int_{\overline{\mathcal{H}}} d\left(\mathbf{x}, \overline{\mathcal{H}}\right) d\mathbf{x} + \int_{\Gamma} \eta d\boldsymbol{\sigma}, \tag{3.7}$$

where the descriptor $d(\cdot)$ defines the weighting factor associated with each pixel. Note that the descriptor is a function of the region itself; for example it could be designed such that when $\mathbf{x}$ belongs to the region $\mathcal{H}$, $d(\mathbf{x}, \mathcal{H})$ tends to zero and vice versa. The cost function has an additional regularisation term η, which acts to minimise the length of the boundary. This term is evaluated as a summation on the surface and it is a special case of the boundary-based segmentation discussed in Section 3.2.1, where the descriptor function is set to a constant η.

The velocity vector associated with the minimisation of this type of function can be obtained by applying the Eulerian framework [43]. The derivative of the cost function can be evaluated as

$$\frac{\partial J(\Gamma(\tau))}{\partial \tau} = \int_{\Gamma(\tau)} \left[d\left(\mathbf{x}, \overline{\mathcal{H}}\right) - d(\mathbf{x}, \mathcal{H}) - \eta\kappa(\mathbf{x})\right] (\mathbf{v} \cdot \mathbf{n})\, d\boldsymbol{\sigma}. \tag{3.8}$$

Observe that $\mathbf{v}$ and $\mathbf{n}$ correspond to the velocity and the normal vectors in (3.2), respectively.

The velocity vector, which evolves Γ in the steepest descent direction can hence be deduced using the Cauchy-Schwarz inequality as:

$$\mathbf{v} = \left[d\left(\mathbf{x}, \mathcal{H}\right) - d\left(\mathbf{x}, \overline{\mathcal{H}}\right) + \eta\kappa\left(\mathbf{x}\right)\right]\mathbf{n}. \tag{3.9}$$

The above framework is also known as 'competition-based' segmentation. This is clear from (3.9), where a point on the boundary will experience a positive force if it belongs to the outer region $\overline{\mathcal{H}}$ and vice versa, hence evolving the contour in the correct direction. In summary, the region-based approach evaluates the cost on the complete region and is therefore more robust. We use this method in the following section to extract layers from multiview images.

3.2.2 Multiview Image Segmentation

In the case of a light field, the goal is to extract L layers, where each volume is modeled by a constant depth Z_k or by the associated disparity gradient Δp_k. In the context of the previous section, this is equivalent to segmenting the data into 4D layers $\{\mathcal{H}_1, \ldots, \mathcal{H}_L\}$, where the boundary of each layer is defined by $\{\Gamma_1, \ldots, \Gamma_L\}$ (the background volume $\mathcal{H}_L$ is assigned the residual regions which do not belong to any other layer).

In this setup $\mathcal{H}_k$ corresponds to a layer which is defined by a contour on one viewpoint and a disparity gradient as outlined in Section 2.3. However, due to occlusions the complete layer will not be visible in the dataset. Therefore, we define the cost function in terms of the visible regions $\mathcal{H}_k^{\mathcal{V}}$, and this leads to the following:

$$\min_{\{\Gamma_1,\ldots,\Gamma_L,\Delta p_1,\ldots,\Delta p_L\}} \left(\sum_{k=1}^{L} \int_{\mathcal{H}_k^{\mathcal{V}}} d_k\left(\mathbf{x}, \Delta p_k\right) d\mathbf{x} \right), \tag{3.10}$$

where $\mathbf{x} = [x, y, V_x, V_y]^T$. In practice, a regularisation term corresponding to each layer is also added to the cost function. Referring to (3.9), the value of η (regularisation constant) controls the smoothness of each layer. In our approach, we set $\eta = 0.02$ as originally proposed in [6]. For clarity, we do not include this term in the derivation of the velocity vector.

Recall that each layer contains EPI lines corresponding to a constant depth in the scene. Therefore, in the ideal setup the intensity along each EPI line is constant. We choose a descriptor $d_k(\mathbf{x}, \Delta p_k)$ to take this into account as [6]:

$$d_k(\mathbf{x}, \Delta p_k) = [I(\mathbf{x}) - \mu(\mathbf{x}, \Delta p_k)]^2, \tag{3.11}$$

where $\mu(\mathbf{x}, \Delta p_k)$ is the mean of the EPI line which passes through the point $\mathbf{x}$ and has a disparity gradient Δp_k. Note that to evaluate $\mu(\mathbf{x}, \Delta p_k)$, we do not take into account the pixels which lie outside the layer $\mathcal{H}_k^{\mathcal{V}}$ [6].

The aim of the segmentation is then to obtain the layer boundaries Γ_k and the disparity gradient Δp_k for $k = 1, \ldots, L$ by minimizing (3.10). However, (3.10) has a large number of unknown variables. In order to minimise the function, we consider the problem of layer evolution and disparity estimation separately.

Assuming the layer disparities are known, the minimisation can be simplified to

$$\min_{\{\Gamma_1,\ldots,\Gamma_L\}} \left(\sum_{k=1}^{L} \int_{\mathcal{H}_k^{\mathcal{V}}} d_k(\mathbf{x}, \Delta p_k)\, d\mathbf{x} \right). \tag{3.12}$$

One way to minimise (3.12) is to evolve iteratively the boundary of each layer. For example assuming that there are three volumes $\mathcal{H}_1$, $\mathcal{H}_2$ and $\mathcal{H}_3$ and that we choose to evolve the boundary of the first layer, the energy function can be expressed as

$$\begin{aligned} J_1 &= \int_{\mathcal{H}_1^{\mathcal{V}}} d_1(\mathbf{x}, \Delta p_1)\, d\mathbf{x} + \underbrace{\int_{\mathcal{H}_2^{\mathcal{V}}} d_2(\mathbf{x}, \Delta p_2)\, d\mathbf{x} + \int_{\mathcal{H}_3^{\mathcal{V}}} d_3(\mathbf{x}, \Delta p_3)\, d\mathbf{x}}_{\int_{\overline{\mathcal{H}_1^{\mathcal{V}}}} d_1^{out}(\mathbf{x}) d\mathbf{x}} && (3.13) \\ &= \int_{\mathcal{H}_1^{\mathcal{V}}} d_1(\mathbf{x}, \Delta p_1)\, d\mathbf{x} + \int_{\overline{\mathcal{H}_1^{\mathcal{V}}}} d_1^{out}(\mathbf{x})\, d\mathbf{x}, && (3.14) \end{aligned}$$

where $d_1^{out}(\mathbf{x}) = d_i(\mathbf{x}, \Delta p_i)$ when $\mathbf{x} \in \mathcal{H}_i^{\mathcal{V}}$ for $i = 2, 3$.

In general, when evolving the k-th layer, the cost function can be simplified to

$$J_k = \int_{\mathcal{H}_k^{\mathcal{V}}} d_k(\mathbf{x}, \Delta p_k)\, d\mathbf{x} + \int_{\overline{\mathcal{H}_k^{\mathcal{V}}}} d_k^{out}(\mathbf{x})\, d\mathbf{x}. \tag{3.15}$$

A possible solution would then be to evaluate the 4D velocity vector of the boundary corresponding to $\mathcal{H}_k^{\mathcal{V}}$. This approach, however, would not explicitly take into account the structure of multiview data in the minimisation. In the following we show how (3.15) is solved by imposing the camera setup and the occlusion constraints.

Imposing camera setup and occlusion constraints

Recall that the background layer corresponds to the object with the largest depth (smallest disparity gradient). If the boundary of this layer increases, it will automatically be occluded by the remaining layers in the dataset. Therefore, the structure of the visible layers will remain unchanged, and hence the cost must also remain the same.

When evolving the k-th layer, we model this by using the following indicator function:

$$\mathcal{I}_k(\mathbf{x}) = \begin{cases} 0, \text{ if } \mathbf{x} \in \left(\bigcup_{j=1}^{k-1} \mathcal{H}_j \right) \\ 1, \text{ otherwise,} \end{cases} \tag{3.16}$$

where the layers $\{\mathcal{H}_1, \ldots, \mathcal{H}_L\}$ are ordered in terms of increasing depth. Incorporating this into (3.14) allows the cost to be expressed in terms of $\mathcal{H}_k$ as follows:

$$J_k = \int_{\mathcal{H}_k} d_k(\mathbf{x}, \Delta p_k) \mathcal{I}_k(\mathbf{x})\, d\mathbf{x} + \int_{\overline{\mathcal{H}_k}} d_k^{out}(\mathbf{x}) \mathcal{I}_k(\mathbf{x})\, d\mathbf{x}. \tag{3.17}$$

Observe that the integration bounds $\mathcal{H}_k$ now correctly correspond to the layer boundary Γ_k. This, therefore, allows the derivative of the cost to be defined as

$$\frac{dJ_k}{d\tau} = \int_{\Gamma_k} \left[d_k^{out}(\mathbf{x}) \mathcal{I}_k(\mathbf{x}) - d_k(\mathbf{x}, \Delta p_k) \mathcal{I}_k(\mathbf{x}) \right] (\mathbf{v}_{\Gamma_k} \cdot \mathbf{n}_{\Gamma_k})\, d\boldsymbol{\sigma}. \tag{3.18}$$

Note that the derivative of the cost function also includes an additive term which corresponds to the change in the descriptor with respect to τ [43]. To simplify the expression we assume this term is zero.

Additionally, recall that using the camera setup constraint, the boundary Γ_k can be parameterised by a 2D contour $\gamma_k(s) = [x(s), y(s)]$ on the image viewpoint ($V_x = 0$).

Substituting this into (3.18) we obtain

$$\frac{dJ_k}{d\tau} = \int_{\gamma_k} \left[D_k^{out}(s) - D_k(s, \Delta p_k)\right] \left(\mathbf{v}_{\gamma_k} \cdot \mathbf{n}_{\gamma_k}\right) ds, \tag{3.19}$$

where $\mathbf{v}_{\gamma_k}$ and $\mathbf{n}_{\gamma_k}$ now correspond to the velocity and the outward normal vector of the 2D boundary. In addition, the new objective functions $D_k(\cdot)$ and $D_k^{out}(\cdot)$ are defined as

$$D_k(s, \Delta p_k) = \int\int d_k(\mathbf{x}, \Delta p_k)\, \mathcal{I}_k(\mathbf{x})\, dV_x dV_y \tag{3.20}$$

$$D_k^{out}(s) = \int\int d_k^{out}(\mathbf{x})\, \mathcal{I}_k(\mathbf{x})\, dV_x dV_y, \tag{3.21}$$

where

$$\mathbf{x} = \begin{bmatrix} x(s) - \Delta p_k V_x \\ y(s) - \Delta p_k V_y \\ V_x \\ V_y \end{bmatrix}.$$

Note that the new descriptors $D_k^{out}(\cdot)$ and $D_k(\cdot)$ are simply the descriptors $d_k^{out}(\cdot)$ and $d_k(\cdot)$ integrated over the viewpoint dimensions.

The velocity vector which reduces the cost in the direction of steepest descent can therefore be chosen as[4]

$$\mathbf{v}_{\gamma_k} = \left[D_k(s, \Delta p_k) - D_k^{out}(s)\right] \mathbf{n}_{\gamma_k}. \tag{3.22}$$

There are two main advantages in simplifying the evolution from a 4D to a 2D contour. First, the approach ensures that the layer boundary remains consistent across the views. Secondly, the complexity is reduced from evolving a 4D hypersurface to a 2D contour. We show a comparison between an unconstrained and constrained boundary evolution in Fig. 3.3. Observe that by imposing the camera setup and occlusion constraints in Fig. 3.3(b) we obtain a segmentation which is consistent with the EPI

[4]Note that in practise we also include a regularisation term to constrain the evolution according to the curvature of the boundary.

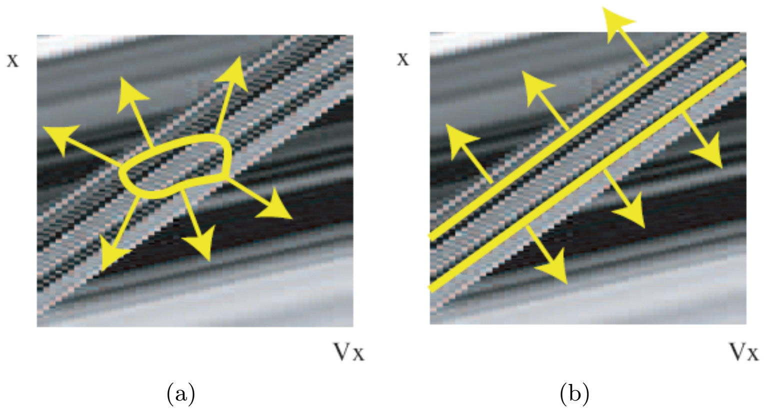


Figure 3.3: 2D EPI volume cross section showing unconstrained and constrained boundary evolution [6]. (a) Unconstrained boundary evolution. (b) Constrained boundary evolution. The segmentation is defined using a contour $\gamma(s)$ on image viewpoint $(V_x = 0)$ and a disparity gradient Δp.

structure. In summary, (3.22) defines a velocity vector, which evolves the layer boundary $\gamma_k(s)$ towards the desired segmentation for each layer.

Disparity Gradient and Number of Layers Estimation

In the previous section we presented an approach to derive the velocity vector for each layer. However, the knowledge of the disparities is required in order to evaluate the correct evolution. We evaluate these parameters by assuming the 2D layer contours $\{\gamma_1, \ldots, \gamma_L\}$ are constant. In this case, the objective function can be simplified to:

$$\min_{\{\Delta p_1, \ldots, \Delta p_L\}} \sum_{k=1}^{L} \int_{\mathcal{H}_k^{\mathcal{V}}} d_k\left(\mathbf{x}, \Delta p_k\right) d\mathbf{x}. \tag{3.23}$$

In contrast to the optimisation of the layer contours, this problem is significantly simpler. A solution can be obtained in an iterative approach by estimating the disparity gradient of each layer assuming the remaining disparity gradients are constant. For each disparity gradient, we find the value which minimises (3.23). The parameter is chosen by using the *fminsearch* function in MATLAB, where (3.23) is empirically evaluated during optimisation.

In addition, observe that we require the knowledge of the number of layers L. In our approach we initialise this value using a stereo matching algorithm [48]. Alternatively,

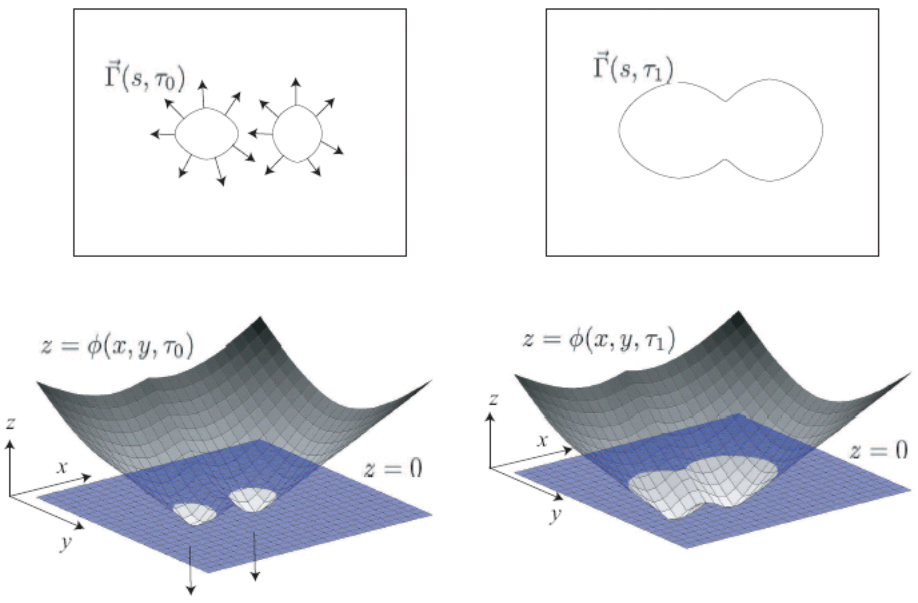


Figure 3.4: The level-set method models the 2D boundary as the zero-*th* level-set of a 3D surface. As the surface evolves, the boundary can implicitly change its topology [6].

one could estimate the number of layers using the spectral properties of the light field [15] as proposed in [9].

Level-set method for the boundary evolution

We defined a 2D velocity vector for each layer in Section 3.2.2. In practise, the evolution is implemented on a number of discrete points on the boundary. However, this approach is limited when the boundary needs to experience topological changes, such as splitting or merging. Although ad-hoc schemes can be used to solve this problem, they are not efficient.

In our approach we deal with this problem by modeling the boundary of each layer using the level-set method [78]. This method models the 2D boundary by a 3D surface. Given the surface, the 2D boundary is implicitly defined by an intersection of the surface with the xy-plane (as shown in Fig. 3.4). In this setup, as surface evolves the intersection with xy-plane can split or merge, and no ad-hoc schemes are required.

The original velocity vector only corresponds to the 2D boundary. To evolve the surface we must also derive the corresponding surface velocity vector. We define the higher dimensional surface with $z = \phi(x, y, \tau)$. The original boundary is then obtained as the zero-level of the new function

$$\begin{aligned} \gamma(s, \tau) &= \arg\ \{\phi(x, y, \tau)\} \\ &\quad \text{such that } \phi(x, y, \tau) = 0, \end{aligned} \tag{3.24}$$

where s parameterises the (x, y) coordinates.

The surface velocity vector is derived by applying two conditions which must be satisfied. The first is that if you move along the 2D boundary (increment s), we must remain on the zero-*th* level-set of the surface. This implies that the partial derivative of the surface with the respect to s must be zero. Therefore

$$\begin{aligned} &\frac{\partial \phi(x, y, \tau)}{\partial s} = 0 \\ \therefore\quad &\frac{\partial \phi}{\partial x}\frac{\partial x}{\partial s} + \frac{\partial \phi}{\partial y}\frac{\partial y}{\partial s} = 0. \end{aligned}$$

Rewriting this in terms of the dot product we obtain

$$\nabla \phi \cdot \frac{\partial \gamma}{\partial s} = 0, \tag{3.25}$$

where $\nabla \phi = \left[\frac{\partial \phi}{\partial x}, \frac{\partial \phi}{\partial y}\right]$ is the gradient operator on the surface ϕ and $\frac{\partial \gamma}{\partial s} = \left[\frac{\partial x}{\partial s}, \frac{\partial y}{\partial s}\right]$ is a tangent vector on the original 2D boundary (as shown in Fig. 3.5).

From (3.25) we observe that at each point on the boundary, the tangent $\frac{\partial \gamma}{\partial s}$ must be orthogonal to the gradient $\nabla \phi$ on the surface. This implies that $-\nabla \phi$ must be in the same direction as the inward normal vector $\mathbf{n}$ of the original 2D boundary:

$$\mathbf{n} = -\frac{\nabla \phi}{|\nabla \phi|}, \tag{3.26}$$

where we divide by $|\nabla \phi|$ to obtain a unit vector.

When the surface is evolved, the zero-*th* level-set must always be the same as the original 2D boundary evolved with the same velocity vector. This implies that the property $\phi(x(\tau, s), y(\tau, s), \tau) = 0$ must always be satisfied. Differentiating both sides

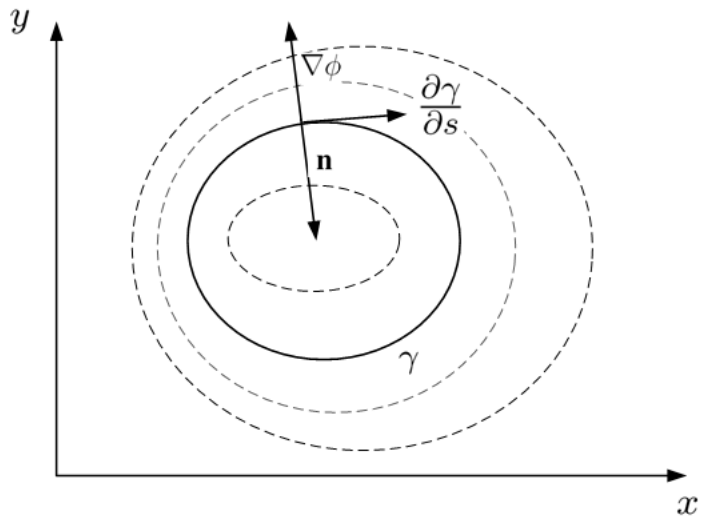


Figure 3.5: Level-sets of a surface. The original boundary γ defined by the solid line is the zero-*th* level-set. Observe that the surface gradient is orthogonal to the tangent vector on the 2D boundary.

with respect to τ we obtain the second condition:

$$\frac{\partial\phi}{\partial\tau}+\frac{\partial\phi}{\partial x}\frac{\partial x}{\partial\tau}+\frac{\partial\phi}{\partial y}\frac{\partial y}{\partial\tau}=0$$
$$\therefore\quad \frac{\partial\phi}{\partial\tau}+\nabla\phi\cdot\mathbf{v}=0, \tag{3.27}$$

where $\mathbf{v}=\left[\frac{\partial x(s)}{\partial\tau},\frac{\partial y(s)}{\partial\tau}\right]$ corresponds to the velocity vector of the original 2D boundary. The intuition here is that the partial change of x and y with respect to τ, defines the evolution of that point with respect to time. By definition, this is the same as the velocity vector.

Combining (3.27) and (3.26) we obtain the level-set evolution equation [78]

$$\frac{\partial\phi\left(x,y,\tau\right)}{\partial\tau}=F\left(x,y\right)|\nabla\phi|. \tag{3.28}$$

This implies that the evolution of the surface is also modeled by a PDE and in comparison to the original evolution of the boundary there is an additional scaling factor $|\nabla\phi|$. The force, $F\left(x,y\right)$ is then obtained in the same way as in the region competition to give [6]

$$\frac{\partial \phi\left(x,y,\tau\right)}{\partial \tau} = \left[D_k\left(x,y,\Delta p_k\right) - D_k^{out}\left(x,y\right) + \eta \kappa_\phi\left(x,y\right)\right] |\nabla \phi|, \tag{3.29}$$

where $\kappa_\phi\left(x,y\right)$ now corresponds to the curvature of the surface.

We can verify that this equation is intuitively correct. Consider that the force F is positive everywhere on the surface. Due to (3.28), this would have an effect of 'lifting' the whole surface and hence causing zero-*th* level-set to shrink. Therefore, this corresponds to the original 2D velocity vector, where a positive force would act to reduce the size of the 2D boundary.

Although the level-set method effectively models topological changes, a drawback in the approach is an increase in complexity. To evolve the surface, the velocity must be evaluated at each point on the surface. In our approach, we deal with this problem by using the narrowband implementation [42], where only a region around the boundary is evolved to reduce the complexity.

Layer segmentation algorithm overview

An overview of the complete layer extraction method is shown in Algorithm 1. First, the 2D contours and the disparity gradient of each layer are initialised using a stereo matching algorithm [48]. The algorithm evaluates the disparity gradient of each layer and then iteratively evolves the boundaries[5] using the velocity vector in (3.22). This process continues for a certain number of iterations or until the change in the overall cost is below a predefined threshold. An example of the extracted layers is shown in Fig. 2.7. In addition, in Fig. 3.6 we show a comparison between an initialised layer boundary using a stereo matching algorithm and the obtained layer contour.

[5]Note that the background layer $\mathcal{H}_L^{\mathcal{V}}$ is automatically assigned all of the regions which do not belong to the remaining layers and is therefore not evolved.

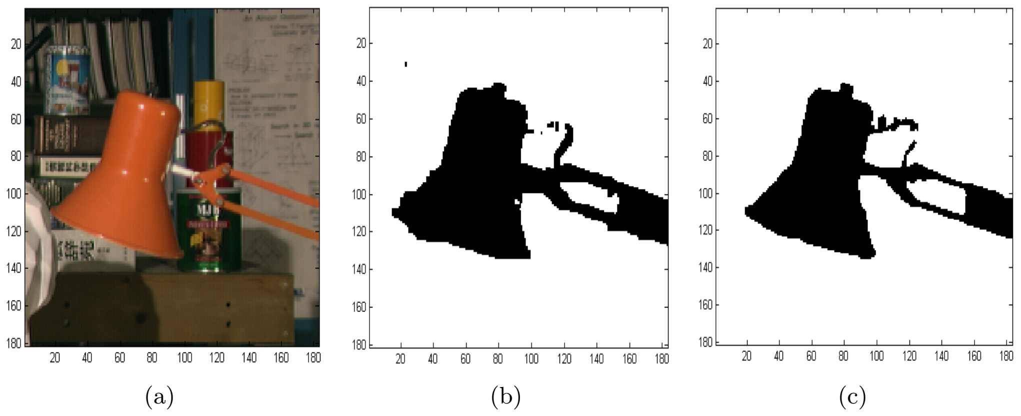


Figure 3.6: (a) Tsukuba dataset. (b) Initialised layer contour using a stereo matching algorithm [48]. (c) Layer contour after running Algorithm 1. The extraction algorithm improves the accuracy of the layer-based representation.

Algorithm 1 Layer extraction algorithm

```
  STEP 1: Initialise the 2D boundary of each layer $\{\gamma_1, \gamma_2, \ldots, \gamma_L\}$ using a stereo matching algorithm (Algorithm [48] in our implementation).
  STEP 2: Estimate the disparity gradient of each layer $\{\Delta p_1, \Delta p_2, \ldots, \Delta p_L\}$ by minimizing the squared error along the EPI lines.
  STEP 3: Reorder the layers in terms of increasing depth.
  STEP 4: Iteratively evolve the layer boundaries assuming the remaining layers are constant:
  for $k = 1$ to L-1 do
    Evaluate the velocity vector $\mathbf{v}_{\gamma_k}$ of the $k$-th layer.
    Evolve the boundary $\gamma_k$ according to the velocity vector.
  end for
  STEP 5: Return to STEP 2 or exit algorithm if the reduction in the cost (3.10) is below a predefined threshold.
```

3.3 Data Decomposition

The extracted multiview image layers shown in Fig. 2.7 are highly redundant in the direction of the disparity gradient. We also explicitly know the locations of any occluded regions in each layer. Here, we outline an approach to obtain a sparse representation which takes this information into account. The representation is obtained by applying a multi-dimensional DWT in a separable fashion across the inter-view and the spatial coordinates of each layer. To maximise the sparsity, the inter-view DWT is applied in the direction of the disparity gradient, and we modify the transform when filtering across occlusions. Next we describe the inter-view and spatial transforms in more detail.

3.3.1 Inter-view 2D DWT

The input to this stage are the extracted layers. Recall that in the case of a light field, each extracted layer consists of a 2D array of images. Only the regions that belong to the layer are visible (the rest of the pixels are set to zero). Thus, we implement the inter-view 2D DWT on each layer in two steps: first by applying a 1D disparity compensated DWT across the row images (V_y) followed by the column images (V_x) as illustrated in Fig. 3.7. The process is iterated on the low-pass components to obtain a multiresolution decomposition.

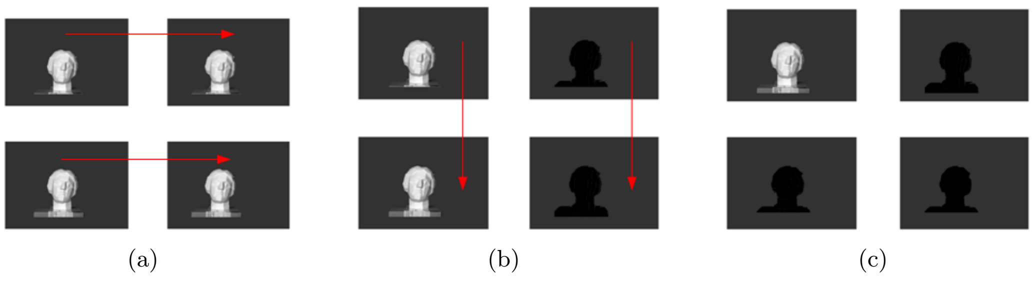


Figure 3.7: Inter-view 2D DWT implemented in a separable approach by filtering the image rows (V_y) followed by the image columns (V_x) using the 1D disparity compensated DWT. (a) Extracted layer: 2×2 light field. (b) Transform coefficients following 1D disparity compensated DWT across each row. (c) Transform coefficients following 1D disparity compensated DWT across each column. Note that the background has been labeled grey and is outside the boundary of the layer.

In our implementation of the inter-view 1D DWT we use the disparity compensated Haar transform. This is motivated by the fact that the light field intensity along the EPI lines is approximately constant. Therefore, a wavelet with one vanishing moment is theoretically enough to set the wavelet coefficients to zero [61]. The transform is applied by modifying the standard lifting equations [27] and including a warping operator $\mathcal{W}$ as follows:

$$\mathscr{L}_o[n] = \frac{P_o[n] - \mathcal{W}\{P_e[n]\}}{2}, \tag{3.30}$$

$$\mathscr{L}_e[n] = P_e[n] + \mathcal{W}\{\mathscr{L}_o[n]\}, \tag{3.31}$$

where, $P_o[n]$ and $P_e[n]$ represent 2D images with spatial coordinates (x, y) located at odd $(2n+1)$ and even $(2n)$ camera locations, respectively. Following (3.30) and

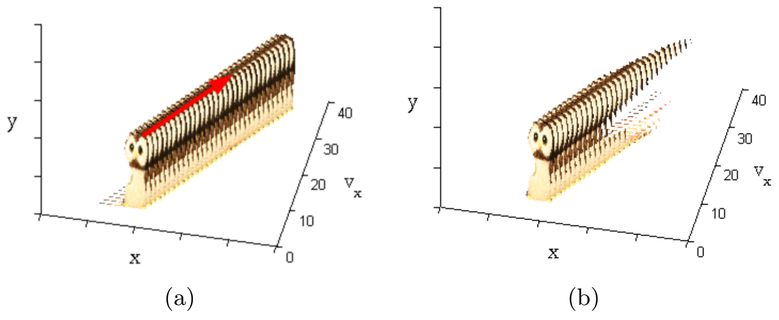


Figure 3.8: (a) By introducing the warping operator $\mathcal{W}$ we can apply the Haar transform in the direction of the disparity gradient denoted by the red arrow. Using this approach, the filtering is applied along the EPI lines, where the pixel intensity is approximately constant. This results in the majority of the wavelet coefficients to be approximately zero. (b) Extracted layers contain occluded regions. Filtering across the artificial boundary will create high pass coefficients and reduce the sparsity efficiency. We modify the inter-view DWT to be shape-adaptive in order to take occlusions into account.

(3.31), $\mathscr{L}_e[n]$ contains the 2D low-pass subband and $\mathscr{L}_o[n]$ the high-pass subband. Assuming that $\mathcal{W}$ is invertible and the images are spatially continuous, the above transform can be shown to be equivalent to the standard DWT applied along the motion trajectories [75]. This concept is further illustrated in Fig. 3.8(a), which shows that by effectively choosing the warping operator $\mathcal{W}$, the Haar transform is applied in the direction of disparity gradient.

In both the prediction (3.30) and update steps (3.31), we choose warping operator $\mathcal{W}$ to maximize the inter-image correlation. This is achieved by using a projective operation that maps one image onto the same viewpoint as its odd/even complement in the lifting step. Using (2.4) and the fact that the layers are modeled by a constant disparity gradient, we define the warping operation from viewpoint n_1 to n_2 along the V_x dimension as:

$$\mathcal{W}_{n_1 \to n_2}\{P[n_1]\}(x,y) = P[n_1](x + \Delta p(n_2 - n_1), y), \tag{3.32}$$

where Δp corresponds to the layer's disparity gradient.

Note that the projection operator $\mathcal{W}$ is not invertible if it corresponds to a non-integer pixel shift. A non-integer shift means that neighbouring pixels must be interpolated and this corresponds to low pass filtering. However, using the lifting scheme, we can still recover the original signal. Rearranging (3.30) and (3.31), the inverse transform is obtained using

$$P_e[n] = \mathscr{L}_e[n] - \mathcal{W}\{\mathscr{L}_o[n]\}, \tag{3.33}$$

$$P_o[n] = 2\mathscr{L}_o[n] + \mathcal{W}\{P_e[n]\}. \tag{3.34}$$

Note that the inverse transform in (3.33) and (3.34) does not require $\mathcal{W}^{-1}$, and this is a powerful result of the lifting scheme. It allows lossless signal reconstruction without the need to use invertible operators in the forward transform. In addition, a further advantage is that the lifting scheme has lower complexity than the traditional implementation [27].

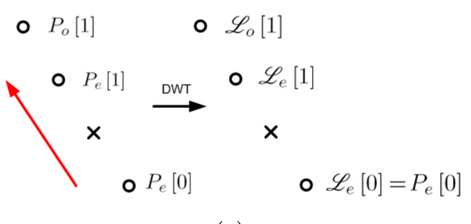
In the forward transform, we must also take into account occlusions. Recall, that the segmentation of the layers is known and thus we explicitly know the occluded region locations (we show an occluded layer in Fig. 3.8(b)). If the transform is not modified, the DWT leads to filtering across an artificial boundary and thus results in a reduced sparsity efficiency. To prevent this, we use the concept proposed in [53] to create a 'shape-adaptive' transform in the view domain. The transform in (3.30) and (3.31) is modified whenever a pixel at an even or odd location is occluded such that

$$\mathscr{L}_e[n] = \begin{cases} P_e[n], & \text{occlusion at } 2n+1 \\ \widehat{\mathcal{W}}\{P_o[n]\}, & \text{occlusion at } 2n \end{cases}, \tag{3.35}$$

and the high pass coefficient in $\mathcal{L}_o[n]$ is set to zero. In (3.35), the warping operator $\widehat{\mathcal{W}}$ is set to an integer pixel precision to ensure invertibility and is set to be the *ceiling* of the disparity in (3.32). This concept is further illustrated in Fig. 3.9, where we demonstrate our approach if a pixel at an odd or even image location is occluded.

Note also that the Haar transform in (3.30) and (3.31) is not orthonormal. Orthonomality plays an important role, since it implies that the squared error in the transform domain is equal to the error in the image domain (this will be further discussed in the evaluation section). In order to approach orthonomality we re-scale the coefficients by the length[6] of the basis functions [85].

[6]By length we imply the ℓ_2 vector norm.



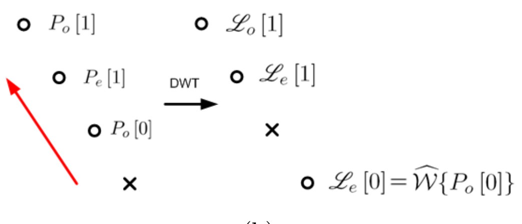


Figure 3.9: Circles correspond to the pixels which lie along an EPI line. The disparity direction is denoted by the red arrow. The occluded pixels in the EPI line are denoted with a cross. In (a) the occluded pixel is at the odd image location. Thus, the low pass coefficient $\mathscr{L}_e[0]$ is set to $P_e[0]$ and the high pass is zero. In (b), the occluded pixel is at the even image location. The low pass is set to $\widehat{\mathcal{W}}\{P_o[0]\}$ and the high pass is again set to zero.

3.3.2 Spatial shape-adaptive 2D DWT

Following the inter-view transform we reduce the spatial redundancy using a 2D DWT. However, prior to applying the 2D DWT on each image, we recombine the transform coefficients into a single layer. This is done to increase the number of decompositions which can be applied by the spatial transform. A comparison between the original and recombined layers is illustrated in Fig. 3.10. Note that due to occlusions and the way in which the inter-view transform is implemented, two or more layers may overlap in each subband. When two layers overlap, the transform coefficients of the layer with the smallest disparity gradient are removed and transmitted separately. A spatial transform is then also applied to these coefficients. However, we emphasize that since the segmentation is known prior to encoding, no additional overhead bits are required for encoding the location and segmentation of these overlapping pixels.

The overlapped pixels are commonly bounded by an irregular (non-rectangular) shape. For this reason, the standard 2D DWT applied to the entire spatial domain is inefficient due to the boundary effect. We therefore use the shape-adaptive DWT [33,53] within arbitrarily shaped objects. The method reduces the magnitude of the high pass coefficients by symmetrically extending the texture whenever the wavelet filter is crossing the boundary. The 2D DWT is built as a separable transform with linear-phase symmetric wavelet filters (9/7 or 5/3 [87]), which, together with the symmetric signal extensions, leads to critically sampled transform subbands.

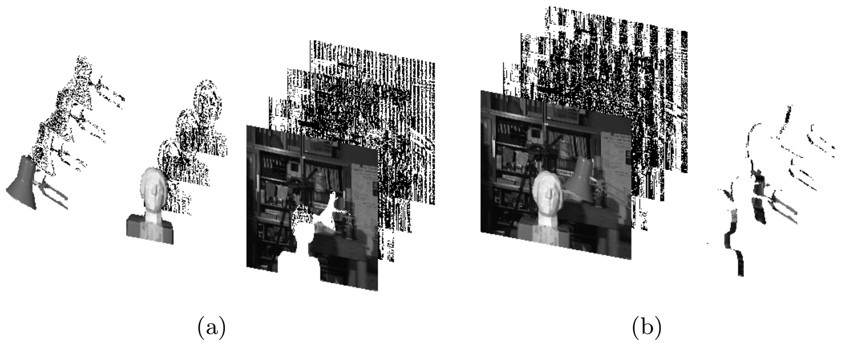


Figure 3.10: (a) Tsukuba transform coefficients following the inter-view transform. Each of the three transformed layers is composed of one low-pass subband and three high frequency images. (b) Recombined layers. The view subbands from each layer are grouped into a single image to increase the number of decompositions that can be applied by the spatial transform. In each subband two or more layers may overlap. We apply a separate shape-adaptive 2D DWT to the overlapped pixels.

3.4 Evaluation

In this section we evaluate the performance of the proposed sparse representation using its N-term nonlinear approximation (NLA) properties. In addition we present denoising results based on the decomposition.

3.4.1 N-term nonlinear approximation

To obtain the N-term NLA, we keep the N largest coefficients in the transform domain according to their magnitude and set the remaining ones to zero. The N-term NLA is then obtained by applying the inverse transform. We evaluate the performance of the decomposition by analysing the decay of the mean squared error (MSE) with respect to N. The faster the decay of the MSE, the sparser the representation.

The aim of NLA is to minimise the MSE error by selecting N coefficients in the transform domain. In fact, any combination of the coefficients can be used[7]. In practise it is not possible to evaluate the MSE for each possible combination. However, in the case of an orthonormal decomposition, the error in the transform domain directly corresponds to the error in the image domain [61]. We assume our decomposition to be approximately orthonormal and therefore, it suffices to select the N largest coefficients

[7]The total number of combinations is $\frac{M!}{N!(M-N)!}$, where M is the total number of transform coefficients.

to minimise the MSE.

Our results show that the proposed layer-based representation offers superior approximation properties when compared to a typical multi-dimensional DWT[8]. We demonstrate this in Fig. 3.11 on three datasets: Tsukuba light field [$272 \times 368 \times 4 \times 4$], Teddy EPI [$368 \times 352 \times 4$] and Doll EPI [$368 \times 352 \times 4$] (all from [74]), which vary in terms of scene complexity, number of images and spatial resolution. We show that in each case our approach achieves a sparser representation across the complete range of retained coefficients, with Peak Signal to Noise Ratio (PSNR) gains of up to 7dB on the Tsukuba light field. The Tsukuba light field has a larger PSNR improvement than the respective Teddy and Doll EPI volumes due to the additional viewing dimension. This means that there exists more redundant information and this is fully exploited by our representation. We also show that the PSNR curves correspond to a subjective improvement in Fig. 3.12, 3.13 and 3.14.

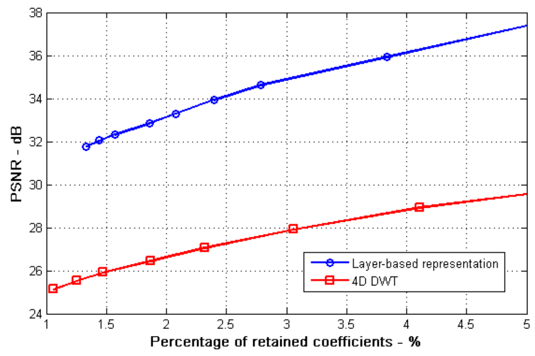
3.4.2 Denoising

Here we present denoising results based on the proposed sparse representation in the presence of additive white Gaussian noise (AWGN). Note that the aim of this section is not to compare the results to the state-of-the-art in multiview denoising techniques but to demonstrate that the sparse representation can be used for denoising applications.

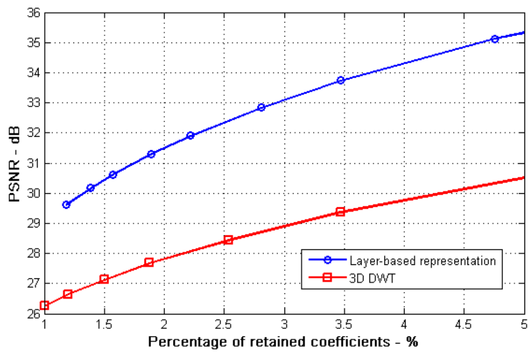
We implement the denoising by soft thresholding the wavelet coefficients in each subband (except the low-pass). As outlined in the introduction, thresholding reduces the noise components by assuming that the underlying signal is sparse in the decomposition. However, by thresholding we not only set the noise components to zero, but also the original signal[9]. This trade-off must be taken into account when selecting the threshold. If the original signal was available, the optimal threshold could be directly chosen by minimising the MSE of the reconstructed signal. However, this would defeat the purpose of denoising. We solve this problem by instead using the Stein's Unbiased Risk Estimate (SURE) of the MSE [31]. The advantages of this approach is that we do

[8]This multi-dimensional DWT has the same decomposition structure as our method, however no disparity compensation is applied.

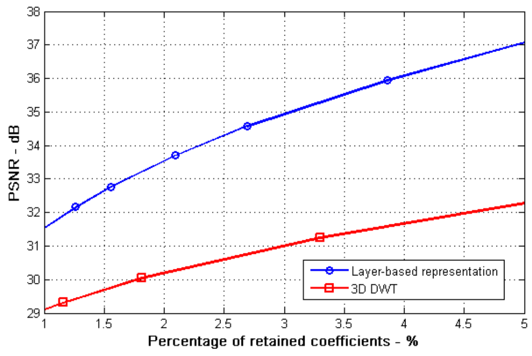
[9]Recall that the transform coefficients are not exactly zero in the sparse decomposition.



(a) Tsukuba light field.



(b) Teddy EPI.



(c) Doll EPI.

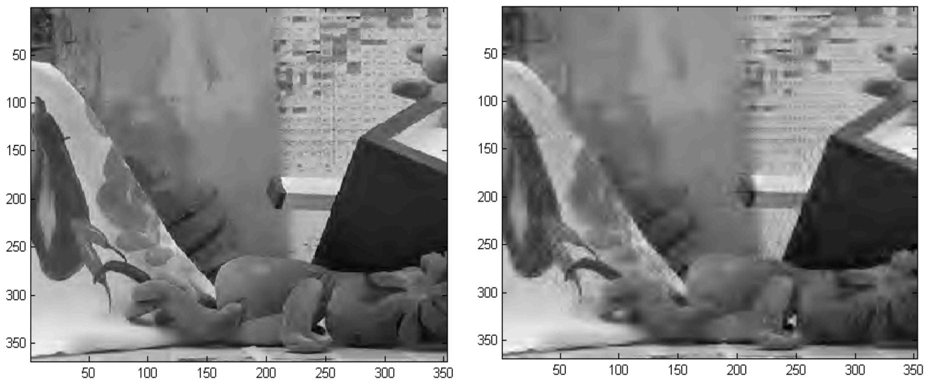
Figure 3.11: N-term NLA of the layer-based representation in comparison to a standard multi-dimensional DWT. Note that the percentage of retained coefficients is evaluated as $\frac{N}{M} \times 100$, where N is the number of retained coefficients and M is the number of coefficients in the original dataset.

not need to know the original signal or its statistics. In fact, the estimate is obtained by using only the corrupted signal and the standard deviation of the noise σ [31].

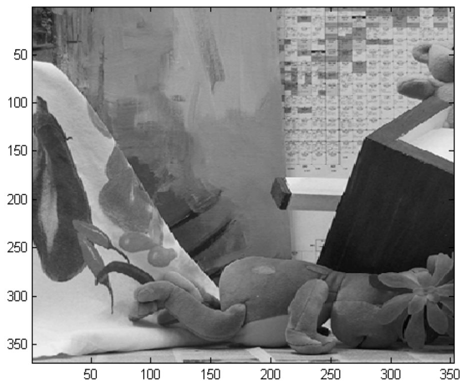
We consider two scenarios for our proposed method: the case when the layers are extracted from the noise-corrupted images and the case when they are extracted from the original dataset and the noise is added after the extraction. Note that in the presence of noise, the intensity along each EPI line is no longer constant, and this can introduce errors in the segmentation stage. In our case, we deal with this problem by denoising each image independently using [10] prior to extracting the layers. This removes significant amounts of noise and improves the layer extraction accuracy. Following the segmentation stage, we perform the denoising on the original noise corrupted images[10].

In Figs. 3.15 and 3.16 we show that our approach outperforms the competitive

[10]Independent denoising removes signal information which can be recovered by our method. Therefore, better results are achieved by denoising the original dataset.



(a) Proposed method - PSNR 29.62dB with 1.18% of coefficients retained. (b) Standard 3D DWT - PSNR 26.6dB with 1.20% of coefficients retained.

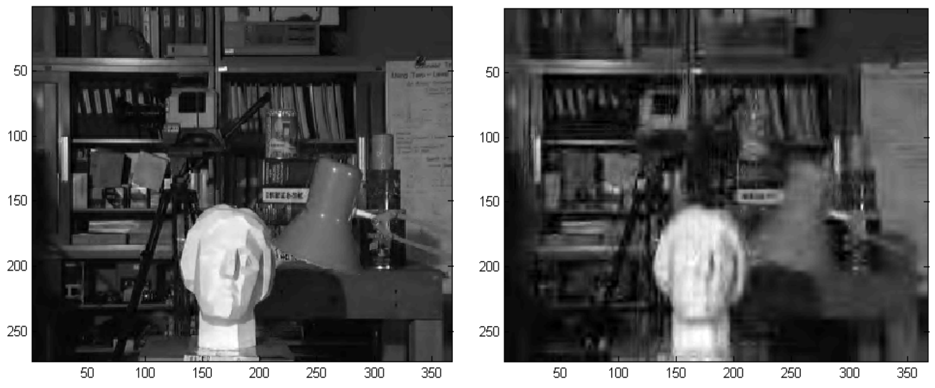


(c) Original dataset.

Figure 3.12: *N*-term NLA comparison between (a) sparse layer-based representation and (b) standard 3D DWT on Teddy EPI. (c) Original Teddy EPI dataset.

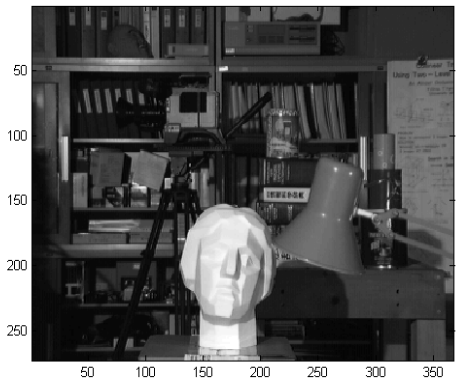
SURE-LET OWT denoising method [10]. We use Teddy EPI [$368 \times 352 \times 4$] and Doll EPI [$544 \times 608 \times 4$] datasets (both from [74]) in the simulation. In the case when the layers are extracted on the original dataset our method outperforms SURE-LET OWT by up to 0.9dB. It is evident that extracting the layers in the presense of noise incurs a small performance loss in comparison to the original layers. However, in this setup our method still outperforms SURE-LET OWT denoising.

The subjective results are illustrated in Figs. 3.17 and 3.18. These clearly show that the proposed sparse representation attains more visually pleasing results than the SURE-LET OWT method.



(a) Proposed method - PSNR 31.77dB with 1.33% of coefficients retained.

(b) Standard 3D DWT - PSNR 25.91dB with 1.47% of coefficients retained.



(c) Original dataset.

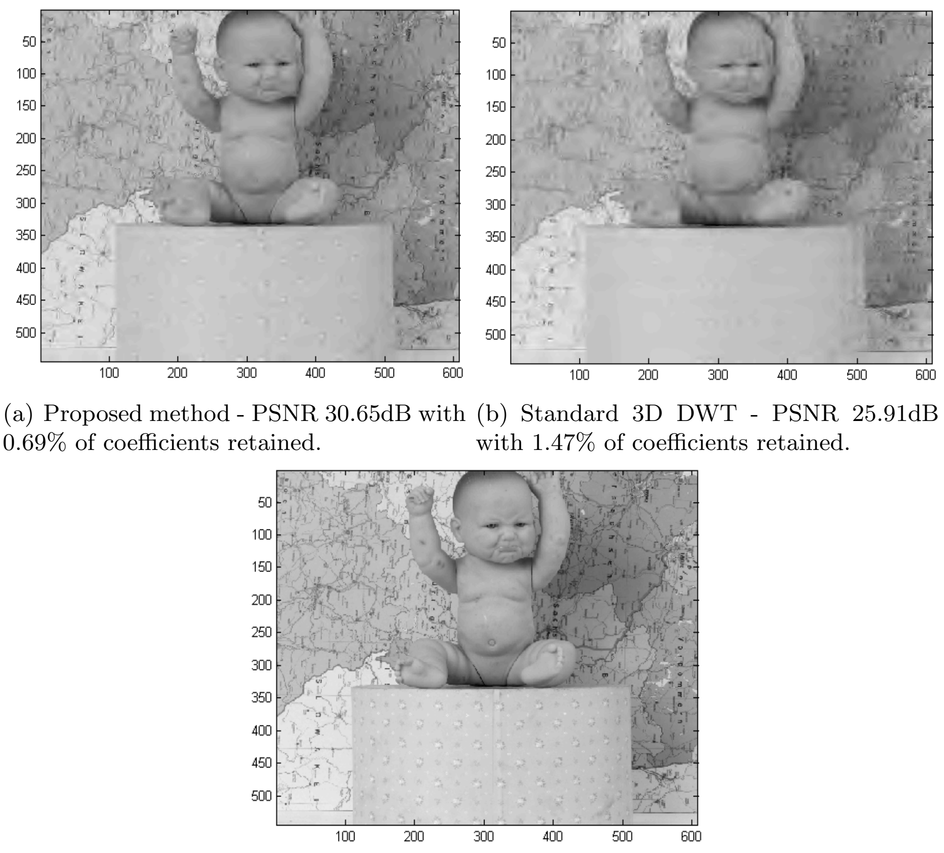
Figure 3.13: N-term NLA comparison between (a) sparse layer-based representation and (b) standard 4D DWT on Tsukuba Light Field. (c) Original Tsukuba light field dataset.

3.5 Summary

In this chapter we outlined an approach to obtain a sparse decomposition of multiview images. The algorithm is composed of two main stages. Firstly, we segment the multiview images into a set of layers, where each one is related to a constant depth in the scene. Each layer contains EPI lines corresponding to the same depth and thus they are redundant in the direction of the disparity gradient.

We outlined the layer extraction algorithm based on the work in [8]. The method segments the layers by minimising the total variance along the EPI lines. Also the structure of multiview images is taken into account to achieve more accurate results and the evolution of each layer is simplified from a surface to a 2D contour.

In the second stage we remove the redundancy in the segmented layers. The layers



(a) Proposed method - PSNR 30.65dB with 0.69% of coefficients retained.

(b) Standard 3D DWT - PSNR 25.91dB with 1.47% of coefficients retained.

(c) Original dataset.

Figure 3.14: *N*-term NLA comparison between (a) sparse layer-based representation and (b) standard 3D DWT on Doll EPI. (c) Original Doll EPI dataset.

are decomposed using a 4D DWT applied in a separable fashion. First we apply a 2D DWT along the camera viewpoint dimensions followed by the spatial coordinates. We modify the viewpoint transform to efficiently deal with occlusions and depth variations. Following the inter-view DWT we recombine the layer to increase the number of decompositions which can be applied by the spatial transform.

Simulation results based on NLA have shown that the sparsity of our representation is superior to a multi-dimensional DWT with the same decomposition structure, but without disparity compensation. In addition, we have shown that the sparse representation can be used for denoising. We considered two scenarios: (a) the layers are extracted on the noisy images and (b) from the original images. Although, extracting the layers on the noisy dataset incurs a small loss in performance, we showed that both schemes outperform SURE-LET OWT [10] applied to each image independently.

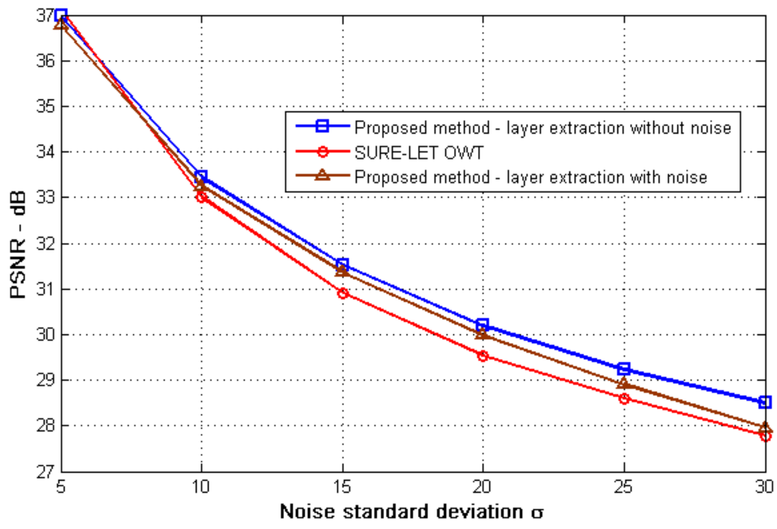


Figure 3.15: Denoising evaluation on Teddy EPI. We consider two scenarios: the setup when the layer extraction is implemented on the original dataset (without noise) and when the layers are extracted using the noisy images. In the latter case, we denoise the images independently prior to extracting the layers. Our results show that extracting the layers in the presence of noise incurs a loss in performance. However, our method still outperforms SURE-LET OWT.

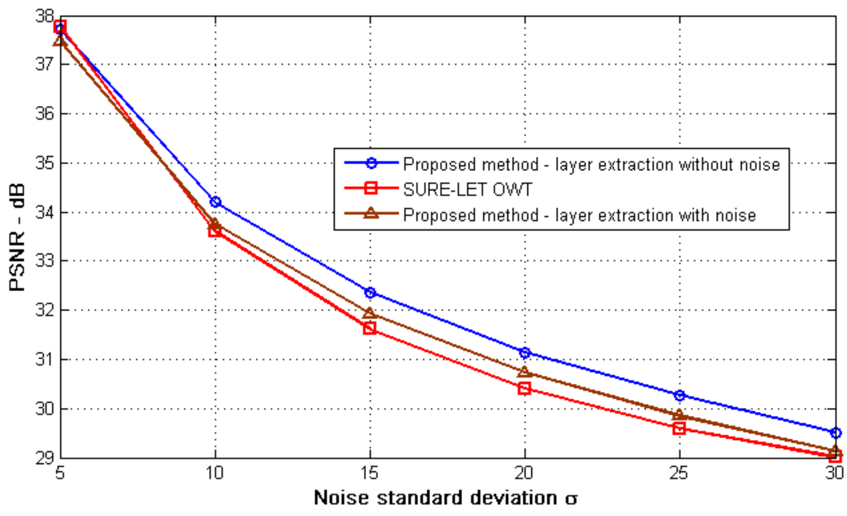
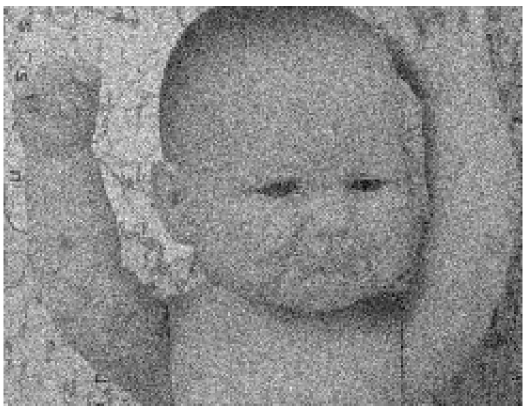
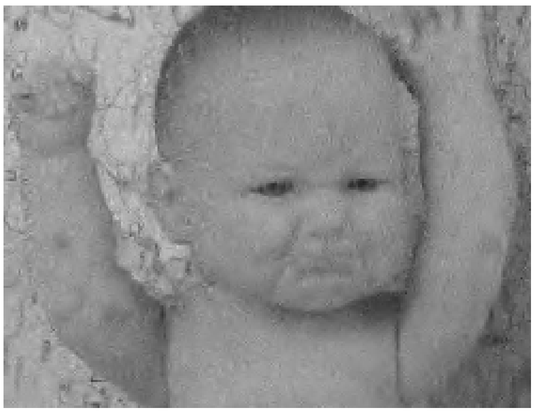
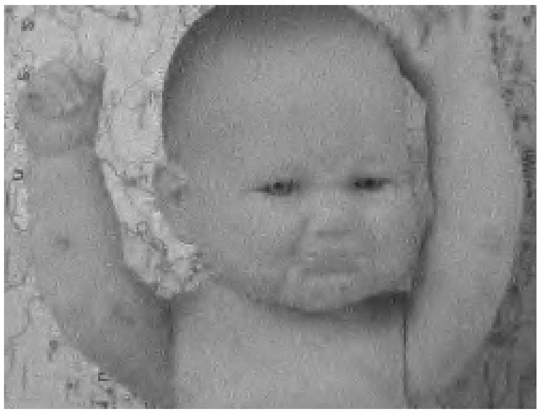


Figure 3.16: Denoising evaluation on Doll EPI. We consider two scenarios: the setup when the layer extraction is implemented on the original dataset (without noise) and when the layer contours are extracted on the noisy images.


(a) Noisy Input (PSNR 22.11 dB).


(b) Proposed Method (PSNR 30.70 dB).


(c) Proposed Method Oracle Layer Extraction (PSNR 31.12 dB).

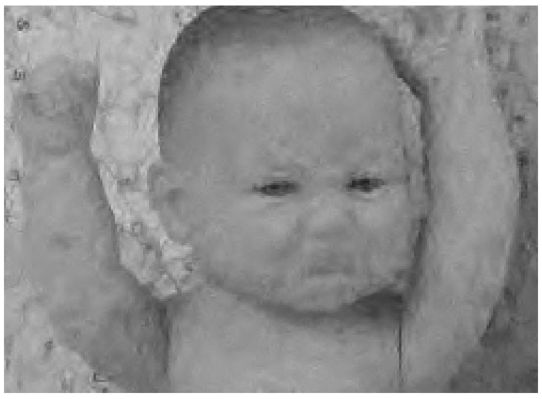
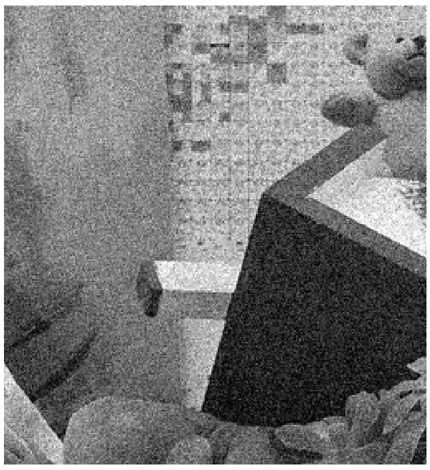
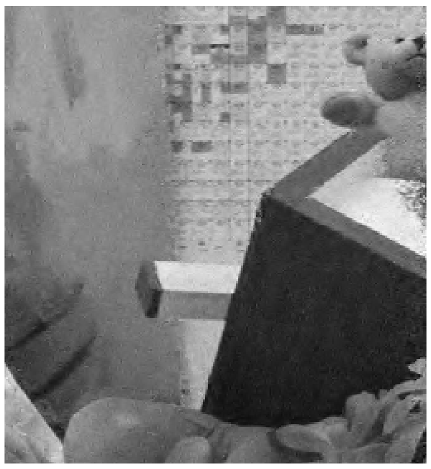
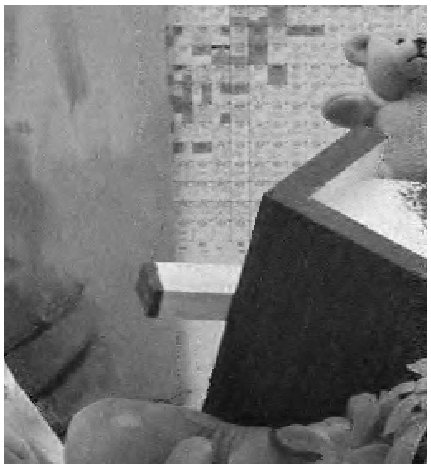

(d) SURE-LET OWT (PSNR 30.39 dB).

Figure 3.17: Subjective evaluation of the proposed denoising method on Doll EPI. (a) Input dataset with AWGN. (b) Denoised image using proposed method (layers extracted using noisy images). (c) Denoised image using proposed method (layers extracted using original dataset). (c) SURE-LET OWT applied to each image independently.

As discussed in this chapter, NLA is a good indicator for coding performance. However, a sparse decomposition is not a sufficient condition. To achieve a good RD performance, we must also correctly allocate the bit budget among the transform coefficients. In addition, we must encode the locations of the transmitted coefficients in order to correctly reconstruct the data. In the following chapter we discuss these issues in more detail.


(a) Noisy Input (PSNR 22.11 dB).


(b) Proposed Method (PSNR 29.89 dB).


(c) Proposed Method Oracle Layer Extraction (PSNR 30.26 dB).

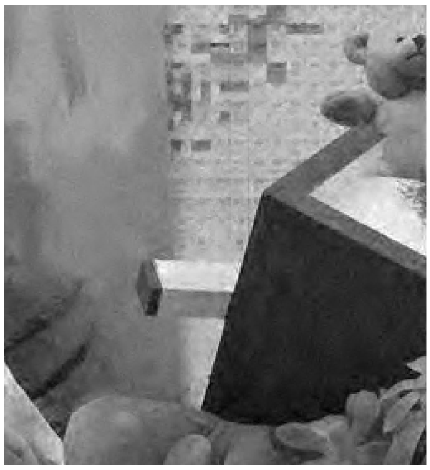

(d) SURE-LET OWT (PSNR 29.53 dB).

Figure 3.18: Subjective evaluation of the proposed denoising method on Teddy EPI. (a) Input dataset with AWGN. (b) Denoised image using proposed method (layers extracted using noisy images). (c) Denoised 'image using proposed method (layers extracted using original dataset). (c) SURE-LET OWT applied to each image independently.

Chapter 4

Multiview Image Compression

In Chapter 3 a layer-based decomposition of multiview images was proposed and it was shown that a small number of transform coefficients in the sparse decomposition give a good approximation of the original dataset. Here we present a multiview image compression method based on this decomposition.

A sparse representation by itself does not guarantee efficient compression in a RD sense. Recall, that in NLA the MSE was evaluated by selecting the N largest coefficients. To compress the data, we must also encode the selected coefficients and their locations. In addition, we must consider how the bit budget should be allocated among the transform coefficients. For example allocating bits to certain coefficients will reduce the MSE significantly more compared to others. Thus, to achieve an efficient compression, it is important to find the correct distribution which minimises the total MSE.

In the proposed method, we must transmit both the transform coefficients and the layers contours. The layer contours are required in order to apply the inverse inter-view and spatial DWT. We encode the contours by approximating them as piecewise linear segments. The transform coefficients are coded using conventional quantisation and entropy coding [36].

To ensure an efficient RD performance, we also address the problem of rate allocation between the layer contours and the transform coefficients. Given a target bit budget, the problem lies in finding the correct ratio which will minimise the output distortion. We show that a closed form solution can be obtained when the contours are piecewise linear. In the more general case of non-piecewise linear contours, the theoretical analysis is combined with an empirical approach to find the correct allocation.

This chapter is organised as follows. Next we present a high-level overview of the compression algorithm. In Section 4.2 we outline our method to encode the layer contours and the texture. We present the RD model to find the rate allocation in Section 4.3. The chapter is summarised in Section 4.4.

4.1 High-level compression algorithm overview

The high-level layout of the proposed compression algorithm is shown in Fig. 4.1. The input data is initially segmented to obtain the layer-based representation. Each layer is assigned a global disparity gradient, which is losslessly encoded and transmitted. The layers are then passed to the coding stage where a disparity compensated 4D DWT is applied to remove the redundancy. The transform coefficients at this stage directly correspond to the sparse representation outlined in Chapter 3. This is followed by quantisation and entropy coding of the transform coefficients. In order to reconstruct the texture at the decoder, the layer contours are also encoded in a lossy or lossless modality and transmitted.

Observe that for a given total bit budget, there exists a trade-off between the number of bits allocated to encode the layer contours and the texture. This problem is solved in the RD control stage, which prior to encoding, correctly distributes the bits between the texture and the layer contours to maximise the output fidelity of the images. The encoding method is also outlined in Algorithm 2.

4.2 Texture and Contour Encoding

In this section, we describe the contour and texture encoding blocks from Fig. 4.1. We denote with $\mathbf{R_s}$ and $\mathbf{R_x}$ the bits allocated for encoding the contours and texture,

Algorithm 2 Compression algorithm overview

```
STEP 1: Segment the multiview images to obtain the layer-based representation.
STEP 2: For a given bit budget $\mathbf{R_t}$, find the correct rate allocation to encode the layer contours $\mathbf{R_s}$ and texture $\mathbf{R_x}$ such that $\mathbf{R_x} + \mathbf{R_s} = \mathbf{R_t}$.
STEP 3: Given the rate constraint $\mathbf{R_s}$, encode the layer-contours in a lossy or lossless modality.
STEP 4: If the layer contours are encoded in a lossy modality, update the layer-based representation such that it corresponds to the encoded version.
STEP 5: Given the target bit rate $\mathbf{R_x}$ encode the texture in each layer.
```

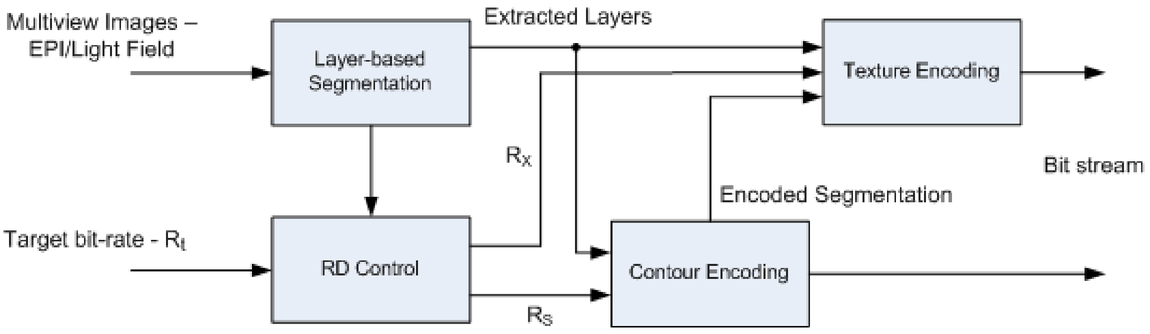


Figure 4.1: High-level algorithm layout. Layers are extracted in the layer-based segmentation block, and each layer is transformed using a 4D DWT in the texture coding stage. The layer contours are encoded in a lossy or lossless modality and transmitted to the decoder. The method includes a RD control stage, which correctly distributes the bit budget between the layer contours and the texture to maximise the output fidelity.

respectively.

4.2.1 Encoding of the contours

The contour encoding block transmits the segmentation needed to correctly decode each layer. Recall, from the properties of multiview data outlined in Section 2.3 that the segmentation of a 4D/3D layer can be defined by a contour on one of the image viewpoints and the layer's disparity gradient. Therefore, the input to the layer contour encoding algorithm is an array of binary images (one for each layer) and a target bit budget $\mathbf{R_s}$. A typical binary layer is illustrated in Fig. 4.2(a).

The 2D contours can be encoded in a lossless or lossy modality. The lossless method is used for $\mathbf{R_s}$ larger than the required lossless bit budget. The problem of lossless contour encoding has been extensively studied with the majority of the work based on Freeman chain codes [34]. Here we use the algorithm proposed in [54], which encodes the

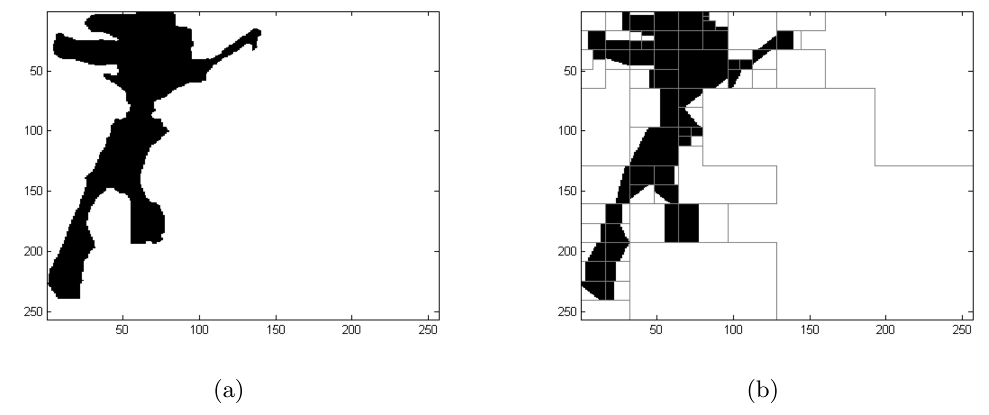


Figure 4.2: (a) Original layer contour from Tsukuba dataset [74]. (b) Quadtree prune-join piecewise linear approximation obtained using the algorithm proposed in [80].

contour as follows: Each pixel on the boundary is connected to a neighbouring vertex, where for a given pixel there are eight possible connections. A sequence of symbols is then obtained by evaluating the relative angle with respect to a neighbouring reference vertex. Assuming the original boundary is smooth, the symbol probabilities are non-uniform. A Huffman dictionary is hence constructed and used to entropy code the symbols.

In the case when the given $\mathbf{R_s}$ is below the lossless encoding rate, a piecewise linear approximation of the layer contours is used. We use a 2D quadtree model to capture and quantise these linear segments. In this representation each block of the quadtree is approximated by a $\{0, 1\}$ or intermediate tile. The intermediate tile is a binary image with an edge modelled by a linear function. These quadtree coefficients are also quantised and entropy coded. The type of tile for an arbitrary block is chosen by minimising the cost $D_p(R_p) + \lambda_s R_p$ subject to the rate constraint $\mathbf{R_s}$, where $D_p(R_p)$ and R_p are the distortion associated to the p-th tile and the corresponding encoding rate, respectively.

To encode the segmentation, we use a bottom-up approach where the quadtree is pruned whenever the sum of the cost of the leaves is higher than the cost of their parent. The merging process iteratively continues as long as the Lagrangian cost decreases or the complete image is represented by a single tile. In addition, a joining scheme is used as proposed in [80], where the neighbouring blocks that cannot be pruned within the quadtree representation are joined to improve the compression performance. Fig. 4.2 shows a comparison between an original layer contour and its quadtree prune-join representation.

The output rate of the coding algorithm is controlled by the parameter λ_s, which defines the trade-off between rate and distortion. To achieve the specified total target bit budget $\mathbf{R_s}$, a bisection search [79] for the correct λ_s is applied.

4.2.2 Texture quantisation and entropy coding

Following the 4D/3D DWT, the transform coefficients are quantised and entropy coded. Our codec uses context adaptive arithmetic coding [63, 93] to attain bit rates close to the entropy of the source.

For a given bit budget constraint $\mathbf{R_x}$, the optimal bit allocation among the transform coefficients is achieved using a method similar to EBCOT [85]. Initially, the coefficients are partitioned into blocks of size $[64 \times 64]$. Each block is losslessly encoded by bit-plane to obtain the operational RD curves using a context adaptive arithmetic coder. Then, given a Lagrangian multiplier λ_x, a bit allocation R^* for each block is chosen such that the cost function J is minimised, where

$$J = D\left(R_i\right) + \lambda_x R_i, \tag{4.1}$$

and $\left(R_i, D\left(R_i\right)\right)$ is the operational RD pair associated to each block. To meet the allowed bit budget $\sum\limits_l R_l^* = \mathbf{R_x}$ a bisection search [79] for the correct multiplier λ_x is applied. Here, R_l^* corresponds to the chosen bit allocation in the l-th block. We note that since the operational RD curves are known, this process is not computationally expensive. Once the optimal values R_l^* are evaluated, the encoded bit streams are truncated and transmitted.

4.3 Rate-distortion modeling and optimisation

We now study the correct distribution of the bit budget between encoding the layer contours and the DWT coefficients so that the overall RD performance is improved. The problem is defined as follows: given a target bit budget $\mathbf{R_t}$, the goal is to find an allocation which will minimise the output distortion:

$$\begin{aligned} [\mathbf{R_s}, \mathbf{R_x}] &= \text{argmin} \left\{ D\left(\mathbf{R_s}, \mathbf{R_x}\right) \right\} \\ & \quad \text{such that } \mathbf{R_s} + \mathbf{R_x} \leq \mathbf{R_t}, \end{aligned} \tag{4.2}$$

where $\mathbf{R_s}$ and $\mathbf{R_x}$ are the number of bits allocated to the layer contours and texture, respectively. The distortion is measured in terms of the sum of squared differences (SSD) between the input and the output.

Note that the distortion $D\left(\mathbf{R_s}, \mathbf{R_x}\right)$ is jointly dependent on the contour and texture encoding. A lossy contour encoding modifies the layer segmentation. In turn, this means that certain pixels will be allocated to an incorrect layer. During coding, these pixels will reduce the coding efficiency and will highly contribute to the texture distortion.

To solve this optimisation problem, we first upper bound the distortion due to the lossy encoding of the layer contours. Then, we approximate the total distortion as the sum of the distortion due to encoding the texture and the upper bound of the distortion of the layer contours. In the following, we derive the RD models for the layer contours and the texture.

4.3.1 RD Texture Modeling

To model the texture, we assume the obtained transform coefficients are independent Gaussian variables with the RD relation [25]:

$$D\left(R\right) = \sigma^2 2^{-2R}, \tag{4.3}$$

where σ^2 is the variance. Therefore, the total distortion due to encoding the texture is modeled by:

$$D_x(\mathbf{R_x}) = \sum_{j=1}^{L}\sum_{i=1}^{K_j} C_j N_{ij} \sigma_{ij}^2 2^{-2R_x^{ij}}, \tag{4.4}$$

where R_x^{ij} is the rate allocated to each transform coefficient, σ_{ij}^2 is the variance of the coefficients and N_{ij} is the number of transform coefficients in each subband. The subscript ij corresponds to the parameter of the i-th subband in the j-th layer. L denotes the total number of layers and K_j is the number of subbands in the j-th layer. The shape-adaptive feature of the inter-view DWT assumes that the complete layer is visible, and this can lead to an incorrect rate allocation. Therefore, we scale the distortion by a factor $0 < C_j \leq 1$ to model occlusions. It is defined as the number of pixels in the visible layer $\mathcal{H}_j^{\mathcal{V}}$ divided by the number of the pixels in the layer $\mathcal{H}_j$ (a layer obtained using a contour on one image viewpoint and a disparity gradient). Note that the parameters C_j and σ_{ij}^2 can be estimated using the original data and transform coefficients, respectively. Also recall that

$$\mathbf{R_x} = \sum_{j=1}^{L}\sum_{i=1}^{K_j} N_{ij} R_x^{ij}, \tag{4.5}$$

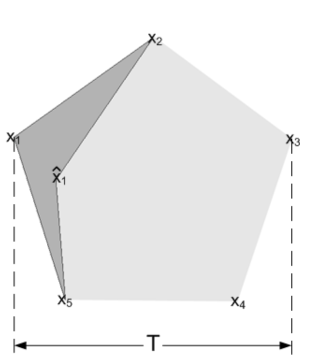
is the total bit budget allocated to the texture.

4.3.2 RD Layer Contour Modeling

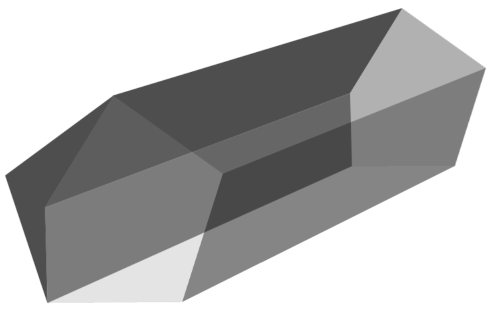
The operational RD function of the layer contours is modeled by computing the upper bound of the distortion for a given bit budget.

To obtain the RD bound, first, recall that the contour of a layer at each image viewpoint is constant unless occluded. Using this property, we upper bound the distortion on a 2D image and scale it to evaluate the distortion in a 3D/4D layer.

Consider a 2D slice of a layer $\mathcal{L}_1$ as shown in Fig. 4.3(a). Assume that the contour is piecewise linear and has a fixed number of vertices (also known as a polygonal model).



(a)



(b)

Figure 4.3: (a) 2D Slice of a layer with a piecewise constant segmentation. Here, x represents the original vertices and $\hat{x}$ the location of a quantised vertex. The shaded region denotes the error due to quantising the vertex x_1. (b) Shaded region outlines the error in a 3D layer, which is obtained by propagating the error from the first viewpoint.

The distortion can be upper bounded by quantizing the locations of the vertices:

$$e^2 \leq T^2 V \zeta^2 2^{-\frac{R_s}{2V}}, \tag{4.6}$$

where V is the number of vertices, T is the size of the bounding image, $\zeta = \max\{\mathcal{L}_1\} \leq 255$ is the maximal amplitude of the texture and R_s is the number of bits allocated to encode the contour. The derivation of this bound is shown in Appendix A.1. As illustrated in Fig. 4.3(b), the error which is denoted by the shaded region can be scaled by the total number of images to compute the distortion in a 3D layer. The total distortion due to encoding L layers can therefore be upper bounded by:

$$D_s(\mathbf{R_s}) \leq \sum_{j=1}^{L} T_j^2 V_j M \zeta_j^2 C_j 2^{-\frac{R_s^j}{2V_j}}, \tag{4.7}$$

where M is the total number of images and the subscript j denotes the parameter corresponding to the j-th layer. Furthermore,

$$\mathbf{R_s} = \sum_{j=1}^{L} R_s^j, \tag{4.8}$$

is the total encoding rate of the layer contours.

Note that the model in (4.7) is evaluated by upper bounding the number of coefficients affected by the quantisation of the layer contours. During coding, these pixels are associated to an incorrect layer and therefore create high-pass transform coefficients. In turn, this will add to the texture distortion when the coefficients are quantised. We model this distortion by scaling the number of pixels by ζ_j. In the worst case scenario, it is the maximal value of the texture pixels equal to 255. At high bit rate, however, observe that the rate allocation between the texture and the layer contours is independent of the scaling ζ_j and only determined by the exponent of the RD functions [25].

Note also that the analysed upper bound of the distortion in (4.7) does not rely on the quadtree algorithm used in Section 4.2.1. However, as shown in [80], the RD performance of the quadtree algorithm is accurately modeled by our analysis, especially at high bit rates. In the continuation, the derived RD relations are used to find the optimal bit budget distribution between the layer contours and texture encoding.

4.3.3 RD Optimisation - Piecewise linear layer contours

We use Lagrangian multipliers [83] to find the bit rate distribution between the layer contours and the texture. Consider approximating the total distortion as the summation of the texture (4.4) and layer contours distortion (4.7):

$$\begin{aligned} D\left(\mathbf{R_x},\mathbf{R_s}\right) &\sim D_x\left(\mathbf{R_x}\right)+D_s\left(\mathbf{R_s}\right) \\ &= \sum_{j=1}^{L} C_j \left(\sum_{i=1}^{K_j} N_{ij}\sigma_{ij}^2 2^{-2R_x^{ij}} + T_j^2 V_j M \zeta_j^2 2^{-\frac{R_s^j}{2V_j}} \right), \end{aligned} \tag{4.9}$$

where $\mathbf{R_x}$ and $\mathbf{R_s}$ are defined in (4.5) and (4.8), respectively. For a given bit budget $\mathbf{R_t}$, the encoding rate must satisfy the following inequality:

$$\mathbf{R_x}+\mathbf{R_s} \leq \mathbf{R_t}. \tag{4.10}$$

The above problem can be solved using the reverse water-filling algorithm [25] based on Lagrangian multipliers. Assuming a high rate analysis, which implies that all the rates are positive, we obtain the following solutions:

$$\mathbf{R_x} = \sum_{j=1}^{L} \sum_{i=1}^{K_j} N_{ij} \left(\frac{1}{2} \log_2 \left[2 \ln(2)\, C_j \sigma_{ij}^2 \right] + \frac{\beta}{2} \right) \tag{4.11}$$

and

$$\mathbf{R_s} = \sum_{j=1}^{L} \left(2V_j \log_2 \left[\frac{\ln(2)\, C_j M T_j^2 \zeta_j^2}{2} \right] + 2V_j \beta \right), \tag{4.12}$$

where the constant β is defined by:

$$\beta = \left(\sum_{j=1}^{L} \left[\sum_{i=1}^{K_j} \frac{N_{ij}}{2} + 2V_j \right] \right)^{-1} \left(\mathbf{R_t} - \sum_{j=1}^{L} \left[\sum_{i=1}^{K_j} \frac{N_{ij}}{2} \log_2 \left[2 \ln(2)\, C_j \sigma_{ij}^2 \right] + 2V_j \log_2 \frac{\ln(2)\, C_j M T_j^2 \zeta_j^2}{2} \right] \right).$$

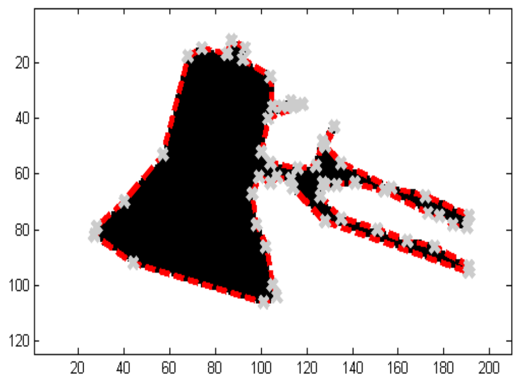
The derivation of these rate distribution equations is shown in Appendix A.2.

The above RD analysis assumes the layer contours are piecewise linear. The closed-form solutions in (4.11) and (4.12), therefore, depend on the number of vertices. Next, we generalise the rate allocation for encoding arbitrary layer contours.

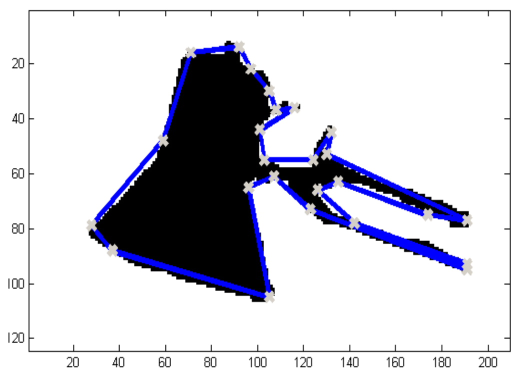
4.3.4 RD optimisation - arbitrary layer contours

The bit allocation strategy in (4.11) and (4.12) requires the knowledge of the number of vertices V_j in each contour. Given a set of arbitrary contours (non-piecewise linear) this parameter also depends on the target bit budget. For example, at low rates only a coarse approximation should be used and this corresponds to a small number of vertices, whereas at high rates an accurate reconstruction is required.

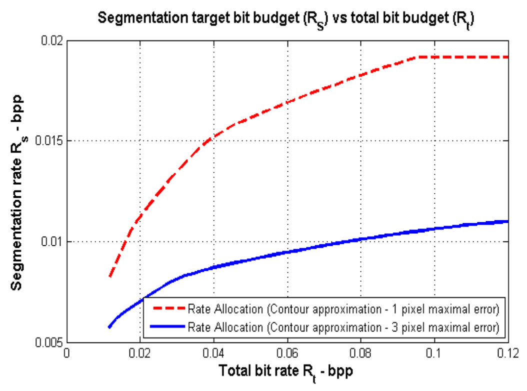
To find the correct allocation for non-piecewise linear contours we use the following approach: first we obtain a number of piecewise linear approximations of the layer contours for different error margins. We illustrate two versions of the resulting approximations in Fig. 4.4(a) and Fig. 4.4(b). These approximations are obtained using a fast algorithm proposed in [42]. The approach initialises a vertex at each point on the contour, which are then iteratively removed until a maximum allowed distortion is achieved. The distortion is measured as the perpendicular error from a vertex to the linear approximation, and is evaluated at each point on the original boundary to evaluate the maximal distortion.



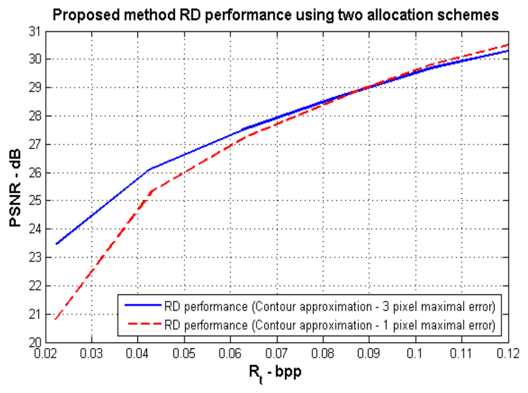
(a)



(b)



(c)



(d)

Figure 4.4: Two piecewise linear approximations of a layer in the Tsukuba dataset [74] and the corresponding rate allocation curves for each model. (a) Approximation obtained using a maximal perpendicular error of 1 pixel. (b) Approximation obtained using a maximal perpendicular error of 3 pixels. (c) Two closed-form rate allocation curves corresponding to each approximation obtained using (4.11) and (4.12). Here, the x-axis corresponds to the total bit budget $\mathbf{R_t}$ and the y-axis to the rate allocated to the segmentation $\mathbf{R_s}$. (d) The obtained PSNR curves using the rate allocation curves in (c).

For each approximation we obtain a different V_j and use it to evaluate the closed-form rate allocation curves in (4.11) and (4.12). The resulting rate allocation curves are illustrated in Fig. 4.4(c). Here, one rate allocation curve is due to the number of vertices in Fig. 4.4(a) and the other to the V_j obtained from Fig. 4.4(b).

We also show in Fig. 4.4(d) the PSNR achieved by the complete compression algorithm when the approximations in Fig. 4.4(a) and Fig. 4.4(b) are used to infer the bit allocation. The figure shows that the bit allocation obtained using the coarse approximation leads to better overall performance at low rates, and vice versa at high rates. Therefore, the problem is to choose the right approximation and thus the right

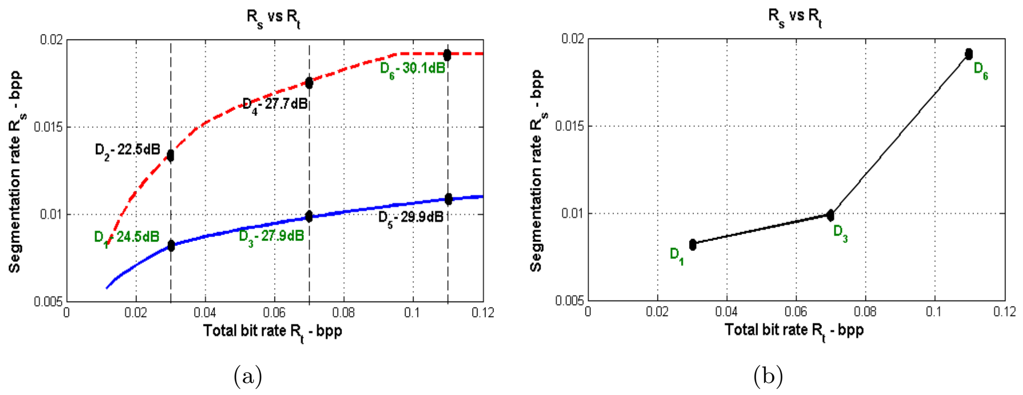


Figure 4.5: Proposed approach to obtain the rate allocation curve for arbitrary layer contours. (a) A number of total bit budgets are chosen and the points which lie on the closed-form rate allocation curves are encoded. For each bit budget, the allocation which corresponds to the least distortion is retained. (b) The intermediate points are subsequently interpolated.

number of vertices given the total bit budget. To solve this, a small number of bit rates are tested and for each of these rates various approximations are considered. The corresponding bit allocations, obtained using (4.11) and (4.12), are encoded using the complete compression algorithm. The operational points with the least distortion are retained and the intermediate bit budgets interpolated. This process is illustrated in Fig. 4.5. In this example (see Fig. 4.5(a)) three bit budgets are chosen (i.e., 0.03bpp, 0.07bpp and 0.11bpp), and, since there are two possible rate allocation curves, we end up with six operational points. For each point the complete compression algorithm is executed leading to six PSNR values $(D_1, D_2, \ldots, D_6)$ shown on the plot. We pick D_1 because $D_1 > D_2$, similarly D_3 and D_6 are chosen since $D_3 > D_4$ and $D_6 > D_5$. The complete bit allocation strategy is obtained by linear interpolation of the three rate allocations related to D_1, D_3 and D_6. This is shown in Fig. 4.5(b).

4.4 Summary

In this chapter we presented a multiview image coding method based on the sparse representation outlined in Chapter 3. The layer contours are encoded in a lossy or lossless modality given a rate constraint $\mathbf{R_s}$. In the lossless case, the contour of each layer is coded using a differential Freeman method. In the lossy scenario we approximate each layer using a piecewise linear contour in the quadtree prune-join representation. Following the multi-dimensional DWT, we quantise and entropy code the coefficients using a bit-plane coding method similar to EBCOT [85]. The rate allocation among the transform coefficients is allocated in the RD sense given a target bit budget $\mathbf{R_x}$.

In addition, given a total bit budget $\mathbf{R_t}$, we outlined a method to find the correct rate allocation among the layer contours $\mathbf{R_s}$ and the transform coefficients $\mathbf{R_x}$. In the case when the layer contours are piecewise linear, we derived a closed-form rate allocation solution. The problem was solved by upper bounding the distortion due to layer contour encoding. The upper-bound of the texture and layer contour distortion was then minimised given a rate constraint using Lagrangian multipliers. For arbitrary layers (non-piecewise linear), we outlined an empirical method to find the rate allocation based on the proposed model. This approach requires that a small number of operating points are used in the compression algorithm. The obtained rate allocation curve can then be reused to encode the dataset for an arbitrary bit budget.

Chapter 5

Multiview Compression Experimental Results

In this chapter we evaluate the performance of the proposed multiview compression method. Next we verify the proposed RD model and the rate allocation strategy between the layer contours and the texture. The performance of the complete compression method is presented in Section 5.2, where we compare our method to the state-of-the-art H.264/AVC and its multiview coding (MVC) extension. The chapter is summarised in Section 5.3.

5.1 RD modeling evaluation

In this section, the performance of the RD optimisation strategy in bit rate allocation between the layer contours and the texture is analysed. First we verify distortion model of Section 4.3.3 and then the proposed rate allocation scheme of Section 4.3.4 for arbitrary layer contours.

5.1.1 Piecewise-linear RD model evaluation

Recall that the derived rate allocation model assumes that the input layer contours are piecewise linear. To analyse the model we use a synthetic dataset shown in Fig. 5.1. The dataset consists of one layer and a background. The layer contains 79 vertices and

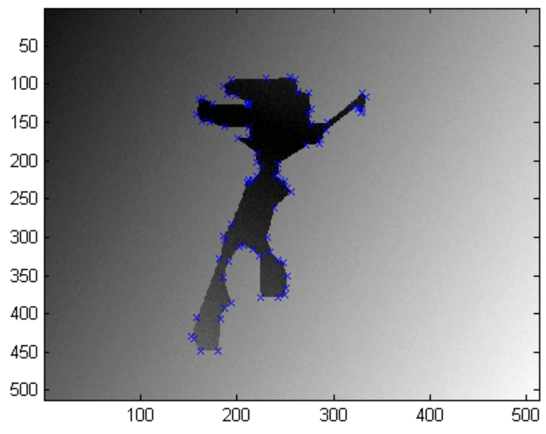


Figure 5.1: Synthetic dataset consisting of one layer and a background. The layer contour is piecewise linear and contains 79 vertices. The vertices are labeled with blue crosses.

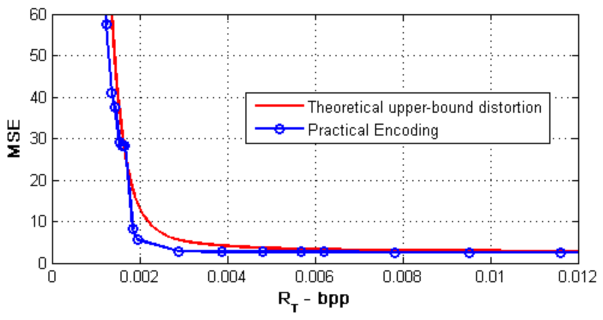


Figure 5.2: Experimental and theoretical RD performance when encoding a synthetic dataset $[512 \times 512 \times 4]$ **with one layer and a backround shown in Fig. 5.1.**

the texture is linear with an additive Gaussian signal.

For an input bit budget $\mathbf{R_t}$, we use the closed-form solutions in (4.11) and (4.12) to evaluate the texture rate $\mathbf{R_x}$ and layer contour encoding rate $\mathbf{R_s}$. The dataset is then encoded using these parameters. We evaluate the MSE of the decoded dataset and compare it to the minimised upper bound of the distortion[1]. The experimental and theoretical RD curves are shown in Fig. 5.2. These results show that the distortion model closely matches the experimental results.

5.1.2 Arbitrary layer contours model evaluation

To verify the rate allocation strategy when encoding arbitrary (non-piecewise linear) layer contours we operate as follows. First we encode the dataset using the outlined

[1]We obtain the minimised upper bound by substituting the values $\mathbf{R_x}$ and $\mathbf{R_s}$ into (4.9).

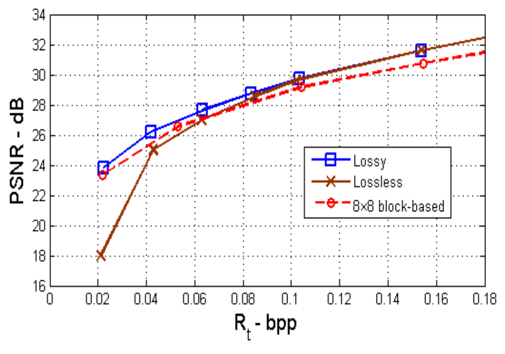
approach. Referring to Section 4.3.4 we use three piecewise approximations and five bit budgets to derive the rate allocation curves for each dataset. The overall RD performance is then compared to an exhaustive scheme where all possible combinations are considered. A RD comparison of the proposed approach and an exhaustive search is shown in Table 5.1. It is shown that our method essentially matches the performance of the exhaustive search.

Table 5.1: RD performance comparison between the proposed rate allocation method and an exhaustive search.

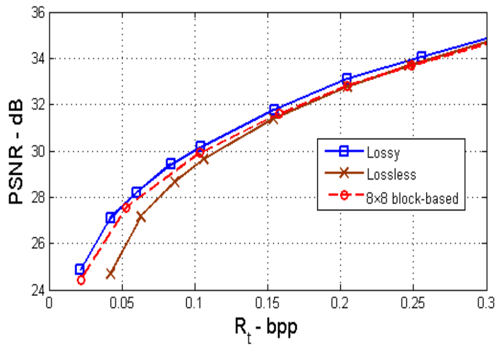
Dataset	**Tsukuba** [74]				**Teddy** [74]			
Rate/bpp	**0.05**	**0.1**	**0.2**	**0.3**	**0.056**	**0.1**	**0.2**	**0.3**
Proposed method/dB	26.65	29.63	33.17	35.32	27.93	29.98	33.01	34.83
Exhaustive search/dB	26.73	29.63	33.17	35.32	28.05	30.05	33.01	34.85

In addition we also show a comparison of proposed allocation strategy and the lossless scenario in Fig. 5.3. In the lossless case, the segmentation is encoded using the modified Freeman algorithm at all bit rates. Observe that at a PSNR of 29dB the proposed approach provides a bit rate saving of 20% when encoding Teddy. We note that due to the layer contours being more irregular and the representation having a larger number of layers, the improvement of the proposed approach is higher on Teddy than Tsukuba. We also compare the proposed approach to a block-based coding strategy. In this case the layer contours are encoded using the quadtree prune-join scheme where the smallest block size is $[8 \times 8]$ pixels and can only be represented using a 0 or 1 tile (with no intermediate tiles). The representation is then losslessly encoded over all of the bit rates. This type of approach is simpler but less effective than our solution since there is no optimisation between the texture and the layer contours.

In summary, the presented results in Fig. 5.3 and Table 5.1 support the proposed rate allocation strategy.



(a) Tsukuba EPI



(b) Teddy

Figure 5.3: A comparison of the proposed allocation strategy with the lossless and block-based coding of the layer contours. In the lossless case the layer contours are encoded using the Freeman algorithm. In the block-based approach the layer contours are encoded using the quadtree prune-join scheme, where the minimal block size is set to $[8 \times 8]$ pixels and no intermediate tiles are used. In (a) the Tsukuba layer-based representation is simple and requires fewer bits. Therefore, beyond a bit rate of 0.12bpp the proposed method encodes the layer contours in the lossless modality and the two curves converge.

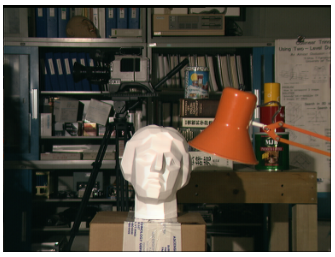
5.2 Compression Evaluation

We now evaluate the performance of our compression method. To comprehensively test the algorithm, four EPI datasets are used as shown in Fig. 5.4: Animal Farm $[232 \times 624 \times 16]$ from [8], Tsukuba $[284 \times 382 \times 4]$, Teddy $[368 \times 352 \times 4]$ and Doll $[544 \times 608 \times 4]$ (the last three from [74]). In addition, to show that the proposed method naturally scales to the additional viewpoint dimension, we also present the Tsukuba light field $[272 \times 368 \times 4 \times 4]$ results. Note that the datasets vary in geometric and texture complexity. Teddy has a wide range of disparities, whereas Animal Farm can be well approximated using a small number of depth planes. The number of extracted layers for each dataset are noted in the caption of Fig. 5.4. In addition, the data vary in terms of the number of views and the spatial resolution. Without loss of generality, only the monochromatic components of the images are encoded.

We compare our encoding method to the state-of-the-art H.264/AVC [1, 92]. To encode the data, the multiview images are treated as a set of video frames along the temporal dimension and they are compressed using the High Profile. We use the 2 pass encoding mode with the hexagonal search and quarter-pixel precision motion estimation settings. All macro-block partitions are considered and the trellis RD optimisation is

(a) Animal Farm [$232 \times 624 \times 16$]

(b) Teddy [$368 \times 352 \times 4$]


(c) Tsukuba EPI [$284 \times 382 \times 4$]

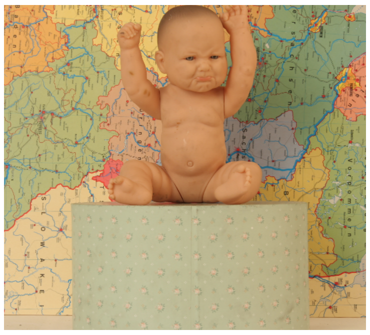

(d) Doll [$544 \times 608 \times 4$]

Figure 5.4: Multiview image analysis datasets. The data is segmented into the following number of layers: (a) 13 layers, (b) 12 layers, (c) 10 layers, (d) 5 layers and (e) 13 layers.

applied during the final encoding of a macro block. In addition the early SKIP and coefficient thresholding on P-frames settings have been disabled [1]. We specify the number of B-frames used in each dataset in Fig. 5.5. Note that although H.264/AVC is specialised for temporal data, the algorithm has a wide range of tools and is RD optimised. Therefore, this makes the method a good compression benchmark even when encoding multiview images. A quantitative comparison of the proposed method and H.264/AVC is shown in Fig. 5.5. In addition to the proposed rate allocation strategy, we include the results of a fixed allocation, where a constant 10% of the total bit rate is allocated to encode the layer contours. Note that the fixed allocation strategy has a lower complexity and is an approximation to the proposed method. In this scheme, the number of bits used to encode the segmentation is set to $\mathbf{R_s} = \mathbf{0.1R_t}$, and this is an approximation of a typical curve obtained in Fig.4.5(b).

The results show that at low bit rates the proposed codec achieves a better RD performance on all datasets. The gains are different and they are influenced by the accuracy of the layer segmentation algorithm outlined in Section 3.2. For example, in the Tsukuba EPI dataset, our proposed algorithm achieves an improvement of almost 3dB at a bit rate of $\mathbf{R_t} = 0.05$bpp, whereas, in case of Teddy, this gain is around 2dB. Moreover, observe that the performance of the alternative allocation strategy is similar to the proposed approach, with a maximal PSNR reduction of 0.13dB and 0.2dB in case of Tsukuba EPI and Teddy, respectively. To show that the proposed algorithm naturally scales to an additional viewpoint dimension, we present the Tsukuba Light Field results in Fig. 5.5(e). In this case, to obtain a fair evaluation, we compare our results to the H.264/MVC implementation specialized for multiview video. In comparison our method attains a bit rate saving of 40% at a PSNR of 30dB. The qualitative results are shown in Fig. 5.6.

Regarding high rate encoding, it is shown that the proposed algorithm is competitive with H.264/AVC and MVC. Analysing Animal Farm and Tsukuba, the proposed codec achieves better RD performance over the complete range of bit rates. In these cases, the layer based representation captures the 3D scene efficiently and the segmentation is accurate. Indeed, the data fits our designed representation and seems to truly consist of layers. Additionally, there are a large number of occluded regions, which our algorithm deals with efficiently. On the other hand, H.264/AVC attains better results when encoding Teddy at higher rates. In this case, the 3D structure of the scene is complicated with a large number of disparities which are not captured effectively by the layer-based representation. Hence, the segmentation errors create high-pass transform coefficients, which degrade the performance of our method.

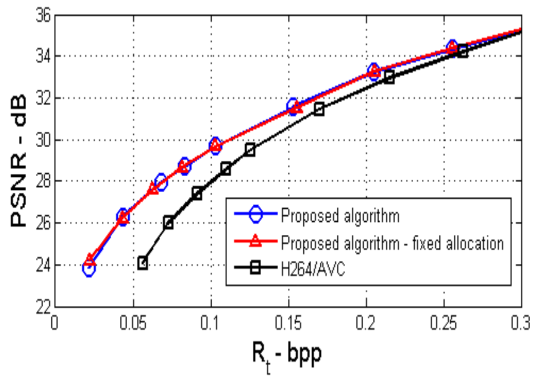
To illustrate the side information overhead, we show the total rate distribution between texture, segmentation, and side information in Fig. 5.7 (Tsukuba EPI). It is shown that at low rate, the side information cost is 9% of the total bit budget. However, as the total bit budget increases to 0.1bpp, the overhead reduces to 3%. Note that side information includes the cost to encode the following parameters: disparity gradients, a list of SKIP symbols which specifies whether a block of transform coefficients is transmitted, largest bit plane and the number of bytes allocated to each block.

Regarding the complexity of the proposed method, we note that the most computationally intensive stage is the layer-based extraction. This process, however, can be implemented offline with the obtained layer contours being used to compress the data for different bit budgets. The complexity of the encoding method itself is determined by the separable wavelet transform which is $O(N)$, where N is the total number of pixels in the dataset.

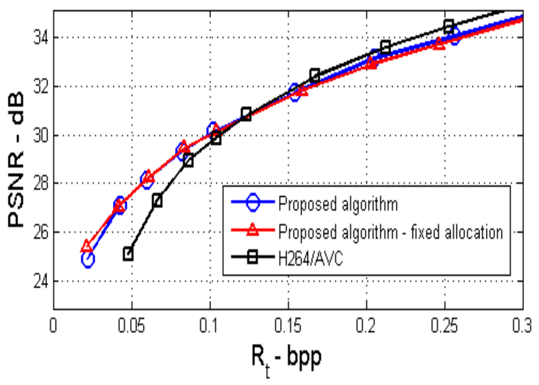
5.3 Summary

Here we evaluated our multiview image compression algorithm proposed in Chapter 4. We first verified the proposed RD model and the rate allocation scheme between the layer contours and the texture. Given a piecewise linear contour, we showed that the closed-form rate allocation expressions correspond to a distortion which is closely related to the operating distortion. In the case of non-piecewise linear contours, it was shown that our approach is competitive to an exhaustive search while having a significantly lower complexity.

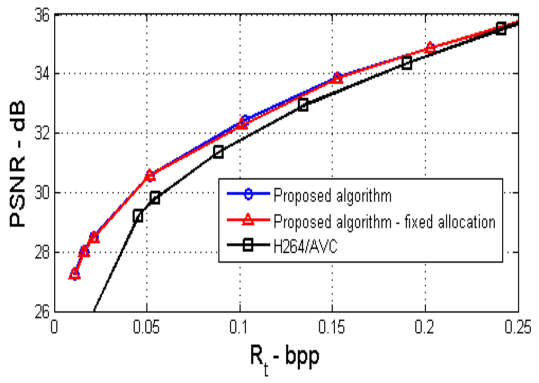
In addition, the presented results have shown that the proposed method achieves an improved RD performance at low rates and is competitive at high rates in comparison to H.264/AVC and MVC.



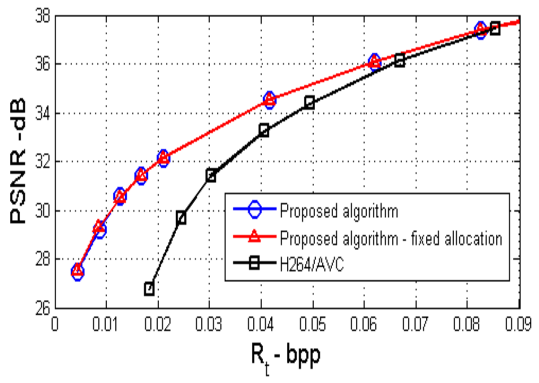
(a) Tsukuba EPI



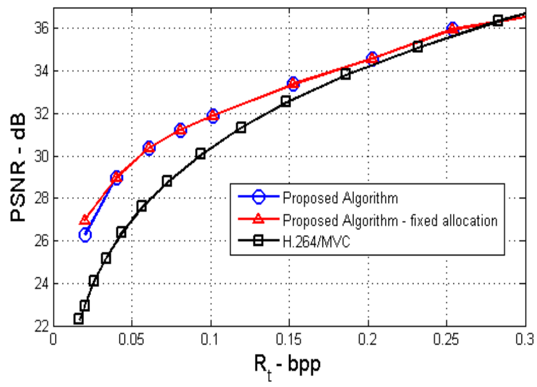
(b) Teddy



(c) Doll



(d) Animal Farm



(e) Tsukuba Light Field

Figure 5.5: Quantitative comparison of the proposed algorithm with H.264/AVC and MVC. In (a), (b) and (c) H.264/AVC encodes the dataset with one I-frame, two P-frames, and one B-frame. In (d) one I-frame, eight P-frames and seven B-frames are used. Regarding the proposed method, we apply a maximal number of decompositions in the inter-view DWT.

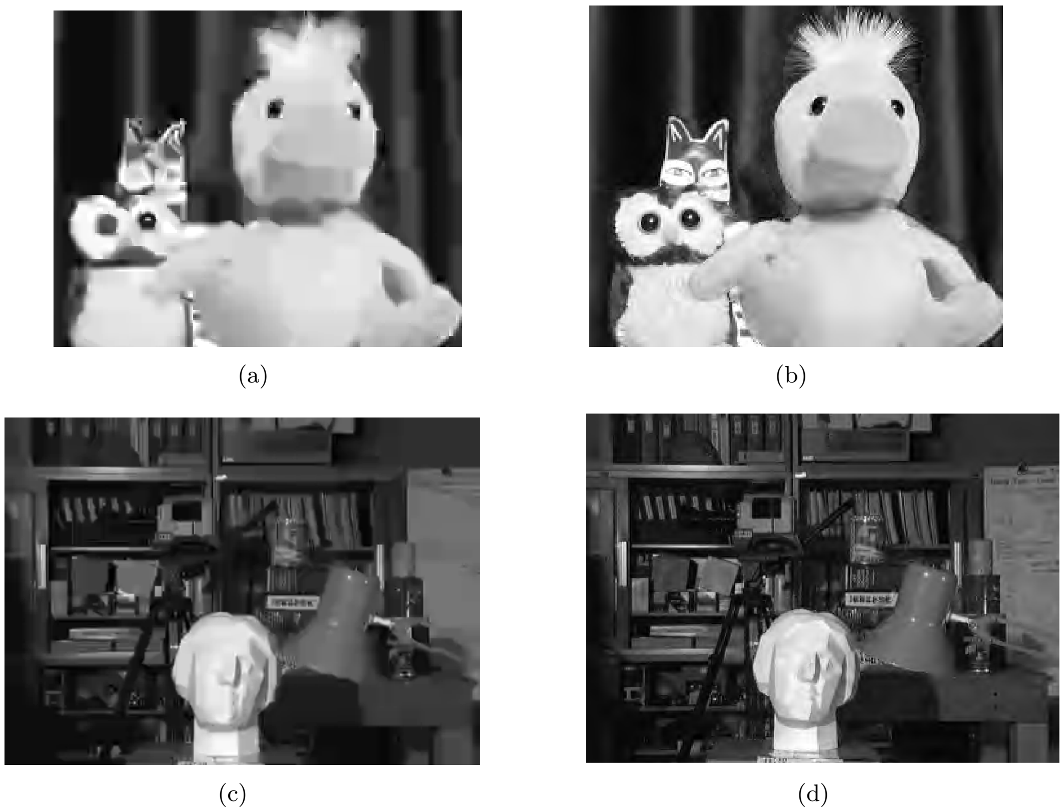
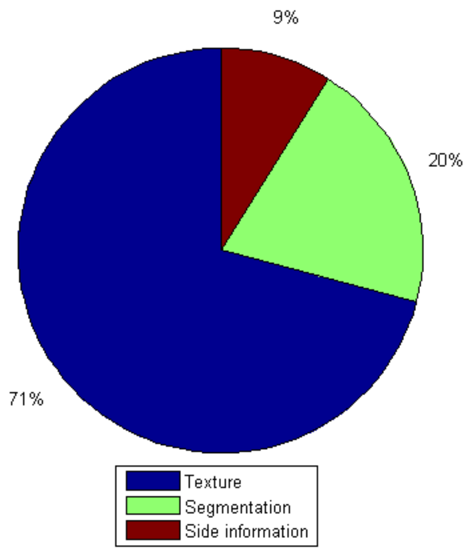
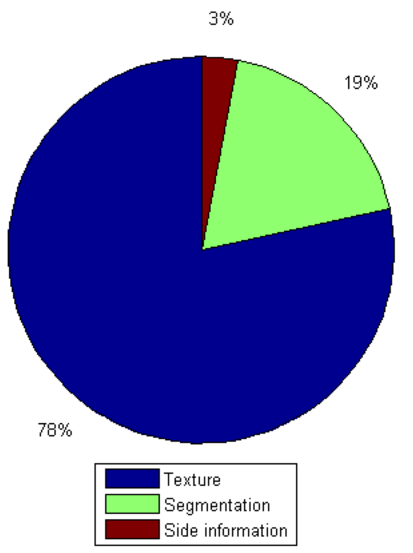


Figure 5.6: Qualitative comparison of the proposed algorithm with H.264/AVC and MVC. (a) Animal Farm encoded using H.264/AVC at 0.024bpp (PSNR 28.93dB). (b) Animal Farm encoded using the proposed algorithm at 0.021bpp (PSNR 32.14dB). (c) Tsukuba Light Field encoded using H.264/MVC at 0.056bpp (PSNR 27.53dB). (d) Tsukuba Light Field encoded using proposed algorithm at 0.052bpp (PSNR 29.77dB).



(a) Tsukuba EPI



(b) Teddy

Figure 5.7: Bit rate distribution between texture, segmentation and side information when encoding Tsukuba EPI. (a) Total bit budget 0.022bpp (PSNR 24.1dB). (b) Total bit budget 0.1bpp (PSNR 29.65dB).

Chapter 6

Interactive Compression

In this chapter we present a compression algorithm for an interactive communication system. In this setup the images are stored at a central server and are transmitted to the user on request. This allows the user to download only a fraction of the dataset by requesting only the images which are going to be viewed.

One approach to implement this kind of system is to use a coding scheme which supports random access. Recall that random access is the ability to decode a single image without decoding the complete dataset. In this way, only the relevant information can be transmitted without decoding the complete dataset. However, the concept of random access conflicts with the idea of efficient compression. In compression, we aim to remove any redundant information, and this introduces interdependencies between images, thus limiting random access.

Here, we outline how the centralised compression algorithm proposed in Chapter 4 can be modified to support random access. The main idea of the method is to use DSC in place of the multi-dimensional DWT. By using DSC we support random access while still reducing the redundancy of the transmitted information.

This chapter is organised as follows. Next we give a brief overview of DSC and outline how the method is used in our coding system. In Section 6.2 we outline the proposed compression algorithm and present the RD model in Section 6.3. The compression algorithm and the RD model are verified in Section 6.4 and we summarise in

Section 6.5.

6.1 Distributed Source Coding

A conventional coding setup usually assumes a high complexity encoder and a low complexity decoder. For example, in traditional video coding, a GOP is jointly coded by using past frames as a prediction for the current frame. This method requires a complex motion estimation process to find an accurate prediction. At the decoder, the frames are reconstructed with low complexity using the transmitted motion vectors.

In certain applications it is not possible to have this kind of setup. For example, in a remote multi-sensor network, the sensors have a limited power supply. This constraint means that the sensors are not allowed to communicate, and therefore it is not possible to design a joint coding scheme which removes the data redundancy.

The theory of DSC outlines that under certain conditions, a proper separate coding of each source can in fact achieve the same performance as a joint coding scheme (if decoding is performed jointly). This process involves knowing the correlation between the sources and shifting the complexity from the encoder to the decoder. We refer to [67] for an in depth overview of DSC. To give an intuitive example of the coding scheme consider the following setup: Assume we wish to transmit the reading from two sources $S_1 = 23$ and $S_2 = 25$. The values are uniformly distributed in the range $[0, 31]$. In addition, the maximum difference between S_1 and S_2 (worst case error) is upper-bounded by $|S_1 - S_2| \leq 3$. We can encode the two symbols using DSC as follows. The first symbol S_1 is encoded independently using $\log_2(32) = 5$ bits and is transmitted to the decoder. Then, to correctly reconstruct the second symbol we only need to transmit the three least significant bits (LSB) from S_2. Denote this with $S_2^3 = \{001\}$. By receiving this information, the decoder can infer that the possible values of S_2 are $\{\ldots, 9, 17, 25, 33, \ldots\}$[1]. Then, to decode S_2 we simply choose the closest value to S_1 from the set of possible solutions. Thus, we correctly deduce that $S_2 = 25$. This approach is known as decoding with side-information[2]. Therefore, using this approach

[1]This set is created by adding different combinations of most significant bits to the received LSB.
[2]This example is interesting because we do not need to know the exact value of S_1 to correctly

we use in total 8 bits to encode the data.

Note that if the two sources were encoded independently, we would require $2 \times \log_2(32) = 10$ bits. Also, in this particular case, a centralised scheme would use the same number of bits as the DSC approach. In the joint setup we would encode S_1 independently followed by the residual error of S_2. This approach would also use $\log_2(32) + \log_2(3) + 1 = 8$ bits (the additional 1 bit is required to specify the sign of the residual error).

In above example, DSC allows us to encode each source independently while reducing the redundancy at the decoder. The approach supports random access, and requires fewer bits than conventional independent coding of each source.

We can use the same coding principles to compress multiview images. Instead of S_1 and S_2 being scalar values, we can analyse them as images. Then, we can encode the image S_1 using an independent coding scheme and S_2 using DSC. In a typical multiview image dataset we encode more than two images, and this means that there is more uncertainty in the side information at the decoder. However, using DSC we can still take this into account by changing the correlation. In the above example, this would be equivalent to increasing the worst case error between the sources.

6.2 Proposed Algorithm

The structure of the centralised and interactive schemes is very similar. They both rely on the layer-based representation to improve the compression performance. The biggest difference in the interactive method is that we apply DSC to the spatial transform coefficients[3].

The overview of the proposed interactive method is shown in Fig. 6.1. In the first stage we obtain the layer-based representation which is followed by a pre-processing stage applied to each layer. Then, we reduce the intra-view redundancy using a shape-adaptive 2D DWT applied to each image in the obtained layers. This is followed by DSC applied to the quantised subband coefficients along the inter-view domain. We

reconstruct S_2. For example we would obtain the same answer with $S_1 = 26$.

[3]We apply the intra-view spatial DWT prior to inter-view DSC.

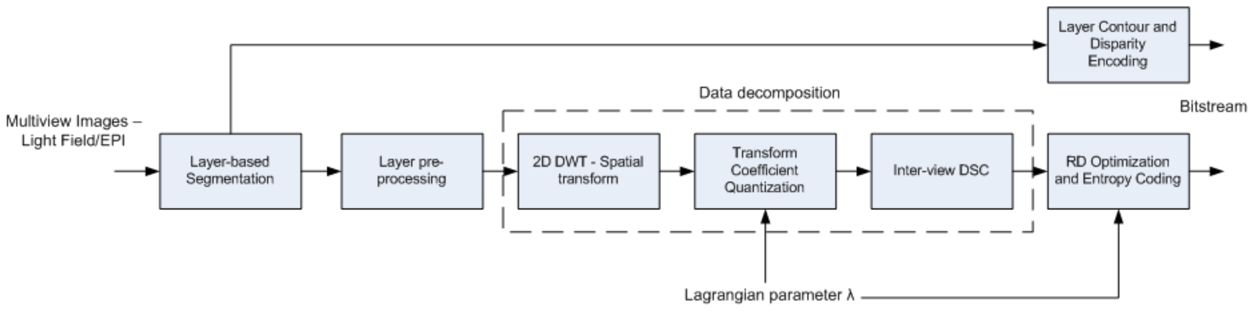


Figure 6.1: Interactive compression method block diagram.

then optimise the coefficients in the RD sense and entropy code the resulting data. In addition, we losslessly encode and transmit the layer contours and the disparity gradient of each layer. Next, we discuss in more detail each of the encoding steps.

6.2.1 Layer pre-processing and Spatial Transform

The input to this stage are the extracted layers shown in Fig. 2.7. The aim is to increase the inter-view correlation so that the compression efficiency of the DSC stage is improved. We implement the pre-processing stage by disparity compensating the images of each layer onto a common viewpoint using (2.6). Additionally, any occluded regions are interpolated using the mean along the EPI lines. The interpolation implies that during reconstruction, certain layer regions will overlap. However, as discussed in Section 2.3, we can infer the correct occlusion ordering by using the associated depth/disparity gradient of each layer. We show an example of the pre-processing in Fig. 6.2.

Following the layer pre-processing, we apply to each image a spatial shape-adaptive 2D DWT. Recall that the contour of each layer has an arbitrary shape. For this reason we use the shape-adaptive implementation, since filtering with a conventional 2D DWT would result in large magnitude wavelet coefficients and reduce the coding gain.

6.2.2 Quantisation of Transform Coefficients

In this stage we quantise the obtained transform coefficients. We overview this for one layer; the same process is then applied to each of the extracted layers.

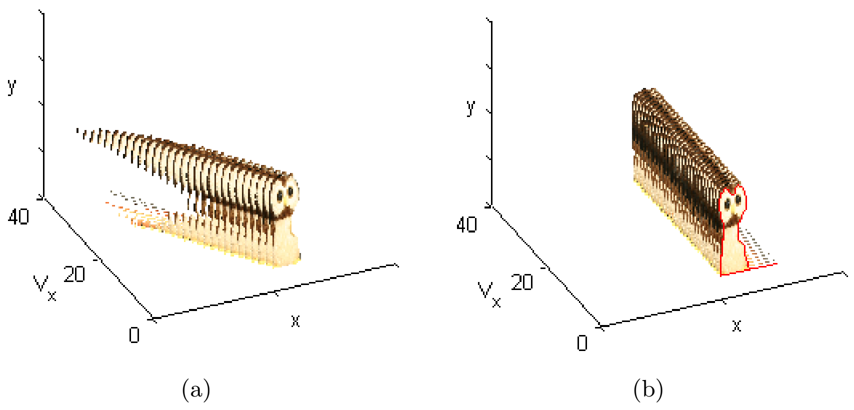


Figure 6.2: (a) Due to occlusions, the extracted layers may contain discontinuities along the EPI lines. (b) Pre-processing is implemented by disparity compensating each image onto a common viewpoint and interpolating the occluded pixels using the mean along the EPI lines. The layer contour outlined with the red curve is efficiently encoded using a modified version of the Freeman algorithm [54] and transmitted.

We select the first image from the layer and partition the transform coefficients into blocks of size $[64 \times 64]$. For each block we then choose a quantisation step-size by minimising the operational RD cost $D_i + \lambda R_i$, where (D_i, R_i) corresponds to the RD point associated with the i-th quantisation step-size.

To reduce the complexity, we obtain the operational RD curve using an embedded bit-plane coder [85]. Using this approach, the block only needs to be coded once to obtain all the possible operational RD points. The method encodes the bit-planes of the block starting from the most significant bit (MSB). This is equivalent to choosing a quantisation step-size according to the bit-planes, i.e $\left\{2^{(N-1)}, 2^{(N-2)}, \ldots, 2^0\right\}$ where N corresponds to the MSB.

In our approach we quantise the blocks in the remaining images of that layer by using the quantisation step-size from the first view. For example the low-pass subbands in each image (of the same layer) are quantised using the same step-size.

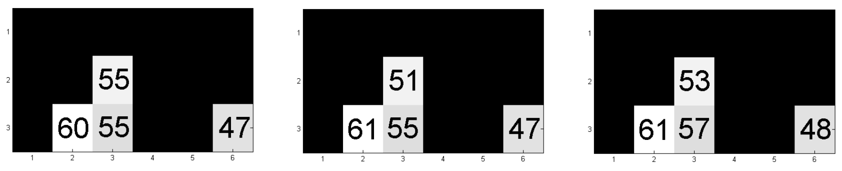
6.2.3 Inter-view Distributed Source Coding

In this stage we reduce the inter-view redundancy in the transform coefficients using DSC principles. Consider the quantised low-pass transform coefficients from three images of one layer as illustrated in Fig. 6.3.

When encoding each block, we assume that one of the blocks will be available in the cache of the user as side information. There is an uncertainty as to which one will be available; this is why we cannot simply encode the residual signal. For example, when encoding the transform coefficients in Fig. 6.3(a), the encoder does not specifically know whether the cache of the user will contain transform coefficients from Fig. 6.3(b) or 6.3(c). However, by using DSC principles, this uncertainty can be removed as follows. Consider the model:

$$y = x + n, \tag{6.1}$$

where y is the transform coefficient requested by the user, x is the side information available in the cache and $|n| \leq \epsilon$ is the residual signal. We can correctly reconstruct y using DSC by transmitting a minimum of $\lceil \log_2 (2\epsilon + 1) \rceil$ LSB from y. Therefore, when encoding each block in Fig. 6.3, we take the worst case scenario where any image can be used as side information. For example, in the sequence $\{55, 51, 53\}$, the worst case error $\epsilon \leq 4$, and this requires that we transmit $\lceil \log_2 9 \rceil = 4$ LSB from each coefficient.



(a) (b) (c)

Figure 6.3: Quantised low-pass subbands from three images of one layer. Observe that the blocks are correlated across the views.

This method is then applied to each set of transform coefficients across the views to determine the number of LSB which must be transmitted.

6.2.4 RD Optimisation and Entropy Coding

Observe that the outlined DSC approach described in the previous section is inefficient when the transform coefficients across the views are the same except for one image. For example to encode $\{55, 55, 57\}$ we have to transmit 3 LSB from each coefficient. Intuitively, at low rates a better solution would be to set 57 to 55, so that zero LSB are encoded. We solve this problem in an RD sense as follows: for each set of transform coefficients, we find the value which corresponds to the largest error and set it to the median of the set. Then, we evaluate the change in distortion ΔD and estimate change in rate ΔR. Using a greedy approach, if

$$\Delta D + \lambda \Delta R < 0 \tag{6.2}$$

we make the substitution and iterate the process until (6.2) is positive. The trade-off between rate and distortion is set using λ, which is chosen when evaluating the quantisation step-size.

The server subsequently encodes the data using a bit-plane context adaptive arithmetic coder to attain rates close to the entropy of the source. The number of retained LSB is also encoded and transmitted along with the data. Note that for a set of blocks across the views, the number of transmitted LSB is the same. Thus, we only transmit this information once prior to decoding. This information is stored by the user for future reference. In addition, the number of transmitted LSB also provides a bit-plane significance map. This is further exploited by the entropy coder to reduce the encoding rate.

6.3 Theoretical Modeling

In this section we present a theoretical model of our encoding scheme. The aim of the model is to show that the proposed algorithm can achieve improved compression performance over independent coding. First, we present a synthetic data model which is a good approximation of real multiview images, then we evaluate the RD relation of the independent and the proposed DSC algorithm for this class of signals.

6.3.1 Data Modeling

2D α-Lipschitz model

Consider an extracted layer. We model each image in the layer using a globally smooth 2D α-Lipschitz function $f_\alpha(x, y)$ [61], which satisfies the following condition:

$$|\, f_\alpha(x_1, y_1) - f_\alpha(x_2, y_2) \,| \leq K \left(|\, x_1 - x_2 \,|^2 + |\, y_1 - y_2 \,|^2\right)^{\alpha/2}, \tag{6.3}$$

where $K > 0$. This type of signal has a property that when it is transformed using a 2D wavelet having at least $\lfloor \alpha + 1 \rfloor$ vanishing moments, the wavelet coefficients in each scale will have the following decay [61]:

$$|d_j| \leq A 2^{-j(\alpha+1)}, \tag{6.4}$$

where j is the wavelet scale and constant $A > 0$. A linear compression scheme [89] based on (6.4) can be designed by choosing a constant quantisation step-size across all the subbands [61]. It can be shown that at high rate, such a method yields the following RD function:

$$D(R) \leq cR^{-\alpha}, \tag{6.5}$$

where R is the total number of bits allocated to encoding the signal and $c > 0$ is constant. The analysis [61] also shows that the constant $c \propto A^2$.

Contour Model

In practice, the layers are outlined by a segmentation and are therefore not globally α-Lipschitz smooth. An example of an α-Lipschitz smooth signal bounded by the layer contours is shown in Fig. 6.4. To obtain the decay in (6.5), we transmit the contours and encode the texture using a shape-adaptive scheme. This supports the proposed layer-based coding method. We model the contour of the texture as a piecewise linear curve having V vertices. The RD function due to quantising the location of the vertices

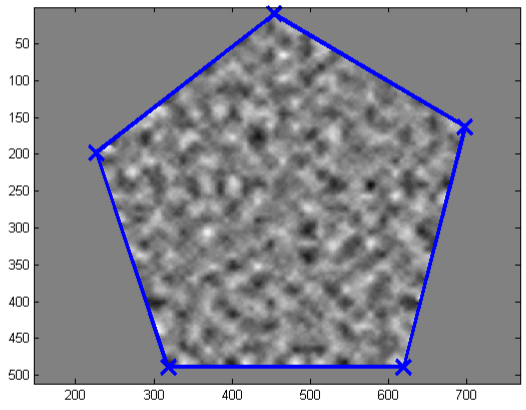


Figure 6.4: Synthetic dataset. The texture is globally α-Lipschitz smooth in the bounds of the contour.

can be upper bounded as:

$$D(R) \leq M\zeta^2 T^2 V 2^{-R/2V}, \tag{6.6}$$

where ζ is the maximal magnitude of the texture and T is the maximal length of a side of the bounding box. M is the number of images in the layer. Note that the model is evaluated by upper bounding the number of pixels affected by quantising the vertex locations and then scaling this number by the amplitude of the texture to obtain the distortion. The derivation of this bound is shown in Appendix A.1.

Multiview Image Model

Using the analysis in Section 2.3, the layer images are modeled as a shifted version of the first image and an additional 2D α-Lipschitz error term. The error corresponds to either lighting changes, layer extraction errors or non-Lambertian surfaces. Therefore,

$$f_i(x, y) = f_1(x + (i-1)\Delta p, y) + \epsilon_\alpha^i(x, y), \tag{6.7}$$

where Δp corresponds to the layer's disparity gradient and i is the image location.

6.3.2 Independent Encoding

In the independent encoding setup, the 2D α-Lipschitz texture in each view is coded independently of the remaining views. The layer contour is coded once and transmitted prior to decoding the texture. Therefore, assuming all of the images are decoded, the total distortion due to encoding the texture and the contour is bounded by:

$$\mathbf{D_{ind}}\left(\mathbf{R_t}\right) \leq \sum_{i=1}^{M} c_i \left(R_x^i\right)^{-\alpha} + M\zeta^2 T^2 V 2^{-R_s/2V}, \tag{6.8}$$

where R_x^i and R_s represent the rate allocated to the α-Lipschitz texture in the i-th view and the contour encoding rate, respectively. Therefore, the total bit rate can be shown to be:

$$\mathbf{R_t} = \sum_{i=1}^{M} R_x^i + R_s. \tag{6.9}$$

The correct rate allocation which minimises the distortion for a total bit budget can be solved using Lagrangian multipliers. A high rate analysis yields:

$$R_x^i \approx \mathbf{R_t} \left(\sum_{l=1}^{M} \left(\frac{c_l}{c_i} \right)^{\frac{1}{\alpha+1}} \right)^{-1}, \tag{6.10}$$

and

$$R_s = 2V \log_2 \left(\frac{M\zeta^2 T^2 \ln(2)}{2V\alpha c_1} \right) + 2V(\alpha+1)\log_2 R_x^1. \tag{6.11}$$

The derivation of these results are shown in Appendix A.3. The minimised RD function in terms of the total rate can be obtained by substituting (6.10) and (6.11) into (6.8).

6.3.3 Proposed Algorithm

To model the proposed interactive method, we assume that the first image is transmitted in intra modality and the remaining ones are coded using DSC. During decoding, the first image is then used as side information to reconstruct the remaining images in the dataset.

Recall that using the layer model in (6.7), we assume that images with locations $V_x = \{2, 3, \ldots, M\}$ (we call these the residual frames) are a shifted version of the first image with an additional 2D α-Lipschitz error term. This implies that we can model the wavelet coefficients in the residual views as:

$$d_i^j = \widehat{d_1^j} + d_\epsilon^j, \tag{6.12}$$

where $\widehat{d_1^j}$ denote the wavelet coefficients of the first view disparity compensated onto the i-th viewpoint, and j corresponds to the wavelet scale. This is true for each subband if the disparity is a multiple of 2^N [89], where N is the number of spatial decompositions applied by the 2D DWT. Then, based on (6.12) we can encode d_i^j by transmitting a certain number of LSB corresponding to the noise component d_ϵ^j (assuming that the side information $\widehat{d_1^j}$ is available at the decoder).

We can analyse this problem as though we are only transmitting the error component from the DSC frames [16,17] . Recall that the α-Lipschitz error is upper bounded by

$$|d_\epsilon^j| \leq A_\epsilon 2^{-j(\alpha+1)}, \tag{6.13}$$

and the constant A_ϵ is significantly smaller than in the independent coding setup. Since A_ϵ only determines the constant c in (6.5), it is reasonable to deduce that encoding the α-Lipschitz error term using DSC has the same RD behaviour as the original independent setup in (6.5), but with a different scaling constant [16].

In practise the side information in (6.12) will be quantised and this will also contribute to the reconstruction error. We use the analysis in [16] to model the total distortion in a DSC frame as:

$$\begin{aligned} D\left(R_1, R_2\right) &= D_1\left(R_1\right) + D_2\left(R_2\right), && (6.14) \\ &= c_1 R_1^{-\alpha} + c_2 R_2^{-\alpha}, && (6.15) \end{aligned}$$

where $D_1\left(R_1\right)$ corresponds to the side information distortion and $D_2\left(R_1\right)$ is the quantisation distortion in the α-Lipschitz error term. The parameters R_1 and R_2 denote the number of bits allocated to encoding the side information and DSC frame, respectively.

Hence, the distortion due to encoding the complete dataset using the proposed scheme is

$$\mathbf{D_{DSC}}\left(\mathbf{R_t}\right) \leq M c_1 R_x^{1-\alpha} + \sum_{i=2}^{M} \hat{c}_i \left(R_x^i\right)^{-\alpha} + M\zeta^2 T^2 V 2^{-R_s/2V}, \tag{6.16}$$

where we have used $\hat{c}_i$ to distinguish the constants from the independent coding setup. Note that the total RD due to encoding the dataset using the proposed method is identical to (6.8) but with different scaling constants, and this leads to the same form of rate allocation equations. We validate this behaviour in the following section.

6.4 Simulation Results and algorithm analysis

We begin by analysing the RD model and then evaluate the compression efficiency of the proposed interactive compression method.

6.4.1 RD model evaluation

We validate the RD model by encoding an $\alpha = 1.5$ Lipschitz multiview image array consisting of four images. We use a synthetic α-Lipschitz signal. First, we create the reference frame by adding zero mean Gaussian noise to each subband and then scaling the noise components to ensure the decay in (6.4) is satisfied. The subsequent frames are obtained by shifting the first image by the disparity and adding an $\alpha = 1.5$ Lipschitz error term.

To estimate the scaling constants c_i and $\hat{c}_i$ we encode each image a number of times to obtain a set of operating RD points. Then, the scaling constants are evaluated by minimising the error between the theoretical and operating RD points. We consider two scenarios: the setup where all of the images are encoded using the intra modality (to evaluate c_i) and the proposed method where the first image is coded in intra mode and the remaining ones using DSC (to evaluate $\hat{c}_i$).

The evaluated scaling constants are then used to find the rate allocation in (6.10) and (6.11), which minimise the upper bound of the distortion for a given total bit budget $\mathbf{R_t}$. We encode the data set using the optimised parameters. The resulting operating and theoretical RD curves are shown in Fig. 6.5. Observe that the performance of the proposed algorithm is better in both the theoretical and practical results. In addition, the RD curves validate our analysis that the independent and proposed methods have the same rate of decay, but with different scaling constants.

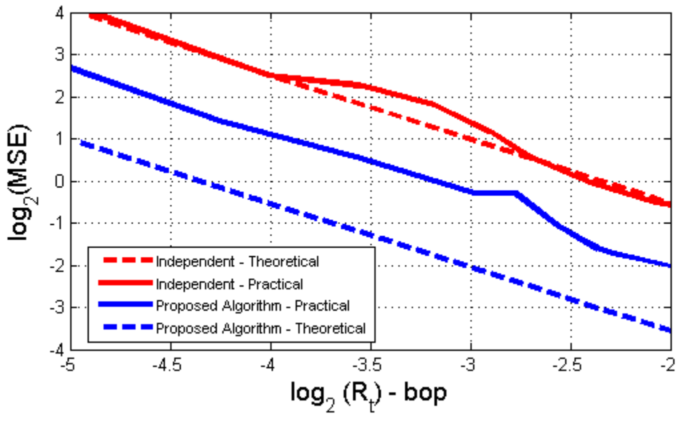


Figure 6.5: Practical and theoretical RD performance when encoding an $\alpha = 1.5$ Lipschitz signal. The proposed algorithm achieves an improved performance in both practical and theoretical cases. Observe that the rate of decay in the theoretical and practical cases is the same. Here, the x-axis corresponds to the total bit budget in terms of bits per object pixels (bop).

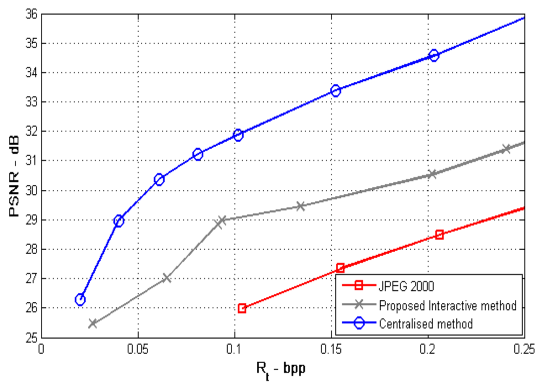
6.4.2 Interactive method

We validate the performance of the proposed compression method on real data using Tsukuba light field $[272 \times 368 \times 4 \times 4]$ and Animal Farm EPI $[232 \times 624 \times 16]$ datasets.

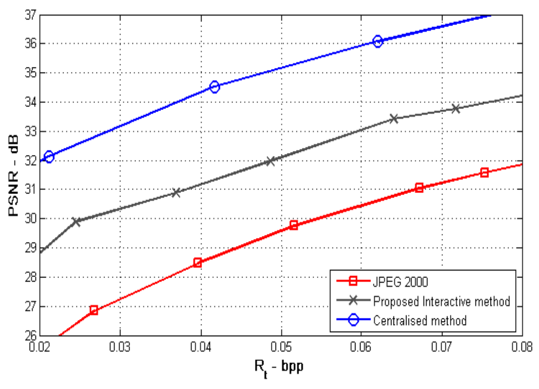
We initialise the proposed interactive method by independently encoding the first image of each layer. Then, the remaining images are randomly chosen and transmitted using the DSC strategy until all the images are decoded. The side information in the cache is always set to the previously decoded image. We compare the method to independent coding of the images using JPEG 2000[4], which has the same random access capabilities as the proposed approach.

We illustrate a quantitative comparison of the proposed method with JPEG 2000 in Fig. 6.6. Observe that the proposed method achieves PSNR gains of up to 3dB. This is due to the fact that our approach reduces the inter-view redundancy whereas JPEG

[4]We modify JPEG 2000 to have the same entropy coding as our approach.



(a) Tsukuba light field.



(b) Animal Farm EPI volume.

Figure 6.6: Quantitative comparison between the proposed interactive method, JPEG 2000 and the centralised scheme presented in Chapter 4. (a) Tsukuba light field. (b) Animal Farm EPI volume.

2000 encodes each image independently. The improvement can also be seen in terms of subjective performance in Fig. 6.7. In addition, to show the performance reduction due to the random access feature, we also include a comparison with respect to the centralised method from Chapter 4. The DSC approach incurs a cost of 3dB at low rates, and this is due to the fact that in the interactive method any image can be used as side information. This implies that we encode the dataset with respect to the worst case error, and this requires additional parity bits to be transmitted.

6.5 Summary

We presented a compression method for an interactive communication scenario. In this setup, the images are stored at a central server and are transmitted to the user when a request is made.

The requested views are typically correlated and our scheme takes this into account by encoding the images using DSC principles. DSC allows us to reduce the redundancy of the images at the client by using the previously decoded image as the side information. Furthermore, the scheme facilitates random access by encoding each image independently.

In our approach, we use the layer-based representation to exploit the correlation between the requested images, and this reduces the number of parity bits which must

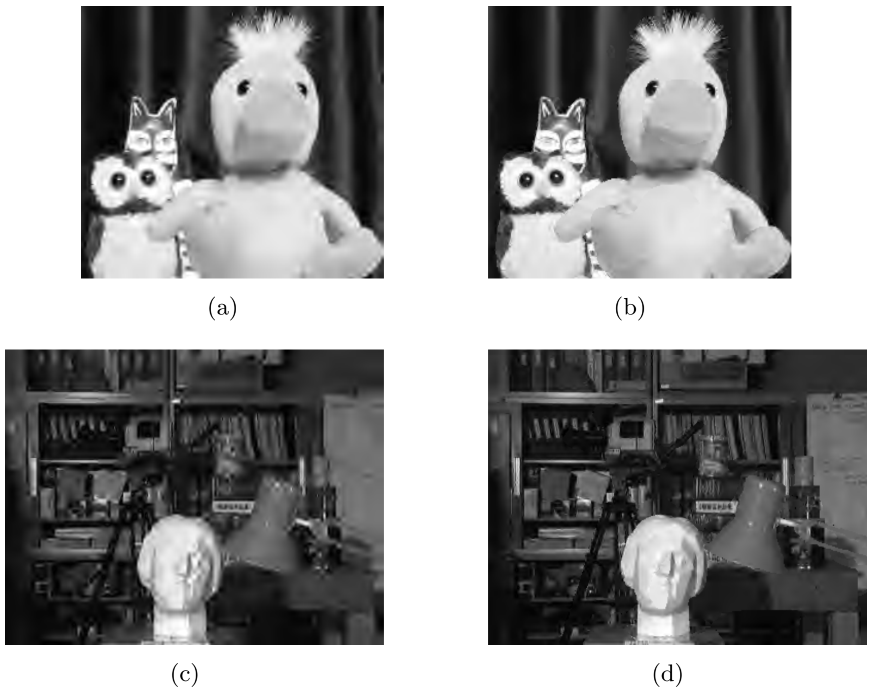


Figure 6.7: Qualitative comparison of the proposed interactive algorithm with JPEG 2000. (a) Animal Farm encoded using JPEG 2000 at 0.07bpp (PSNR 31.33dB). (b) Animal Farm encoded using the proposed interactive method at 0.07bpp (PSNR 33.76dB). (c) Tsukuba light field encoded using JPEG 2000 at 0.13bpp (PSNR 26.83dB). (d) Tsukuba light field encoded using the proposed interactive method at 0.134bpp (PSNR 29.44dB).

be transmitted when the images are coded using DSC. The DSC is implemented in the DWT transform domain. We assume the worst case correlation model, and this approach means that any image in the dataset can be correctly decoded using any other image as side information. We show that our method achieves an improved RD performance with gains of up to 3dB over independent encoding.

Furthermore, we presented a RD analysis of our algorithm which demonstrates that the proposed method can achieve a better performance in comparison to independent coding. We showed that when encoding an α-Lipschitz smooth signal, the independent coding of the images and the proposed method have the same MSE decay, but with different scaling constants.

Chapter 7

Conclusion

In this chapter we summarise the thesis, discuss which camera setup constraints can be relaxed in the formulation and outline possible future research directions.

7.1 Summary

7.1.1 Sparse representation of multiview images

We presented an approach to obtain a sparse decomposition of multiview images. The underlying idea is that a multiview image array can be analysed as a set of layers, where each layer corresponds to an object located at a constant depth in the scene. We call this the layer-based representation. After extracting the layers, we obtain the sparse decomposition by using a multi-dimensional DWT. The DWT is applied in a separable approach. First, we use an inter-view DWT applied to the camera viewpoint dimensions. The DWT is implemented using the lifting implementation, and it is applied in the direction of the EPI lines to maximise sparsity. In addition, we modify the transform to take into account occluded regions. This is followed by a 2D DWT applied to the spatial coordinates. Our evaluation shows that the proposed decomposition is sparser than a conventional DWT with the same decomposition structure.

7.1.2 Centralised compression of multiview images

We then presented a multiview image coding method based on the sparse decomposition. In this algorithm, we first encode the layer contours of the layer-based representation in a lossy or lossless modality. In the lossy case, we approximate the contours using a piecewise linear curve obtained with a quadtree prune-join representation. In the lossless setup, we use a variation of the Freeman method by encoding the outline of the layer contours. Then, following the multi-dimensional DWT we encode the transform coefficients using a method similar to EBCOT [85]. The approach uses a context-adaptive arithmetic coder and achieves rates close to the entropy of the source.

In addition given a total target bit budget, we find the correct rate allocation between encoding the layer contours and the texture. We show that if the layer contours are piecewise linear, there exists a closed-form solution. In the more general case of arbitrary layer contours, we combine the theoretical analysis with an empirical approach to find the correct allocation. Evaluation of the method showed that the algorithm outperforms H.264/AVC and MVC on a number of datasets.

7.1.3 Interactive compression of multiview images

Although the proposed centralised method achieves a high compression, it does not support random access to arbitrary images. Therefore, the approach cannot be used for applications such as interactive communication.

To support random access, we modified the proposed method by substituting the multi-dimensional DWT with DSC. Using this approach we encoded each image independently and reduced the redundancy of the transmitted data at the client. To ensure there is no drift while decoding, we used the worst case correlation error to encode the images using DSC. Simulation results showed that the method outperforms an independent coding of the images. In addition, we developed a RD model of the algorithm. The model is based on the assumption that the input images are globally α-Lipschitz smooth within the layer contours. An analysis of the model showed that independent coding of the images has the same rate of decay as the proposed method, but with different scaling constants. Simulation results confirmed this analysis.

7.2 Camera setup constraints

To obtain the sparse representation and develop the compression methods, we made a number of camera setup assumptions. Here, we outline which assumptions are essential and which ones can be relaxed. Firstly, we constrained the cameras to be uniformly spaced. When encoding an EPI volume, this can in fact be fully relaxed; the cameras can be placed at arbitrary locations on a line. This is due to the fact that the disparity between neighbouring images can be evaluated by using the camera spacing and the disparity gradient. However, in the case of a light field, our method does require that the cameras are uniformly spaced, and this is because the inter-view DWT is implemented in a separable approach; first across the row images, followed by the column images.

Secondly, we constrained the cameras to be placed on a 2D plane (light field), where the focal plane of each camera is parallel to the array. This is in fact a necessary condition for the method to operate. If the cameras are located at arbitrary locations (i.e. non-planar), the 2D contour of a layer on each image viewpoint would no longer be a shifted version, and this means that the current inter-view DWT could not be applied to the data.

However, a constraint that the focal length of each camera is the same can be removed. In this case, the disparity gradient would be camera dependent. The pixel disparity could then be evaluated by using the change in the camera location, the focal length of each camera and the object depth.

7.3 Future Work

There are a number of possible future research directions which can be explored.

7.3.1 Constant depth plane assumption

Currently each layer in the layer-based representation is modelled by a constant depth plane. A possible extension could be to use a slanted plane. The motivation is that a more accurate model would result in a sparser representation and hence improve the compression performance.

7.3.2 Image-based rendering and compression

In IBR, the compressed images are used to interpolate novel viewpoints. This implies that certain images will be used more than others. For example the images corresponding to the objects of interest will be used for interpolation more often than the background or the corner viewpoints. Therefore, a coding strategy which encodes these images at a higher rate would result in a significantly improved performance. An interesting approach would be to incorporate this into the RD model.

7.3.3 Perceptual based coding schemes

The majority of coding schemes rely on MSE to evaluate the performance and to implement rate allocation. Although the metric leads to an objective evaluation, it does not take into account the redundancy of the Human Visual System (HVS). For example, detailed texture regions lead to a high MSE, but in terms of HVS they appear stationary. An innovative scheme based on this idea in [68] proposed to remove blocks of highly textured regions at the encoder. At the decoder, these regions are then synthesised using image editing tools such as [66]. Although the approach leads to a high MSE, the perceptual quality of the image is high due to the properties of the HVS. Furthermore, since the highly textured regions are not transmitted, the method leads to significant savings in terms of the bit rate. This type of approach can be also be adapted to encode multiview images.

7.3.4 Extensions to multiview video

To encode multiview videos the proposed centralised scheme could be used to compress multiview images at each specific moment in time. Then, further redundancy could be removed from the low-pass components corresponding to the temporal dimension. A disadvantage of this approach is that it would be computationally expensive to extract the layer-based representation for multiple time frames. The complexity could perhaps be reduced by initialising the contours using the previously extracted layers or by using a lower complexity layer extraction method, such as [65].

7.3.5 Extensions to interactive compression method

In the interactive compression method, we use the worst case error across the views to encode the images. This means that when the size of the dataset becomes large, the worst case error would also increase. Therefore, the performance of the method decreases with the number of images.

A possible approach to solve this issue is to consider that the user will only request images within a window of neighbouring views. This means that even for a large dataset, the worst case error within neighbouring views would still be relatively low. Finally, performance could be further improved by using more advanced DSC schemes, such as LDPC [72] or Turbo codes [28].

Appendix A

A.1 Proof of (4.6)

Consider a polygon, bounded by a box of size T having V vertices. The distortion bound is derived by quantising the vertex locations for a given bit budget of R bits, where each vertex is, therefore, allocated $\frac{R}{V}$ bits. This corresponds to quantising the x and y locations with a step-size $\Delta = T2^{-\frac{R}{2V}}$. The maximal quantisation error ϵ along each dimension can be upper bounded by:

$$\begin{aligned} \epsilon &\leq \frac{\Delta}{2} \\ &= \frac{T}{2}2^{-\frac{R}{2V}}. \end{aligned} \tag{A.1}$$

Therefore, using Pythagoras, the maximal distance to the original vertex can be upper bounded by $\frac{T}{\sqrt{2}}2^{-\frac{R}{2V}}$. Since each side of the polygon is bounded by $T\sqrt{2}$, the number of pixels affected by quantising the vertices is bounded by $T^2V2^{-\frac{R}{2V}}$.

Denote with ζ the maximal amplitude of the texture in the polygon. Assuming the texture is set to zero in the quantisation error regions, the total distortion can be upper bounded by:

$$e^2 \leq \zeta^2 T^2 V 2^{-\frac{R}{2V}}, \tag{A.2}$$

which coincides with (4.6).

A.2 Proof of (4.11) and (4.12)

Here, we prove the optimum rate distribution between segmentation and texture in (4.11) and (4.12). The cost function to be minimised is defined by:

$$\begin{aligned} D(\mathbf{R_x}, \mathbf{R_s}) &\sim D_x(\mathbf{R_x}) + D_s(\mathbf{R_s}) \\ &= \sum_{j=1}^{L} C_j \left(\sum_{i=1}^{K_j} N_{ij}\sigma_{ij}^2 2^{-2R_x^{ij}} + MT_j^2 V_j \zeta_j^2 2^{-\frac{R_s^j}{2V_j}} \right), \end{aligned} \tag{A.3}$$

where $\mathbf{R_x}$ and $\mathbf{R_s}$ are defined as in (4.5) and (4.8), respectively. The minimisation is subject to a bit rate constraint:

$$\begin{aligned} \mathbf{R_t} &\geq \mathbf{R_x} + \mathbf{R_s} && \text{(A.4)} \\ &= \sum_{j=1}^{L}\sum_{i=1}^{K_j} N_{ij}R_x^{ij} + \sum_{j=1}^{L} R_s^j. && \text{(A.5)} \end{aligned}$$

The constrained optimisation can be transformed into an unconstrained one using Lagrangian multipliers, where the new objective function to be minimised is given by:

$$J(\mathbf{R_x}, \mathbf{R_s}, \lambda) = \sum_{j=1}^{L} C_j \left(\sum_{i=1}^{K_j} N_{ij}\sigma_{ij}^2 2^{-2R_x^{ij}} + MT_j^2 V_j \zeta_j^2 2^{-\frac{R_s^j}{2V_j}} \right) + \lambda \left(\sum_{j=1}^{L}\sum_{i=1}^{K_j} N_{ij}R_x^{ij} + \sum_{j=1}^{L} R_s^j - \mathbf{R_t} \right). \tag{A.6}$$

The minimum is obtained by partially differentiating the cost function with respect to the free variables and setting the derivative to zero:

$$\begin{aligned} \frac{\partial J}{\partial R_x^{ij}} &= -2\ln(2)\, C_j N_{ij}\sigma_{ij}^2 2^{-2R_x^{ij}} + \lambda N_{ij} = 0, && \text{(A.7)} \\ \frac{\partial J}{\partial R_s^j} &= -\frac{\ln(2)}{2} C_j MT_j^2 \zeta_j^2 2^{\frac{-R_s^j}{2V_j}} + \lambda = 0. && \text{(A.8)} \end{aligned}$$

Therefore,

$$\begin{aligned} R_x^{ij} &= \frac{1}{2}\log_2\left[2\ln(2)\, C_j\sigma_{ij}^2\right] - \frac{1}{2}\log_2\lambda, && \text{(A.9)} \\ R_s^j &= 2V_j\log_2\left[\frac{\ln(2)\, C_j MT_j^2\zeta_j^2}{2}\right] - 2V_j\log_2\lambda. && \text{(A.10)} \end{aligned}$$

We can now substitute (A.9) and (A.10) into the bit rate constraint in (A.5) and solve for $\log_2 \lambda$ to obtain:

$$\log_2 \lambda = -\left(\sum_{j=1}^{L}\left[\sum_{i=1}^{K_j}\frac{N_{ij}}{2}+2V_j\right]\right)^{-1}\left(\mathbf{R_t}-\sum_{j=1}^{L}\left[\sum_{i=1}^{K_j}\frac{N_{ij}}{2}\log_2\left[2\ln(2)\,C_j\sigma_{ij}^2\right]+2V_j\log_2\frac{\ln(2)\,C_jMT_j^2\zeta_j^2}{2}\right]\right). \tag{A.11}$$

We denote with $\beta = -\log_2 \lambda$. Substituting (A.9) and (A.10) into (4.5) and (4.8), we obtain the following solutions:

$$\mathbf{R_x} = \sum_{j=1}^{L}\sum_{i=1}^{K_j} N_{ij}\left(\frac{1}{2}\log_2\left[2\ln(2)\,C_j\sigma_{ij}^2\right]+\frac{\beta}{2}\right), \tag{A.12}$$

and

$$\mathbf{R_s} = \sum_{j=1}^{L}\left(2V_j\log_2\left[\frac{\ln(2)\,C_jMT_j^2\zeta_j^2}{2}\right]+2V_j\beta\right), \tag{A.13}$$

which coincides with the layer contour and texture rates in (4.11) and (4.12), respectively.

A.3 Proof of (6.10) and (6.11)

Here, we minimise the upper bound of the cost function

$$\mathbf{D_{ind}}(\mathbf{R_t}) \le \sum_{l=1}^{M} c_l\left(R_x^l\right)^{-\alpha} + M\zeta^2T^2V2^{-R_s/2V}, \tag{A.14}$$

given the constraint

$$\mathbf{R_t} = \sum_{l=1}^{M} R_x^l + R_s. \tag{A.15}$$

We transform the constrained problem into an unconstrained one using Lagrangian multipliers as follows:

$$J(\mathbf{R_t},\lambda) \le \sum_{l=1}^{M} c_l\left(R_x^l\right)^{-\alpha} + M\zeta^2T^2V2^{-R_s/2V} + \lambda\left(\sum_{l=1}^{M} R_x^l + R_s - \mathbf{R_t}\right). \tag{A.16}$$

The minimum is obtained by differentiating the Lagrangian cost with respect to the free parameters, and setting the partial derivatives to zero:

$$\frac{\partial J}{\partial R_x^1} = -\alpha c_1 \left(R_x^1\right)^{-(\alpha+1)} + \lambda = 0, \tag{A.17}$$

$$\frac{\partial J}{\partial R_x^2} = -\alpha c_2 \left(R_x^2\right)^{-(\alpha+1)} + \lambda = 0, \tag{A.18}$$

$$\vdots$$

$$\frac{\partial J}{\partial R_x^M} = -\alpha c_M \left(R_x^M\right)^{-(\alpha+1)} + \lambda = 0, \tag{A.19}$$

$$\frac{\partial J}{\partial R_s} = -\frac{\ln 2}{2V} M\zeta^2 T^2 2^{-\frac{R_s}{2V}} + \lambda = 0. \tag{A.20}$$

Substituting out λ, we can derive the following equations:

$$R_x^l = R_x^1 \left(\frac{c_l}{c_1}\right)^{\frac{1}{\alpha+1}}, \text{ for } l = 1, 2, \ldots, M. \tag{A.21}$$

$$R_s = 2V \log_2 \left(\frac{\ln 2 M\zeta^2 T^2}{2V\alpha c_1}\right) + 2V(\alpha+1)\log_2\left(R_x^1\right). \tag{A.22}$$

Substituting these equations into (A.15), $\mathbf{R_t}$ can be expressed in terms of R_x^1 as follows:

$$\mathbf{R_t} = R_x^1 \left(\sum_{l=1}^{M} \left(\frac{c_l}{c_1}\right)^{\frac{1}{\alpha+1}}\right) + 2V \log_2 \left(\frac{\ln 2 M\zeta^2 T^2}{2V\alpha c_1}\right) + 2V(\alpha+1)\log_2\left(R_x^1\right). \tag{A.23}$$

For a high rate $\mathbf{R_t}$, we obtain the approximation:

$$R_x^1 \approx \mathbf{R_t} \left(\sum_{l=1}^{M} \left(\frac{c_l}{c_1}\right)^{\frac{1}{\alpha+1}}\right)^{-1}. \tag{A.24}$$

This analysis can be generalised to obtain

$$R_x^i \approx \mathbf{R_t} \left(\sum_{l=1}^{M} \left(\frac{c_l}{c_i}\right)^{\frac{1}{\alpha+1}}\right)^{-1}, \text{ for } i = 1, 2, \ldots, M. \tag{A.25}$$

The results in (A.25) and (A.22) coincide with (6.10) and (6.11), respectively.

Bibliography

[1] "H.264/AVC Video Coding Algorithm." [Online]. Available: http://x264.nl/

[2] A. Aaron, P. Ramanathan, and B. Girod, "Wyner-Ziv coding of light fields for random access," in *Proceeedings of the IEEE Workshop on Multimedia Signal Processing (MMSP)*, Sept. 2004, pp. 323–326.

[3] E. H. Adelson and J. R. Bergen, "The plenoptic function and the elements of early vision," in *Computational Models of Visual Processing.* MIT Press, 1991, pp. 3–20.

[4] I. Bauermann and E. Steinbach, "RDTC optimized compression of image-based scene representations (part I): Modeling and theoretical analysis," *IEEE Transactions on Image Processing*, vol. 17, no. 5, pp. 709–723, May 2008.

[5] I. Bayram and I. W. Selesnick, "On the dual-tree complex wavelet packet and m -band transforms," *IEEE Transactions on Signal Processing*, vol. 56, no. 6, pp. 2298 –2310, june 2008.

[6] J. Berent, "Coherent multi-dimensional segmentation of multi-view images using a variational framework and applications to image based rendering," *PhD Thesis*, 2008, Imperial College. [Online]. Available: http://www.commsp.ee.ic.ac.uk/∼pld/group/

[7] J. Berent and P. L. Dragotti, "Segmentation of epipolar-plane image volumes with occlusion and disocclusion competition," in *Proceedings of IEEE Workshop on Multimedia Signal Processing (MMSP)*, Oct. 2006, pp. 182–185.

[8] ——, “Plenoptic manifolds: Exploiting structure and coherence in multiview images,” *IEEE Signal Processing Magazine*, vol. 24, no. 6, pp. 34–44, Nov. 2007.

[9] J. Berent, P. L. Dragotti, and M. Brookes, “Adaptive layer extraction for image based rendering,” in *Proceedings of IEEE Workshop on Multimedia Signal Processing (MMSP)*, Oct. 2009.

[10] T. Blu and F. Luisier, “The SURE-LET approach to image denoising,” *IEEE Transactions on Image Processing*, vol. 16, no. 11, pp. 2778–2786, November 2007.

[11] R. Bolles, H. Baker, and D. Marimont, “Epipolar-plane image analysis: An approach to determining structure from motion,” *International Journal of Computer Vision*, vol. 1, pp. 7–55, March 1987.

[12] A. M. Bruckstein, D. L. Donoho, and M. Elad, “From sparse solutions of systems of equations to sparse modeling of signals and images,” *SIAM Review*, vol. 51, no. 1, pp. 34–81, 2009.

[13] E. J. Candès and D. L. Donoho, “Curvelets: a surprisingly effective nonadaptive representation of objects with edges,” in *Curve and Surface Fitting: Saint-Malo.* University Press, 2000.

[14] V. Caselles, R. Kimmel, and G. Sapiro, “Geodesic active contours,” *International Journal of Computer Vision*, vol. 22, pp. 61–79, February 1997.

[15] J. X. Chai, X. Tong, S. C. Chan, and H. Y. Shum, “Plenoptic sampling,” in *Proceedings of Computer Graphics (SIGGRAPH).* ACM Press/Addison-Wesley Publishing Co., 2000, pp. 307–318.

[16] V. Chaisinthop, “Centralized and distributed semi-parametric compression of piecewise smooth function,” *PhD Thesis*, 2010, Imperial College. [Online]. Available: http://www.commsp.ee.ic.ac.uk/∼pld/group/

[17] V. Chaisinthop and P. L. Dragotti, “Centralized and distributed semiparametric compression of piecewise smooth functions,” *IEEE Transactions on Signal Processing*, vol. 59, no. 7, pp. 3071–3085, July 2011.

[18] C.-L. Chang, X. Zhu, P. Ramanathan, and B. Girod, “Shape adaptation for light field compression,” in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, vol. 1, Sept. 2003, pp. 765–768.

[19] G.-C. Chang and W.-N. Lie, “Multi-view image compression and intermediate view synthesis for stereoscopic applications,” in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 2, May 2000, pp. 277–280.

[20] Y. Chen and W. A. Pearlman, “Three-dimensional subband coding of video using the zero-tree method,” in *Proceedings of SPIE 2727 Visual Communication and Image Processing*, Mar. 1996, pp. 1302–1309.

[21] G. Cheung, A. Ortega, and N.-M. Cheung, “Interactive streaming of stored multiview video using redundant frame structures,” *IEEE Transactions on Image Processing*, vol. 20, no. 3, pp. 744–761, Mar. 2011.

[22] ——, “Generation of redundant frame structure for interactive multiview streaming,” in *Proceedings of the Internation Packet Video Workshop*, May 2009, pp. 1–10.

[23] N.-M. Cheung, A. Ortega, and G. Cheung, “Rate-distortion based reconstruction optimization in distributed source coding for interactive multiview video streaming,” in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, Sept. 2010, pp. 3721 –3724.

[24] C. Christopoulos, A. Skodras, and T. Ebrahimi, “The JPEG2000 still image coding system: an overview,” *IEEE Transactions on Consumer Electronics*, vol. 46, no. 4, pp. 1103–1127, Nov. 2000.

[25] T. M. Cover and J. A. Thomas, *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, 2006.

[26] I. Daribo, C. Tillier, and B. Pesquet-Popescu, “Motion vector sharing and bitrate allocation for 3D video-plus-depth coding,” *EURASIP Journal on Advances in Signal Processing*, vol. 2009, pp. 1–13, 2009.

[27] I. Daubechies and W. Sweldens, "Factoring wavelet transforms into lifting steps," *Journal of Fourier Analysis and Applications*, vol. 4, no. 3, pp. 247–269, 1998.

[28] D. Divsilar and F. Pollara, "On the design of turbo codes," in *Technical Report TDA 42-123*, Nov. 1995.

[29] M. N. Do and M. Vetterli, "The finite ridgelet transform for image representation," *IEEE Transactions on Image Processing*, vol. 12, no. 1, pp. 16–28, Jan. 2003.

[30] ——, "The contourlet transform: an efficient directional multiresolution image representation," *IEEE Transactions on Image Processing*, vol. 14, no. 12, pp. 2091 –2106, Dec. 2005.

[31] D. Donoho and I. M. Johnstone, "Adapting to unknown smoothness via wavelet shrinkage," *Journal of the American Statistical Association*, vol. 90, pp. 1200–1224, 1995.

[32] P. L. Dragotti and M. Gastpar, *Distributed Source Coding: Theory, Algorithms and Applications.* Academic Press, 2009.

[33] J. E. Fowler, "Qccpack: An open-source software library for quantization, compression, and coding," in *Proceedings of SPIE Applications of Digital Image Processing XXIII*, August 2000, pp. 294–301.

[34] H. Freeman, "On the encoding of arbitrary geometric configurations," *IEEE Transactions on Electronic Computers*, vol. EC-10, no. 2, pp. 260–268, Jun. 1961.

[35] A. Gelman, P. L. Dragotti, and V. Velisavljević, "Multiview image compression using a layer-based representation," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, Sep. 2010.

[36] A. Gersho and R. M. Gray, *Vector quantization and signal compression.* Norwell, MA, USA: Kluwer Academic Publishers, 1991.

[37] B. Girod, C.-L. Chang, P. Ramanathan, and X. Zhu, "Light field compression using disparity-compensated lifting," in *Proceedings of the IEEE International*

Conference on Acoustics, Speech, and Signal Processing (ICASSP), vol. 4, Apr. 2003, pp. IV–760–3.

[38] B. Girod and M. Magnor, "Two approaches to incorporate approximate geometry into multi-view image coding," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, vol. 2, Sep. 2000, pp. 5–8.

[39] B. Girod, C.-L. Chang, P. Ramanathan, and X. Zhu, "Light field compression using disparity-compensated lifting," *IEEE Transactions on Image Processing*, vol. 15, no. 4, pp. 793–806, Apr. 2006.

[40] S. J. Gortlerm, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, "The lumigraph," in *Proceedings of Computer Graphics (SIGGRAPH)*. ACM, 1996, pp. 43–54.

[41] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.

[42] M. Hötter, "Object-oriented analysis-synthesis coding based on moving two-dimensional objects," *Signal Processing: Image Communication*, vol. 2, no. 4, pp. 409–428, 1990.

[43] S. Jehan-Besson, M. Barlaud, and G. Aubert, "DREAM2S: Deformable regions driven by an Eulerian accurate minimization method for image and video segmentation," *International Journal of Computer Vision*, vol. 53, no. 1, pp. 45–70, Jun. 2003.

[44] M. Johnson, K. Dale, S. Avidan, H. Pfister, W. Freeman, and W. Matusik, "CG2Real: Improving the realism of computer generated images using a large collection of photographs," *IEEE Transactions on Visualization and Computer Graphics*, pp. 1273–1285, Sept. 2011.

[45] M. Karczewicz and R. Kurceren, "The SP- and SI-frames design for H.264/AVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 637 – 644, July 2003.

[46] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988.

[47] M. H. Kiu, X. S. Du, R. J. Moorhead, D. C. Banks, and R. Machiraju, "Two dimensional sequence compression using MPEG," *in Visual Communication and Image Processing (VCIP)*, pp. 914–921, Jan. 1998.

[48] V. Kolmogorov and R. Zabih, "Multi-camera scene reconstruction via graph cuts," in *Proceedings of the 7th European Conference on Computer Vision-Part III (ECCV).* Springer-Verlag, 2002, pp. 82–96.

[49] A. Kubota, A. Smolic, M. Magnor, M. Tanimoto, T. Chen, and C. Zhang, "Multiview imaging And 3DTV," *IEEE Signal Processing Magazine*, vol. 24, no. 6, pp. 10–21, November 2007.

[50] E. Le Pennec and S. Mallat, "Bandelet image approximation and compression," *SIAM Journal of Multiscale Modeling and Simulation*, vol. 4, no. 3, pp. 992–1039, 2005.

[51] ——, "Sparse geometric image representations with bandelets," *IEEE Transactions on Image Processing*, vol. 14, no. 4, pp. 423 –438, april 2005.

[52] M. Levoy and P. Hanrahan, "Light field rendering," in *Proceedings of Computer Graphics (SIGGRAPH).* ACM, Aug. 1996, pp. 31–42.

[53] S. Li and W. Li, "Shape-adaptive discrete wavelet transforms for arbitrarily shaped visual object coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 5, pp. 725–743, Aug. 2000.

[54] Y. Liu and B. Zalik, "An efficient chain code with huffman coding," in *Pattern Recognition*, vol. 38, no. 4, Apr. 2005, pp. 553–557.

[55] M. Magnor and P. Eisert, "Model-aided coding of multi-viewpoint image data," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, vol. 2, Sep. 2000, pp. 919–922.

[56] M. Magnor, A. Endmann, and B. Girod, "Progressive compression and rendering of light fields," in *Proceedings of the Vision, Modelling and Visualization*, Nov. 2000, pp. 199–203.

[57] M. Magnor and B. Girod, "Hierarchical coding of light fields with disparity maps," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, vol. 3, Oct. 1999, pp. 334–338.

[58] ——, "Model-based coding of multi-viewpoint imagery," in *Proceedings SPIE Visual Communications and Image Processing (VCIP)*, vol. 1, Jun. 2000, pp. 14–22.

[59] ——, "Adaptive block-based light field coding," in *Proceedings of International Workshop on Synthetic and Natural Hybrid Coding and Three-Dimensional Imaging*, Sept. 1999, pp. 140–143.

[60] M. Maitre, Y. Shinagawa, and M. N. Do, "Wavelet-based joint estimation and encoding of depth-image-based representations for free-viewpoint rendering," *IEEE Transactions on Image Processing*, vol. 17, no. 6, pp. 946–957, June 2008.

[61] S. Mallat, *A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way*, 3rd ed. Academic Press, 2008.

[62] L. McMillan and G. Bishop, "Plenoptic modeling: An image-based rendering system," 1995. [Online]. Available: citeseer.comp.nus.edu.sg/17738.html

[63] M. Nelson. Arithmetic coding + statistical modeling = data compression. [Online]. Available: http://marknelson.us/1991/02/01/arithmetic-coding-statistical-modeling-data-compression/

[64] J.-R. Ohm, "Advances in scalable video coding," *Proceedings of the IEEE*, vol. 93, no. 1, pp. 42–56, Jan. 2005.

[65] J. Pearson, P. L. Dragotti, and M. Brookes, "Accurate non-iterative depth layer extraction algorithm for image based rendering," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2011, pp. 901–904.

[66] P. Pérez, M. Gangnet, and A. Blake, "Poisson image editing," in *Proceedings of Computer Graphics (SIGGRAPH).* ACM, 2003, pp. 313–318.

[67] S. Pradhan and K. Ramchandran, "Distributed source coding using syndromes (discus): design and construction," *IEEE Transactions on Information Theory*, vol. 49, no. 3, pp. 626 – 643, Mar 2003.

[68] F. Racape, S. Lefort, E. Francois, M. Babel, and O. Deforges, "Adaptive pixel/patch-based synthesis for texture compression," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, Sept. 2011, pp. 609–612.

[69] P. Ramanathan and B. Girod, "Random access for compressed light fields using multiple representations," in *Proceedings of IEEE Workshop on Multimedia Signal Processing (MMSP)*, Sept.-1 Oct. 2004, pp. 383–386.

[70] P. Ramanathan, M. Kalman, and B. Girod, "Rate-distortion optimized streaming of compressed light fields," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, vol. 3, Sept. 2003, pp. III–277–80.

[71] P. Ramanathan and B. Girod, "Rate-distortion analysis for light field coding and streaming," *Signal Processing: Image Communication*, vol. 21, pp. 462–475, 2006.

[72] W. Ryan, "An introductionto LDPC codes," in *CRC Handbook for Coding and Signal Processing for Recording Systems (B.Vasic,ed.).* CRC Press, 2004.

[73] X. San, H. Cai, J.-G. Lou, and J. Li, "Multiview image coding based on geometric prediction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 11, pp. 1536 –1548, Nov. 2007.

[74] D. Scharstein and R. Szeliski. Middlebury data sets. [Online]. Available: http://www.vision.middlebury.edu/stereo/

[75] A. Secker and D. Taubman, "Lifting-based invertible motion adaptive transform (LIMAT) framework for highly scalable video compression," *IEEE Transactions on Image Processing*, vol. 12, no. 12, pp. 1530–1542, Dec. 2003.

[76] I. W. Selesnick, R. G. Baraniuk, and N. G. Kingsbury, "The dual-tree complex wavelet transform," *IEEE Signal Processing Magazine*, vol. 22, no. 6, pp. 123–151, Nov. 2005.

[77] S.-H. Seo, M. Azimi-Sadjadi, and B. Tian, "A least-squares-based 2-D filtering scheme for stereo image compression," *Image Processing, IEEE Transactions on*, vol. 9, no. 11, pp. 1967 –1972, Nov. 2000.

[78] J. A. Sethian, *Level Set Methods.* Cambridge University Press, 1996.

[79] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 36, no. 9, Sep. 1988, pp. 1445–1453.

[80] R. Shukla, P. L. Dragotti, M. N. Do, and M. Vetterli, "Rate-distortion optimized tree-structured compression algorithms for piecewise polynomial images," *IEEE Transactions on Image Processing*, vol. 14, no. 3, pp. 343–359, Mar. 2005.

[81] H. Y. Shum, S. C. Chan, and S. B. Kang, *Image-Based Rendering.* Springer-Verlag, 2007.

[82] H. Y. Shum and S. B. Kang, "A review of image-based rendering techniques," *Proceedings SPIE Visual Communications and Image Processing (VCIP)*, vol. 213, pp. 1–12, 2000.

[83] G. J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Processing Magazine*, vol. 15, no. 6, pp. 74–90, Nov. 1998.

[84] M. Tanimoto, "Overview of free viewpoint television," *Signal Processing: Image Communication*, vol. 21, no. 6, pp. 454–61, 2006.

[85] D. Taubman, "High performance scalable image compression with EBCOT," *IEEE Transactions on Image Processing*, vol. 9, no. 7, pp. 1158–1170, Jul. 2000.

[86] D. Taubman and A. Zakhor, "Multirate 3-d subband coding of video," *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 572–588, Sep 1994.

[87] M. Unser and T. Blu, “Mathematical properties of the JPEG2000 wavelet filters,” *IEEE Transactions on Image Processing*, vol. 12, no. 9, pp. 1080 – 1090, Sept. 2003.

[88] V. Velisavljevic, B. Beferull-Lozano, M. Vetterli, and P. L. Dragotti, “Directionlets: anisotropic multidirectional representation with separable filtering,” *IEEE Transactions on Image Processing*, vol. 15, no. 7, pp. 1916 –1933, July 2006.

[89] M. Vetterli and J. Kovacevic, *Wavelets and Subband Coding.* Prentice Hall, 1995.

[90] J. Wang, J. Y. A. Wang, Edward, and H. Adelson, “Representing moving images with layers,” *IEEE Transactions on Image Processing*, vol. 3, pp. 625–638, 1994.

[91] Z. Wang and A. C. Bovik, “Mean squared error: Love it or leave it? a new look at signal fidelity measures,” *IEEE Signal Processing Magazine*, vol. 26, no. 1, pp. 98 –117, Jan 2009.

[92] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, “Overview of the H.264/AVC video coding standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, July 2003.

[93] I. H. Witten, R. M. Neal, and J. G. Cleary, “Arithmetic coding for data compression,” *Communications of the ACM*, vol. 30, no. 6, pp. 520–540, 1987.

[94] W. Woo and A. Ortega, “Stereo image compression with disparity compensation using the mrf model,” in *Proceedings SPIE Visual Communications and Image Processing (VCIP)*, 1996, pp. 28–41.

[95] J. Xu, Z. Xiong, and S. Li, “High performance wavelet-based stereo image coding,” in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, vol. 2, 2002, pp. 273–276.

[96] C. Zhang and J. Li, “Compression of lumigraph with multiple reference frame (MRF) prediction and just-in-time rendering,” in *Proceedings of the IEEE Data Compression Conference (DCC),*, 2000, pp. 253–262.

[97] C. Zhang and T. Chen, “A survey on image-based rendering–representation, sampling and compression,” *Signal Processing: Image Communication*, vol. 19, no. 1, pp. 1–28, 2004.