ANALYSIS AND DEVELOPMENT OF REVERSIBLE DATA HIDING SCHEME BASED ON PIXEL VALUE ORDERING

SULTAN ABDUL HASIB
BSc Engg. (EEE), RUET
Student No - 1014160003 (F)

A THESIS SUBMITTED FOR THE DEGREE OF

MASTER OF SCIENCE
IN
ELECTRICAL, ELECTRONIC AND COMMUNICATION ENGINEERING

DEPARTMENT OF
ELECTRICAL, ELECTRONIC AND COMMUNICATION ENGINEERING
MILITARY INSTITUTE OF SCIENCE AND TECHNOLOGY
MIRPUR CANTONMENT, DHAKA 1216

2019

APPROVAL CERTIFICATE

The thesis titled "**Analysis and Development of Reversible Data Hiding Scheme based on Pixel Value Ordering**" submitted by Sultan Abdul Hasib, Roll No: 1014160003 (F), Session: 2014-2015 has been accepted as satisfactory in partial fulfillment of the requirements for the degree of Master of Science in Electrical, Electronic and Communication Engineering on 28 May 2019.

BOARD OF EXAMINERS

1. _____ Chairman
   Lieutenant Colonel Hussain Md. Abu Nyeem, PhD, EME (Supervisor)
   Instructor Class A
   Department of EECE, MIST, Dhaka - 1216

2. _____ Member
   Brigadier General A K M Nazrul Islam, PhD (Ex-officio)
   Head
   Department of EECE, MIST, Dhaka - 1216

3. _____ Member
   Lieutenant Colonel Md Tawfiq Amin, PhD, EME (Internal)
   Instructor Class A
   Department of EECE, MIST, Dhaka - 1216

4. _____ Member
   Dr. Satya Prasad Majumder (External)
   Professor
   Department of EEE, BUET, Dhaka - 1205

CANDIDATE'S DECLARATION

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

Signature: _____

(Sultan Abdul Hasib)

Date: May 2019

DEDICATION

To my parents and family

ACKNOWLEDGEMENTS

Foremost, I would like to express my sincere gratitude to my supervisor, Lieutenant Colonel Hussain Md. Abu Nyeem, PhD, EME of the Department of EECE at MIST, for the useful comments, remarks and engagement through the learning process of this master thesis. I always received appropriate guidelines, whenever I ran into a trouble spot or had a question about my research or writing. He consistently steered me in the right direction to make original contribution in this thesis.

I would also like to thank the instructors of the EECE departments, who were very helpful to provide me any materials and other support, whenever required for the successful completion of my research.

I would also like to extend my heartfelt gratitude to the reviewers and experts in the area of my research work for their valuable and constructive feedback and assistance on my research publications.

I also thank the post-graduate course coordinator, my course teachers and staffs of EECE department, MIST for their kind support. I will remain indebted for the support I received from the department and MIST to complete this work.

Finally, I must express my profound gratitude to my parents and to my family for providing me with relentless support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

SULTAN ABDUL HASIB

Military Institute of Science and Technology

Dhaka, Bangladesh

May 2019

CONTENTS

ABSTRACT

Reversible Data hiding (RDH) is an evolving forensic and covert-communication technology that embeds data into a cover image (or other media like video or audio) so that the embedded data later can be extracted for the copyright protection, integrity establishment or annotation. Developing such a scheme with better rate-distortion performance is challenging since a higher embedding rate usually causes more distortion in the embedded image. Recently, the pixel-value-ordering (PVO) based RDH schemes have shown better rate-distortion performance so far. However, the existing PVO-based RDH schemes have not considered a suitable scanning order in a kernel that can further improve the embedding rate-distortion performance.

This thesis, therefore, contributes to the development of a PVO-based RDH scheme with a new PVO-kernel and backward embedding technique. Firstly, a new triangular kernel is proposed that captures the pixels correlated in the horizontal, vertical and diagonal directions simultaneously. The proposed kernel is employed in a prominent PVO-based RDH scheme and is verified for a better (or occasionally similar) rate-distortion performance than the existing schemes that rely on only the column- or row-kernel. Additionally, a new backward embedding technique is introduced to counterbalance the distortion caused in a forward embedding phase.

Besides, the computational modeling, and the evaluation, analysis, and validation of the new PVO-based RDH scheme are presented in the thesis. The simulation results has demonstrated a promising performance of the proposed scheme and its improvement over the popular and state-of-the-art PVO-based RDH schemes. Particularly, a significantly better rate-distortion performance is obtained at the higher embedding rate, which means the proposed scheme is more promising to the applications with a high embedding capacity requirement like electronic patient record hiding in medical images. Moreover, the proposed RDH scheme developed using the new kernel and backward-embedding would create a new paradigm of RDH for the future data hiding research.

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

LIST OF ABBREVIATIONS

DE	Difference Expansion
dPVO	dual Pixel Value Ordering
HS	Histogram Shifting
LSB	Least Significant Bit
MSB	Most Significant Bit
MSE	Mean Squire Error
NMI	Neighbor Mean Interpolation
NNI	Nearest Neighbor Interpolation
PEE	Prediction Error Expansion
PG	Pixel Grouping
PVO	Pixel Value Ordering
PSNR	Peak Signal to Noise Ratio
RCM	Reversible Contrast Matching
RDH	Reversible Data Hiding
SSIM	Structural Similarity

LIST OF SYMBOLS

I	An input image
$\hat{I}$	An embedded version of the input image, I
M	Number of pixels in a row: $M \in \mathbb{N}$
N	Number of pixels in a column: $N \in \mathbb{N}$
(i, j)	A pixel position such that $i = (1, 2, \cdots, M)$ and $j = (1, 2, \cdots, N)$
n	Total number of pixels in a block of an image
x	A pixel in a block of input image
$\hat{x}$	Embedded version of x
ψ	An image partitioning kernel that outputs image-blocks
X	An input image-block of n pixels such that $X = (x_1, x_2, \cdots, x_n)$
$\hat{X}$	An embedded version of X
k	An index of an image block such that $k = \frac{M \times N}{n}$
X_k	The k^{th} block of an input image, I such that $I = [X_k]$
$\hat{X}_k$	The k^{th} block of an embedded image, $\hat{I}$
$\sigma(\cdot)$	A function that sorts the pixels of a block in ascending order
$x_{\sigma(1)}$	The smallest pixel in an input image-block
$\hat{x}_{\sigma(1)}$	An embedded version of $x_{\sigma(1)}$
$x_{\sigma(n)}$	The largest pixel in an input image-block
$\hat{x}_{\sigma(n)}$	An embedded version of $x_{\sigma(n)}$
e	A prediction error in an image-block
$\hat{e}$	An expanded version of e
E_n	A block of predicted errors of an input image
$\hat{E}_n$	An expanded version of E_n
P_n	An input image-block of sorted pixels
$\hat{P}_n$	An embedded version of P_n
b	A data bit such that $b \in \{1, 0\}$
D	The set of data bits to be embedded such that $D = \{b\}$

CHAPTER 1

INTRODUCTION

1.1 Introduction

Multimedia has a remarkably growing impact on today's life-hood, society, research, and industry. A broad spectrum of emerging multimedia applications has been showing new promises in different areas of medical, space, military, security and surveillance. At the same time, with the integration of communication technologies, exchange of multimedia information is also simultaneously raising many security concerns including forgery, copyright violation and privacy invasion of multimedia information. To this end, different Reversible Data Hiding (RDH) schemes are widely investigated [1, 2].

The RDH is an evolving forensic and covert-communication technology that embeds data into a cover media like an image, video or audio so that the embedded data later can be extracted for the cover media's authentication, copyright protection, integrity establishment or annotation. A better data hiding scheme has to offer better embedding rate-distortion performance. Since a higher data embedding rate naturally causes higher embedding distortion, achieving better rate-distortion performance is a challenging task.

Generally, an RDH scheme has three main processes: *generation*, *embedding* and *extraction* [3, 4]. A general framework of an RDH scheme is illustrated in Fig. 1.1. In the *generation*, required secret-data are generated for an intended application such as content authentication, copyright protection, annotation, *etc*. The *embedding* process, on the other hand, deals with how and where the data are to be embedded in the cover media. The *extraction* process later extracts the embedded data and completely

recovers the original media.

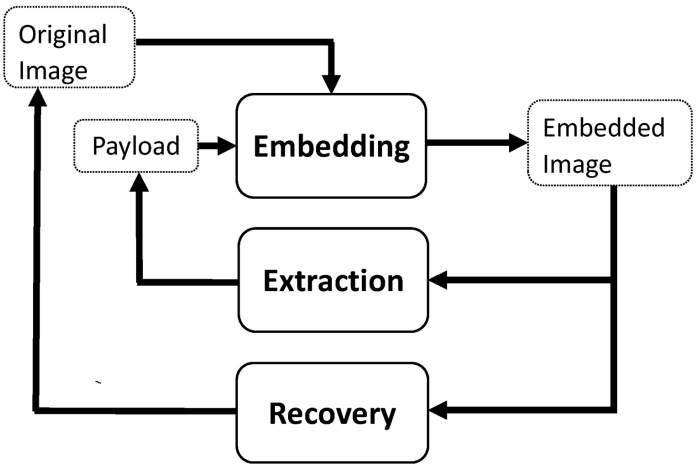


Figure 1.1: A general RDH framework.

1.2 Literature Review

The development of an RDH scheme is being steered by a better rate-distortion performance. In other words, the higher embedding capacity with an invertible and minimum possible distortion (or better image quality) is an attractive RDH property. For example, the pioneering difference expansion (DE)-based RDH scheme [5] with invertible distortion was immediately improved for higher capacity with generalized expansion [6], reduced location map [7, 8], sorting and prediction [9, 10], and adaptive embedding [11, 12]. On the other hand, for minimizing distortion, the histogram shifting (HS)-based scheme [13] was improved using the difference-histogram [14–16] and multiple histograms [17, 18]. Other potential developments include the RDH schemes with prediction error expansion (PEE) [19–38], vector quantization [39, 40], interpolation [41, 42], encryption [43, 44], and transform techniques [45, 46].

Among the varieties of RDH principle, PEE is much investigated for its efficient rate-distortion performance. Generally, prediction errors are computed from a set of local pixels from a reference pixel. An error histogram is computed from the prediction-

errors as such the high frequency bins can be expanded for embedding data. The other bins are shifted accordingly to make room for an invertible expansion. Each expanded prediction-error thus can embed 1 bit without much affecting the original pixels. Such reversible expansion of prediction errors thus made a significant step forward in reversible data hiding research.

Particularly, a PEE-based RDH scheme can better utilize the combined principles of DE and HS to expand prediction errors for data hiding. Unlike the use of pixel-histogram in basic HS, it deals with the prediction errors to obtain a much sharper histogram with a set of higher peak beans resulting in higher embedding capacity. Additionally, unlike the direct change of pixels in basic DE, it expands the prediction errors to offer minimum possible changes in the pixels resulting in higher quality embedded images. Further developments of the PEE-based schemes can also be tracked with the context modification [20,21,47], prediction error classification [22–24], adaptive block size [25,26], two-dimensional histogram modification [28], pair-wise PEE [27,35] and pixel value ordering [19, 29–34, 36–38].

Pixel value ordering (PVO) has been an easy solution to minimize the prediction errors in PEE. With PVO, pixel values are ordered in a group or block of a pixels for better utilization of the local-pixels correlation. Although the pixel value selection, grouping and sorting principles for prediction were utilized in [9, 19], the principle of PVO was well established by Li *et al*. [29]. That scheme predicted the maximum-minimum pixel pairs to embed with lower distortion. Peng *et al*. [30] improved the PVO with a new histogram-modification principle. Ou *et al*. [31] extended the basic PVO to PVO-k for adaptive embedding in the blocks according to the numbers of maximum- and minimum-valued pixels. Unlike the block-wise prediction in the original PVO, Qu *et al*. [32] then extended it to be a pixel-wise for a larger capacity and better image fidelity. Wang *et al*. [33] introduced a dynamic partitioning of blocks for PVO according to the block complexity to improve the embedding capacity.

Other recent developments include multiple histograms modification [48], pair-wise PEE [34] and multi-pass PVO [38].

For the performance evaluation and validation of the PVO-based RDH schemes, a conventional approach is generally used. The embedding rate-distortion performance of an RDH scheme is evaluated in terms of embedding capacity, embedding rate, and popular visual degradation metric. For example, the embedding capacity is the total number of bits embedded in an input image. The embedding rate is measured by bit-per-pixel (*bpp*), which is a ratio of the embedding capacity and the total number of pixels in the input image. For quantifying visual degradation in the embedded images, PSNR–*peak signal to noise ratio* and SSIM–*structural similarity* [49] are being widely used. Overall rate-distortion performance of a new or improved scheme is also validated for the test image database: USC-SIPI [50] and Kodak [51].

In summary, the PVO-based RDH schemes mentioned above demonstrated a better rate-distortion performance for lower embedding capacity requirement. However, their rate-distortion performances sharply decrease with higher embedding rate. Their maximum embedding capacity limits are also lower and they mostly rely on the complex and recursive embedding conditions. On any higher requirement of image fidelity and/or embedding capacity, their computational complexities thus grow significantly. Therefore, it is necessary and makes sense to investigate a simple, yet effective technique of utilizing PVO for both the better image quality and higher embedding rate, which the research presented in this thesis is particularly aimed at.

1.3 Research Motivation

While much development in the area of RDH has been reported in the last few years as discussed above, there is still need for investigation to further develop a PVO-based embedding technique for an RDH scheme. Development of an RDH scheme may have

several considerations, for example, design requirements and performance requirements. Design requirements include how a PVO-based embedding technique would be designed, where the data would be embedded and how it affects the original pixels. Performance requirements include computational complexity of the processes and the visual quality of the embedded images and the embedding capacity (*i.e.* the rate-distortion performance). To be more specific to the research problem, the following research question can be posed: '*can a PVO-based RDH scheme be developed to embed more data with a better image quality?*'

The above question leads us to address two main challenges: developing a suitable kernel for PVO and utilizing it in developing an embedding. To address these challenges, a few sub-questions also arise as follows: (*i*) *how can a better kernel be developed for a PVO-based embedding to obtain the best possible embedded image quality?* and (*ii*) *can the prediction errors in a PVO-based embedding be counterbalaced to further minimize the distortion*? The first sub-question entails an investigation among the existing PVO-based embedding techniques aiming at identifying a suitable kernel for the best possible embedded image quality. Particularly, a kernel that captures the pixels in the maximum neighborhood context would contribute to maintain a better embedded image quality.

Moreover, with the second sub-question, improvement of the embedded image quality is investigated. The distortion introduced in a PVO-based embedding will be studied in an additional level of embedding as such the distortion can be counterbalanced successively. Such an additional embedding would also improve the embedding capacity. It is also required to verify that the proposed embedding would not introduce a significant computational overhead.

1.4 Research Objectives

In light of the identified gap in the area of the RDH, the research presented in this thesis sets its primary goal to develop a PVO-based embedding technique and utilize it in modeling a new RDH scheme as mentioned in the previous section. To carry out the project, the specific objectives of this work are outlined as follows.

a) To investigate the all possible scanning orders and kernel shapes used in local image processing, to determine a set of scanning orders and kernel shapes promising for an RDH scheme.

b) To analyze the embedding rate-distortion performance of existing PVO-based RDH schemes for the selected scanning orders and kernels to determine a suitable scanning order and a kernel that offer higher embedding capacity and better embedded image quality.

c) To develop algorithms of the PVO, data embedding and extraction processes for complete formulation of the proposed PVO-based RDH scheme.

d) To develop an experimental setup for the performance evaluation and benchmarking of the proposed scheme to demonstrate the expected improvement in embedding rate-distortion performance for different images.

The expected outcome of this work is, therefore, a new PVO-based RDH scheme to offer better embedding rate-distortion performance for digital image applications.

1.5 Organization of the Thesis

The remainder of this thesis is organized as follows.

Chapter 2 captures the background of the proposed research of this dissertation. With an overview of the prominent PVO-based RDH schemes, their associated techniques are reviewed and the key potentials and limitations are studied.

The scopes of improvement in the PVO-based RDH schemes are investigated. Thereby, a further study towards developing a new PVO-based embedding technique and its use in an RDH scheme is incorporated.

Chapter 3 presents the proposed development of an RDH scheme based on a new kernel of PVO and backward embedding. With specific algorithmic details of the kernel, and embedding and extraction processes, the new scheme is illustrated with necessary technical notes, example and figures.

Chapter 4 presents the evaluation of the proposed scheme for its embedding rate - distortion performance. The performance of the proposed scheme is validated in this chapter with high capacity and better image quality compared to prominent RDH schemes. The presented analysis verified the potential of the proposed scheme leaving it as a promising candidate for different applications.

Chapter 5 presents the conclusion of the thesis with a summary of the original contributions and future work.

CHAPTER 2

PVO-BASED RDH SCHEMES

2.1 Introduction

This chapter summarized the recent developments of the RDH schemes related to the work presented in this thesis. Before describing those schemes, the main focus of this work is mentioned here. For example, this research presents in this thesis mainly focuses on the applications of RDH schemes on the natural digital image for the development of a PVO-based RDH model. However the developing model of this work may be extended to other media applications without loss of generality.

Additionally, with conventional trend of improving an RDH scheme, the performance development, evaluation, analysis, and validation are mainly concerned with the embedding capacity and embedded image quality, as conventionally done in the literature. Any data, images, processes, algorithms, schemes or methods considered in this thesis either are digital themselves, or deal with digital inputs and outputs. However, for readability, the word *digital* is often omitted in writing. For example, *digital image* is merely written as *image*, respectively without change of their meaning and context.

The basic principle of PVO [29] and its other successive developments leading to the Jung's minimum block PVO based RDH scheme [36] is now briefly introduced here and in the following sections. A generalized framework of PVO-based embedding is first presented below to define the principle of PVO and its improvements in existing schemes [29–31, 33, 34, 38].

A PVO-based embedding generally starts with partitioning a cover image I of size $M \times N$ into a set of non-overlapping blocks, *i.e.*, $I = [X_k]$. With each block containing n pixels, *i.e.*, $X = (x_1, x_2, \cdots x_n)$, total number of blocks is $k = \frac{M \times N}{n}$. For each block

X, its block pixels, $(x_1, x_2, \cdots x_n)$ are now sorted in ascending order using a sorting function $\sigma(\cdot)$ to output $(x_{\sigma(1)}, x_{\sigma(2)}, \cdots x_{\sigma(n)})$. The function, $\sigma : \{1, 2, \cdots n\} \rightarrow \{1, 2, \cdots n\}$ is a unique one-to-one mapping such that $x_{\sigma(1)} \leq x_{\sigma(2)} \leq \cdots \leq x_{\sigma(n)}$ with $\sigma(i) < \sigma(j)$ if $x_{\sigma(i)} = x_{\sigma(j)}$ and $i < j$. Once the block pixels are sorted, they are used for prediction with a suitable PEE-based embedding of $data$ leading to the development of different PVO-based schemes as follows. Without loss generality, we illustrate different PEE-based embedding conditions below for a data bit, $b \in \{0, 1\}$.

2.2 Li et al.'s PVO-based RDH Scheme

Li *et al*. [29] proposed to use the second maximum block-pixel, *i.e*., $x_{\sigma(n-1)}$ to predict the maximum $x_{\sigma(n)}$. The prediction error, e is computed using (2.1). PEE-based embedding is then carried out with the histogram of e. Only bin 1 with $e = 1$ is used for embedding of a data-bit, b, and other bins are expanded with the condition of higher values of e in (2.2a). With the modified error, $\hat{e}$, the maximum block-pixel is updated accordingly as in Eq (2.2c).

$$e = x_{\sigma(n)} - x_{\sigma(n-1)} \tag{2.1}$$

$$\hat{e} = \begin{cases} e, & \text{if } e = 0 \\ e + b, & \text{if } e = 1 \\ e + 1, & \text{if } e > 1 \end{cases} \tag{2.2a}$$

$$\hat{x}_{\sigma(n)} = x_{\sigma(n-1)} + \hat{e} \tag{2.2b}$$

$$= \begin{cases} x_{\sigma(n)}, & \text{if } e = 0 \\ x_{\sigma(n)} + b, & \text{if } e = 1 \\ x_{\sigma(n)} + 1, & \text{if } e > 1 \end{cases} \tag{2.2c}$$

This process of PEE embedding repeats for all the blocks (until the last data-bit is embedded assuming the embedding capacity requirement is attained) to output an embedded image, $\hat{I}$. It is apparent that this embedding does not change the pixel value order after embedding to ensure perfect data extraction and lossless recovery.

Embedded data are extracted with the inverse embedding conditions. Specifically, with the embedded image, $\hat{I}$, image blocks $\hat{X}_k$ are obtained. For each block, $\hat{X}$, the prediction errors are re-generated using (2.3a). The extracted data-bits and the original pixels are obtained using the reverse PEE conditions in (2.3).

$$\hat{e} = \hat{x}_{\sigma(n)} - \hat{x}_{\sigma(n-1)} \tag{2.3a}$$

$$\text{if } \hat{e} \in \{1, 2\}: \begin{cases} b = \hat{e} - 1 \\ \\ x_{\sigma(n)} = \hat{x}_{\sigma(n)} - b \end{cases} \tag{2.3b}$$

$$\text{if } \hat{e} > 2: \; x_{\sigma(n)} = \hat{x}_{\sigma(n)} - 1 \tag{2.3c}$$

$$\text{if } \hat{e} = 0: \; x_{\sigma(n)} = \hat{x}_{\sigma(n)} \tag{2.3d}$$

With the maximum possible change to a pixel value by 1, the embedded image quality also remains high. With the consideration of minimum block-pixels, this basic PVO-based embedding is further improved in [30, 36].

2.3 Peng et al.'s Improved PVO-based Scheme

Peng *et al*. [30] extended the principle of Li *et al*.'s PVO-to utilize both bins 0 and 1 for embedding. This is achieved by redefining the prediction error with the consideration of location information, *i.e*., the spatial order of the maximum and second maximum

block-pixels are used. The prediction error is thus redefined here as:

$$e = x_u - x_v \tag{2.4}$$

where $u = min\left(\sigma\left(n\right),\ \sigma\left(n-1\right)\right)$ and $v = max\left(\sigma\left(n\right),\ \sigma\left(n-1\right)\right)$. With this new definition of e in Peng *et al.*'s scheme, the value of e can no longer be non-negative, and the bins 0 and 1 are used for embedding as in (2.5a). Finally, the maximum block-pixel is modified with (2.5b), where $x_{\sigma(n-1)}$ is always kept unchanged and $x_{\sigma(n)}$ is either increased or unchanged after embedding.

$$\hat{e} = \begin{cases} e + b, & \text{if } e = 1 \\ e + 1, & \text{if } e > 1 \\ e - b, & \text{if } e = 0 \\ e - 1, & \text{if } e < 0 \end{cases} \tag{2.5a}$$

$$\begin{aligned} \hat{x}_{\sigma(n)} =& x_{\sigma(n-1)} + |\hat{e}| \\ =& \begin{cases} x_{\sigma(n)} + b, & \text{if } e \in \{0, 1\} \\ x_{\sigma(n)} + 1, & \text{otherwise} \end{cases} \end{aligned} \tag{2.5b}$$

$$\hat{e} = \hat{x}_u - \hat{x}_v \tag{2.6a}$$

$$\text{if } \hat{e} > 0 \ :\ u = \sigma(n) \ \&\ v = \sigma(n-1) \tag{2.6b}$$

$$\begin{aligned} &\text{if } \ \hat{e} \in \{1, 2\} \ :\ b = \hat{e} - 1 \ \ \&\ \ x_{\sigma(n)} = \hat{x}_u - b \\ &\text{else} \qquad\qquad :\ x_{\sigma(n)} = \hat{x}_u - 1 \end{aligned} \tag{2.6c}$$

$$\text{else} \ :\ u = \sigma(n-1) \ \&\ v = \sigma(n) \tag{2.6d}$$

$$\begin{aligned} &\text{if } \hat{e} \in \{0,-1\} \ : \ b = -\hat{e} \ \ \& \ \ x_{\sigma(n)} = \hat{x}_v - b \\ &\text{else} \qquad\qquad\quad : \ x_{\sigma(n)} = \hat{x}_v - 1 \end{aligned} \tag{2.6e}$$

This process continues for embedding all the data bits in a block to output an embedded image, $\hat{I}$. For the extraction of embedded data-bits and recovery of the original image, the expanded prediction errors are regenerated from the block-pixels for each block of $\hat{I}$. With that $\hat{e}$ in (2.6a), the inverse PEE embedding conditions in (2.6b) to (2.6e) are followed to extract the data-bit, b and the original block-pixels, $x_{\sigma(n)}$ from each block, $\hat{X}$. This is detailed in Peng *et al.*'s scheme [30].

Data embedding in the minimum block-pixels can also be used as the above embedding in the maximum block-pixels as shown in [30]. However, with the computation of new prediction error and its use to embed in both the bins 0 and 1, Peng *et al.* improved the embedding capacity of the Li *et al.*'s original PVO-based RDH scheme with a reasonably better-embedded image quality.

2.4 Jung's Minimum PVO-based RDH Scheme

Jung [36] recently proposed a minimal case of the PVO-based RDH scheme to embed 2 bits in an image-block of size 1×3. Particularly, an image I is partitioned into a set blocks of three pixels as such $I = [X_k]$ with $k = \{1, 2, \cdots \frac{M \times N}{3}\}$. This means that, with the general PVO framework presented at the beginning of this section, Jung's scheme operates on each block X with the number of block-pixels, $n = 3$. Thus the sorting function, $\sigma(\cdot)$ is used to sort the block-pixels (x_1, x_2, x_3) to be $(x_{\sigma(1)}, x_{\sigma(2)}, x_{\sigma(3)})$, where $x_{\sigma(3)}$ and $x_{\sigma(1)}$ are the maximum and minimum block-pixels, respectively. A pair of prediction errors, e_{max} and e_{min} for each block is calculated from the middle block-pixel, $x_{\sigma(2)}$ according to the (2.7a) and (2.7b). These errors are

expanded with the embedding of a data-bit, b or shifting by the value 1 with (2.7c) and (2.7d). The maximum and minimum block-pixels are then predicted from the middle block-pixel and the expanded errors with (2.7e) and (2.7f), respectively.

$$e_{max} = x_{\sigma(3)} - x_{\sigma(2)} \tag{2.7a}$$

$$e_{min} = x_{\sigma(1)} - x_{\sigma(2)} \tag{2.7b}$$

$$\hat{e}_{max} = \begin{cases} e_{max}, & \text{if } e_{max} = 0 \\ e_{max} + b, & \text{if } e_{max} = 1 \\ e_{max} + 1, & \text{if } e_{max} > 1 \end{cases} \tag{2.7c}$$

$$\hat{e}_{min} = \begin{cases} e_{min}, & \text{if } e_{min} = \ 0 \\ e_{min} - b, & \text{if } e_{min} = -1 \\ e_{min} - 1, & \text{if } e_{min} < -1 \end{cases} \tag{2.7d}$$

$$\hat{x}_{\sigma(3)} = x_{\sigma(2)} + \hat{e}_{max} \tag{2.7e}$$

$$\hat{x}_{\sigma(1)} = x_{\sigma(2)} + \hat{e}_{min} \tag{2.7f}$$

The data extraction and original block-pixels' recovery follow the inverse PEE embedding principle of the Jung's scheme in (2.8) like other PVO-based RDH schemes. With the recovery of the maximum and minimum block-pixels of all expanded pixels, the original image is recovered. At the same time, the data-bits are extracted from each embedded blocks and concatenated to get the original data.

With a single reference pixel in a block, Jung's scheme predicts the maximum and minimum block-pixels as such in every three pixels of an image, two bits of data can be embedded. Thus, the embedding capacity is improved with a reasonably good embedded image quality. However, the overall embedding rate-distortion perfor-

mance at lower embedding rate is still much lower than the advanced PVO-based RDH schemes [32, 34, 38]. In this paper, a more effective use of the Jung's PVO is investigated and thus the development of a higher capacity RDH scheme with a competitive visual quality of an embedded image is presented in the section below.

$$b = \begin{cases} \hat{e}_{max} - 1, & \text{if } 1 \leq \hat{e}_{max} \leq 2 \\ -\hat{e}_{min} - 1, & \text{if } -2 \leq \hat{e}_{min} \leq -1 \end{cases} \tag{2.8a}$$

$$x_{\sigma(3)} = \begin{cases} \hat{x}_{\sigma(3)}, & \text{if } \hat{e}_{max} = 0 \\ \hat{x}_{\sigma(3)} - b, & \text{if } 1 \leq \hat{e}_{max} \leq 2 \\ \hat{x}_{\sigma(3)} - 1, & \text{if } \hat{e}_{max} > 2 \end{cases} \tag{2.8b}$$

$$x_{\sigma(2)} = \hat{x}_{\sigma(2)} \tag{2.8c}$$

$$x_{\sigma(1)} = \begin{cases} \hat{x}_{\sigma(1)}, & \text{if } \hat{e}_{min} = 0 \\ \hat{x}_{\sigma(1)} + b, & \text{if } -2 \leq \hat{e}_{min} \leq -1 \\ \hat{x}_{\sigma(1)} + 1, & \text{if } \hat{e}_{min} < -2 \end{cases} \tag{2.8d}$$

2.5 Chapter Summary

This chapter presented a generalized computational model of the PVO-based embedding and its development in the popular RDH schemes. In such schemes, while pixel values are grouped and arranged in a numerical order to better utilize their correlations for improving the embedded image quality, not much attention has been paid in computation of PG with better pixel correlation. Additionally, no effort has been made to counterbalance the embedding distortion by an additional level of embedding, which has been addressed in the next chapter.

CHAPTER 3

A NEW PVO-BASED DATA HIDING SCHEME

3.1 Introduction

This chapter presents a new scheme with an improved kernel and backward embedding. Particularly, with the mixed neighborhood (*i.e.*, horizontal, vertical and diagonal) pixels, an image block of size 3×2 (or 2×3) that creates a pair of pixels-trios of triangular shape possesses a greater possible correlation among the pixels. In Section 3.2, the proposed kernel is used in modeling a new PVO-based RDH scheme for computing image blocks. The computational model of the PVO-based RDH scheme thus captures the principle of both the Jung's scheme [36] and the proposed new kernels.

Additionally, the proposed RDH scheme also introduces a *dual pixel value ordering* (dPVO) with prediction error expansion (PEE) in Section 3.3. The scheme is modeled with two phases of embedding; namely, (*i*) forward embedding with PVO and PEE, and (*ii*) backward embedding with dPVO and pairwise-PEE. These two phases of embedding are expected to improve both the embedding rate and visual quality of the embedded image.

3.2 The Proposed Kernels for PVO

A PVO-based embedding utilizes image correlations for minimum embedding distortion with higher possible embedding capacity as mentioned in the previous section. Particularly, embedding of the Jung's scheme computes the unit prediction error in a non-overlapping block with horizontal neighborhood pixels. This consideration however restricts the embedding distortion controlled by the inter-pixel correlations of a block. Thus, the consideration of defining blocks with a better inter-correlated

pixels may demonstrate a better embedding rate-distortion performance.

To this end, the embedding performance of the proposed PVO-based RDH scheme is investigated for different orientations of non-overlapping image blocks. Unlike the consideration of only horizontal neighborhood pixels to constitute image blocks with the Jung's RDH scheme [36], in this chapter, a set of kernels are proposed to define more suitable image blocks. The image blocks of size 1×3 for horizontal neighborhood pixels' correlation and 3×1 for vertical neighborhood pixels' correlation are shown in Fig. 3.1. These are the simplest possible orientations for a kernel, which has been used in the existing PVO embedding techniques. However, this is obvious that these kernels do not capture correlations in the context of nearest neighborhood pixels.

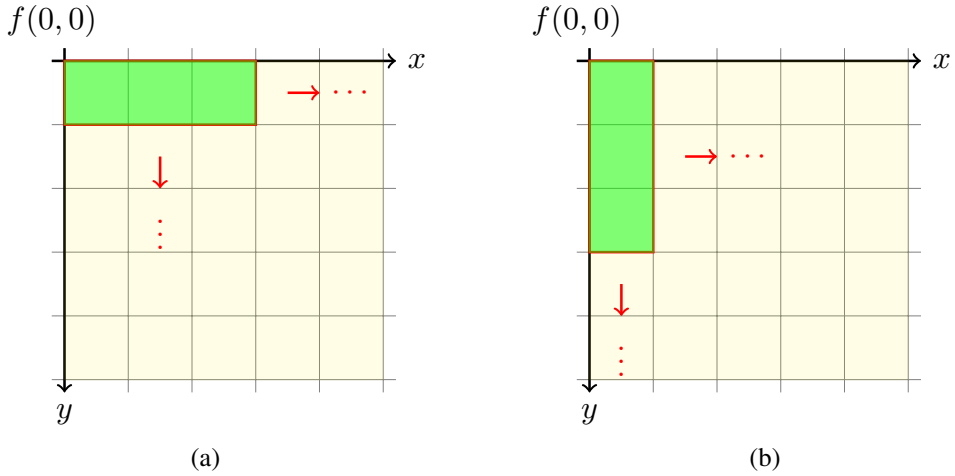
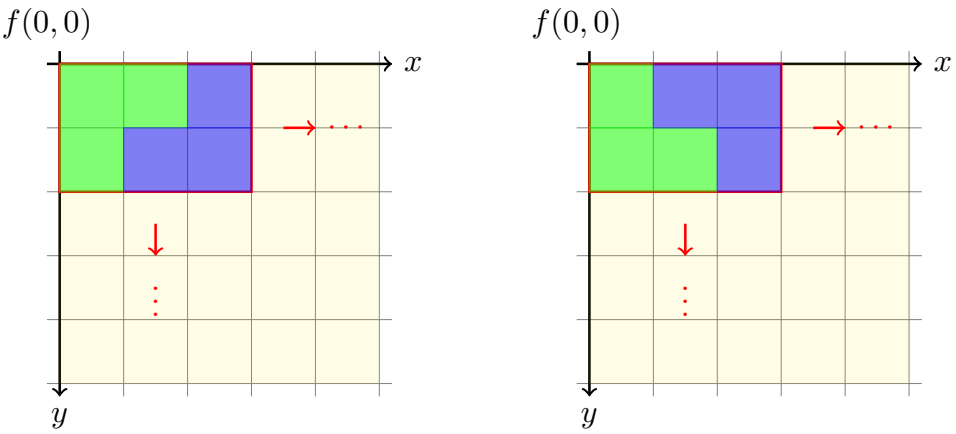
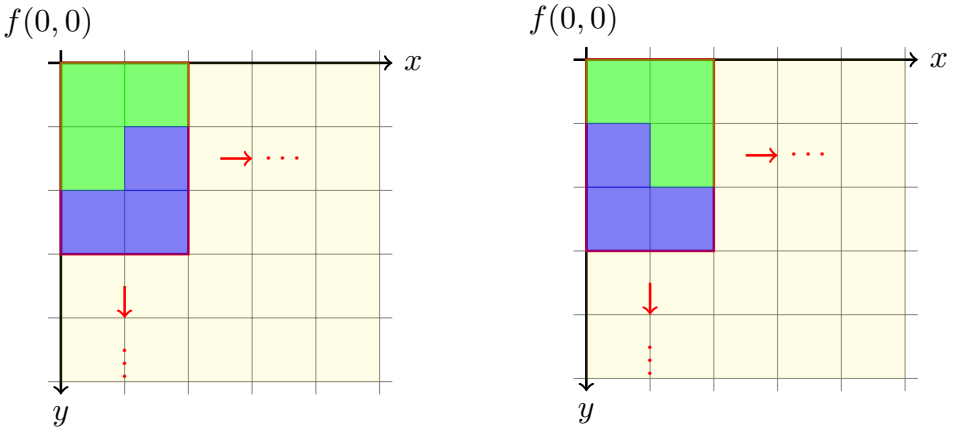


Figure 3.1: Kernels for PVO-based RDH scheme: (a) 3×1 and (b) 1×3. Jung [36] used the kernel in (a) for their PVO embedding.

In order to better exploit the nearest neighborhood correlations, an improved kernel may therefore be developed. We propose here a number of candidates to improve the existing kernel definition, for example, with the blocks of size 2×3 and 3×2 for mixed (*e.g.*, horizontal, vertical and diagonal) neighborhood pixels' correlations. As illustrated in Fig. 3.2, these image blocks of size 3×2 and 2×3 give a pair of pixel-trios. In other words, each block is constituted of a pair of structures with three pixels that form two triangular shapes.



(a) centered (b) non-centered



(c) centered (d) non-centered

Figure 3.2: The proposed kernels for PVO-based RDH scheme with the possible orientations of (a,b) 2×3 and (c,d) 3×2.

For simplicity, these kernels can also be distinguished with the structures of trios in the proposed kernels. For example, see the structures in Fig. 3.2 (a and c), where centers of both the upper trios capture the first pixel of an image at position $f(0, 0)$. With this convention of structure, the kernels can be classified as the *centered* kernel. The other kernels that do not capture the $f(0, 0)$ pixel in the center position of a trio, can be classified as *non-centered* as illustrated in Fig. 3.2 (b and d). In a particular kernel, either 3×2 or 2×3, such structures may have a trivial variation on the embedding effectiveness. Thus, we only focus on the performance of our proposed kernels disregarding their *centered* or *non-centered* classes, for which the proposed scheme would

better exploit the pixel correlations in an image block.

In the proposed scheme, a suitable kernel, ψ is thus utilized in image blocking and de-blocking functions, $block\,(\cdot)$ and $de\_block\,(\cdot)$ with an additional input argument ψ (see Section 3.2.1). That is, for Jung's scheme, $\psi = [1, 3]$ is defined for a block of size 1×3, which has been investigated with $\psi = [3, 1], [3, 2]$ and $[2, 3]$ for a block of size 3×1, 3×2 and 2×3, respectively. Thus, the presented PVO-based embedding and extraction models are expected to offer better embedding rate-distortion performance with a suitable ψ.

3.2.1 PVO-based embedding with the proposed kernel

Let an image I of size $M \times N$ is to be given as input (or *cover*) image and used for embedding of secret-data D. The embedding process follows the following steps to output the embedded image $\hat{I}$. As in the Algorithm 1, steps of the embedding are discussed below.

Algorithm 1: PVO Embedding

Input: an image I, block-size ψ, and payload D
Output: an embedded image $\hat{I}$
$\{B_n\} \leftarrow block\,(I, \psi)$ {n is total no. of blocks}
for all B_n **do**
    $P_n \leftarrow sort\,(B_n)$
    $E_n \leftarrow predict\,(P_n)$
    $\hat{P}_n \leftarrow embed\,(P_n, E_n, d)$
    $\hat{P}_n \leftarrow inverse\_sort\left(\hat{P}_n\right)$
end for
$\hat{I} \leftarrow de\_block\left(\hat{B}_n\right)$
return $\hat{I}$

Step 1: $B \leftarrow block\,(I,\, \psi)$

▷ Input image I is divided into a set of non-overlapping blocks B by the function $block\,(\cdot)$ with a given block-size ψ (*e.g.*, $\psi = 1 \times 3$ in the Jung's scheme) such that $B = \{B_n\}$, where B_n is a set of three pixels of n^{th} block. That is, $B_n = \{b_n^i, b_n^{i+1}, b_n^{i+2}\}$ with $i \in \{1, 2, \cdots, M \times N\}$ for $n \in \{1, 2, \cdots, \frac{M \times N}{3}\}$.

Step 2: $P_n \leftarrow sort\,(B_n)$

▷ A set of three sorted pixels of n-th block P_n is obtained by the block-wise sorting function $sort\,(\cdot)$ for each B_n. That is, $P_n = \{p_n^i, p_n^{i+1}, p_n^{i+2}\}$, where $p_n^i \leq p_n^{i+1} \leq p_n^{i+2}$.

Step 3: $E_n \leftarrow predict\,(P_n)$

▷ For each sorted block P_n, the function $predict\,(\cdot)$ outputs a set of predicted errors E_n. That is, for each P_n, predicted error $E_n = \{e_n^{max}, e_n^{min}\}$ of n-th block is obtained as per (3.1).

$$e_n^{max} = p_n^{i+2} - p_n^{i+1} \tag{3.1a}$$

$$e_n^{min} = p_n^{i} - p_n^{i+1} \tag{3.1b}$$

Step 4: $\hat{P}_n \leftarrow embed\,(P_n, E_n, d)$

▷ A pair of predicted errors e_n^{max} and e_n^{min} of n-th block gets embedded by the secret bits $d \in D$ or expanded by unit value as in (3.2) and (3.3) to obtain the modified errors, $\hat{e}_n^{max}$ and $\hat{e}_n^{min}$ for each n-th block. These modified errors are then used as per (3.4) to compute the set of estimated pixels $\hat{P}_n = \{\hat{p}_n^i, p_n^{i+1}, \hat{p}_n^{i+2}\}$.

$$\hat{e}_n^{max} = \begin{cases} e_n^{max}, & \text{for } e_n^{max} = 0 \\ e_n^{max} + d, & \text{for } e_n^{max} = 1 \\ e_n^{max} + 1, & \text{for } e_n^{max} > 1 \end{cases} \tag{3.2}$$

$$\hat{e}_n^{min} = \begin{cases} e_n^{min}, & \text{for } e_n^{min} = \ 0 \\ e_n^{min} - d, & \text{for } e_n^{min} = -1 \\ e_n^{min} - 1, & \text{for } e_n^{min} < -1 \end{cases} \tag{3.3}$$

$$\hat{p}_n^{i+2} = p_n^{i+1} + \hat{e}_n^{max} \tag{3.4a}$$

$$\hat{p}_n^{i} = p_n^{i+1} + \hat{e}_n^{min} \tag{3.4b}$$

Step 5: $\hat{B}_n \leftarrow inverse\_sort\left(\hat{P}_n\right)$

▷ Relocate the sorted embedded pixels of each block to their original locations.

Step 6: $\hat{I} \leftarrow de\_block\left(\hat{B}_n,\ \psi\right)$

▷ Rearrange each of the embedded blocks to return the embedded image.

3.2.2 PVO-based extraction

The PVO-based extraction process follows similar steps with inverse computation of embedding in Algorithm 2, which are briefly discussed below. The embedded image $\hat{I}$ and block-size ψ are given as inputs to obtain the original image I and extracted data D.

Step 1: $\hat{B} \leftarrow block\left(\hat{I},\ \psi\right)$

Algorithm 2: PVO Extraction

Input: an embedded image $\hat{I}$
Output: original image I and extracted payload D
Initialize: $D \leftarrow \varnothing$
$\psi \leftarrow blocksize\left(\hat{I}\right)$
$\{\hat{B}_n\} \leftarrow block\left(\hat{I}, \psi\right)$
for all $\hat{B}_n$ **do**
    $\hat{P}_n \leftarrow sort\left(\hat{B}_n\right)$
    $\hat{E}_n \leftarrow predict\left(\hat{P}_n\right)$
    $(P_n, d) \leftarrow extract\left(\hat{P}_n, \hat{E}_n\right)$
    $D \leftarrow concat\,(D, d)$
    $B_n \leftarrow inverse\_sort\,(P_n)$
end for
$I \leftarrow de\_block\,(B_n)$
return I, D

▷ The embedded image $\hat{I}$ is divided into a set of non-overlapping blocks $\hat{B}$ by the function $block\,(\cdot)$ with block size of ψ such that $\hat{B} = \{\hat{B}_n\}$, where $\hat{B}_n$ is a set of three pixels of n-th block. That is, $\hat{B}_n = \{\hat{b}_n^i, \hat{b}_n^{i+1}, \hat{b}_n^{i+2}\}$ with $i \in \{1, 2, \cdots,\ M \times N\}$ for $n \in \{1, 2, \cdots,\ \frac{M \times N}{3}\}$.

Step 2: $\hat{P}_n \leftarrow sort\left(\hat{B}_n\right)$

▷ A set of three sorted pixels of n-th block $\hat{P}_n$ is obtained by the block-wise sorting function $sort\,(\cdot)$ for each $\hat{B}_n$. That is, $\hat{P}_n = \{\hat{p}_n^i, \hat{p}_n^{i+1}, \hat{p}_n^{i+2}\}$, where $\hat{p}_n^i \leq \hat{p}_n^{i+1} \leq \hat{p}_n^{i+2}$.

Step 3: $\hat{E}_n \leftarrow predict\left(\hat{P}_n\right)$

▷ For all sorted block-pixels $\{\hat{P}_n\}$, the function $predict\,(\cdot)$ outputs a set of predicted errors $\{\hat{E}_n\}$. That is, for each $\hat{P}_n$, a pair of predicted errors $\hat{E}_n = \{\hat{e}_n^{max}, \hat{e}_n^{min}\}$ is obtained as per (3.5).

$$\hat{e}_n^{max} = \hat{p}_n^{i+2} - \hat{p}_n^{i+1} \tag{3.5a}$$

$$\hat{e}_n^{min} = \hat{p}_n^{i} - \hat{p}_n^{i+1} \tag{3.5b}$$

Step 4: $(P_n, d) \leftarrow extract\left(\hat{P}_n, \hat{E}_n\right)$

▷ An embedded secret bit $d \in D$ is extracted from the embedded blocks $\hat{P}_n$ and the pair of embedded/expanded predicted errors $\hat{e}_n^{max}$ and $\hat{e}_n^{min}$ of $\hat{E}_n$ as in (3.6) for each block. These modified errors are also used to compute the set of original sorted pixels $P_n = \{p_n^i, p_n^{i+1}, p_n^{i+2}\}$ as per (3.7). It may be noted that this extraction function is computationally inverse of the embedding function such that $extract\,(\cdot) = embed^{-1}\,(\cdot)$.

$$d = \begin{cases} \hat{e}_n^{max} - 1, & \text{for } 1 \leq \hat{e}_n^{max} \leq 2 \\ -\hat{e}_n^{min} - 1, & \text{for } -2 \leq \hat{e}_n^{min} \leq -1 \end{cases} \tag{3.6}$$

$$p_n^{i+2} = \begin{cases} \hat{p}_n^{i+2}, & \text{for } \hat{e}_n^{max} = 0 \\ \hat{p}_n^{i+2} - d, & \text{for } 1 \leq \hat{e}_n^{max} \leq 2 \\ \hat{p}_n^{i+2} - 1, & \text{for } \hat{e}_n^{max} > 2 \end{cases} \tag{3.7a}$$

$$p_n^{i+1} = \hat{p}_n^{i+1} \tag{3.7b}$$

$$p_n^{i} = \begin{cases} \hat{p}_n^{i}, & \text{for } \hat{e}_n^{min} = 0 \\ \hat{p}_n^{i} + d, & \text{for } -2 \leq \hat{e}_n^{min} \leq -1 \\ \hat{p}_n^{i} + 1, & \text{for } \hat{e}_n^{min} < -2 \end{cases} \tag{3.7c}$$

Step 5: $D \leftarrow concat\,(D, d)$

▷ The extracted data, D is initialized as an empty set as in line 3 of the Algorithm 2. A set of extracted bits from an embedded block are then concatenated with D and this continues with the extraction process until the last embedded bit is extracted.

Step 6: $B_n \leftarrow inverse\_sort\,(P_n)$

▷ Relocate the sorted embedded pixels of each block to their original locations.

Step 7: $I \leftarrow de\_block\,(B_n,\ \psi)$

▷ Rearrange each restored image-blocks to return the original image. Finally, return I and the extracted data D.

3.3 A New dPVO for the Proposed RDH Scheme

A PVO-based embedding has evolved to utilize image correlations for a better possible rate-distortion performance. With the classic PVO, pixel values in a block are kept unchanged or expanded (either for embedding or shifting) centering the reference pixel(s). This principle of embedding has been better utilized with the adaptive block size or multilevel embedding in the recent schemes for a better rate-distortion performance. However, expanded pixels have not been considered yet for a reverse expansion to restore them to their respective original pixel values partially as mentioned in the last chapter. In this chapter, a reverse expansion property is therefore introduced with the 'backward embedding' to demonstrate how it can further improve the rate-distortion performance.

Embedding of the proposed RDH scheme constitutes two phases: (*i*) forward embedding with PVO and PEE, and (*ii*) backward embedding with dPVO and

pairwise-PEE. These two phases are explained below and expected to improve both the visual quality of the embedded image and the embedding rate. With the utilization of two PVO, the proposed embedding is called here a dual PVO (dPVO) based embedding. Extraction of the proposed scheme, on the other hand, follows the inverse processing of this dPVO-based embedding.

3.3.1 Forward embedding

The forward embedding with PEE employs the Jung's PVO-based embedding that starts with partitioning an input image, I into a set of non-overlapping blocks of size 1×3. This is discussed in Section 2.4. Each block-pixels (x_1, x_2, x_3) are sorted to obtain $(x_{\sigma(1)}, x_{\sigma(2)}, x_{\sigma(3)})$, where $x_{\sigma(1)}$ and $x_{\sigma(3)}$ are the minimum and maximum block-pixels, respectively. With the computation and expansion of the pair of prediction errors, e_{min} and e_{max} using (2.7a) and (2.7d), either a data-bit, b is embedded or error-value is shifted by 1. The minimum and maximum block-pixels are then predicted from the middle block-pixel and the expanded errors with (2.7e) and (2.7f).

The overflow and underflow problem is tackled with the conventional process of recording a location map M_k for the k-th block (for example, see Ref. [37, 38]). The map is initialized as an empty-set for each k-th block, and for each pixel of the block, either a '1' for a boundary pixel or '0' for any other pixel is appended to M_k. Continuing this for all k, a complete location map $M_{ou} = \{M_k\}$ is obtained, compressed and appended to the embedding data-bits. With an input image of bit-depth 8-bit, for example, a boundary pixel, x in a block is then updated using (3.8).

$$x = \begin{cases} x - 1, & \text{if } x = 255 \\ x + 1, & \text{if } x = 0 \\ x, & \text{otherwise} \end{cases} \tag{3.8}$$

Given the input image, I and a set of data-bits, D_f, with this forward embedding, the embedded image, $\hat{I}$ is thus obtained. For simplicity, the denoting difference between the original input image and its pre-processed version with modified boundary pixels is omitted in the presented model without loss of generality.

3.3.2 Backward embedding

This phase of embedding operates on $\hat{I}$ and aims to counterbalance the expansion made in the forward embedding. It is obvious that the maximum and minimum pixels of a block, *i.e.* $x_{\sigma(1)}$ and $x_{\sigma(3)}$, can experience a maximum expansion of value 1 either for embedding of a data-bit '1' or for shifting the pixel by the value 1. Thus, all the predicted pixels that experience this expansion are considered for minimum and maximum groups, $\hat{X}_{min}$ and $\hat{X}_{max}$, respectively. Additionally, the predicted pixels that remain unchanged in the first phase of embedding is located by recording their location map in LM. Computing of $\hat{X}_{min}$ and $\hat{X}_{max}$ is presented in Algorithm 3.

Particularly, the proposed dPVO first separates two sets of pixels expanded in the forward embedding. With a classic PVO on a pixel block of size 1×3, the lowest and highest pixels remain in the same order after their left-and right-ward expansion. So, the lowest and highest predicted pixels of all blocks can be separated in two sets as such applying a pairwise PVO based backward embedding on these two sets individually can restore their original pixel values. For example, in the forward embedding, an embedded image, $\hat{I}$ is obtained with the Jung's scheme (see Section 2.4). In the backward embedding with dPVO, a set $\hat{X}_{min}$ is computed with the lowest predicted pixels, $\{x_{\sigma(1)}\}$ of all blocks $[\hat{X}_k]$. Similarly, another set $\hat{X}_{max}$ captures the largest predicted pixels, $\{x_{\sigma(3)}\}$ of all the blocks. All pairs of pixel values in each of these two sets, $\hat{X}_{min}$ and $\hat{X}_{max}$ then follow a pairwise PVO-based PEE for embedding.

Once $\hat{X}_{min}$ and $\hat{X}_{max}$ are obtained, the pixels of each set in the backward embedding are pairwise expanded. In other words, both $\hat{X}_{min}$ and $\hat{X}_{max}$ are individually

Algorithm 3: $dPVO \cdot encode\,(\cdot)$

Input: $\hat{I}$
Output: $\hat{X}_{max}, \hat{X}_{min}, LM$

$\hat{X}_{min} \leftarrow empty$
$\hat{X}_{max} \leftarrow empty$
$LM \leftarrow zeros\,(2, k)$
$(M, N) \leftarrow size\left(\hat{I}\right)$
for all $k = 1$ to $\frac{M \times N}{3}$ **do**
  $[\hat{x}^k_{\sigma(1)}, \hat{x}^k_{\sigma(2)}, \hat{x}^k_{\sigma(3)}] \leftarrow sort\left([\hat{x}^k_1, \hat{x}^k_2, \hat{x}^k_3] \in \hat{I}\right)$
  if $\hat{x}^k_{\sigma(2)} - \hat{x}^k_{\sigma(1)} > 1$ **then**
    $\hat{X}_{min} \leftarrow append\left(\hat{X}_{min}, \hat{x}^k_{\sigma(1)}\right)$
  else
    $LM\,(1, k) \leftarrow 1$
  end if
  if $\hat{x}^k_{\sigma(3)} - \hat{x}^k_{\sigma(2)} > 1$ **then**
    $\hat{X}_{max} \leftarrow append\left(\hat{X}_{max}, \hat{x}^k_{\sigma(3)}\right)$
  else
    $LM\,(2, k) \leftarrow 1$
  end if
end for
return $\hat{X}_{max}, \hat{X}_{min}, LM$

pairwise partitioned, sorted and used for embedding.

$$e_{xmin} = \hat{x}_{\sigma(2)} - \hat{x}_{\sigma(1)} | (\hat{x}_{\sigma(1)}, \hat{x}_{\sigma(2)}) \in \hat{X}_{min} \tag{3.9a}$$

$$\hat{e}_{xmin} = \begin{cases} e_{xmin}, & \text{if } e_{xmin} = 0 \\ e_{xmin} + b, & \text{if } e_{xmin} = 1 \\ e_{xmin} + 1, & \text{if } e_{xmin} > 1 \end{cases} \tag{3.9b}$$

$$\hat{\hat{x}}_{\sigma(2)} = \hat{x}_{\sigma(1)} + \hat{e}_{xmin} | \hat{\hat{x}}_{\sigma(2)} \in \hat{\hat{X}}_{min} \tag{3.9c}$$

$$e_{xmax} = \hat{x}_{\sigma(1)} - \hat{x}_{\sigma(2)} | (\hat{x}_{\sigma(1)}, \hat{x}_{\sigma(2)}) \in \hat{X}_{max} \tag{3.10a}$$

$$\hat{e}_{xmax} = \begin{cases} e_{xmax}, & \text{if } e_{xmax} = 0 \\ e_{xmax} - b, & \text{if } e_{xmax} = -1 \\ e_{xmax} - 1, & \text{if } e_{xmax} > -1 \end{cases} \tag{3.10b}$$

$$\hat{\hat{x}}_{\sigma(1)} = \hat{x}_{\sigma(2)} + \hat{e}_{xmax} | \hat{\hat{x}}_{\sigma(1)} \in \hat{\hat{X}}_{max} \tag{3.10c}$$

For example, a pixel-pair $[\hat{x}_1, \hat{x}_2] \in \hat{X}_{min}$ with sorting becomes $[\hat{x}_{\sigma(1)},\ \hat{x}_{\sigma(2)}]$. These partitioning and sorting also apply to $\hat{X}_{max}$. Then, $\hat{x}_{\sigma(2)}$ is predicted from $\hat{x}_{\sigma(1)}$ for $\hat{X}_{min}$ using (3.9), and predict $\hat{x}_{\sigma(1)}$ from $\hat{x}_{\sigma(2)}$ for $\hat{X}_{max}$ using (3.10). This prediction will increase the value of $\hat{x}_{\sigma(2)} \in \hat{X}_{min}$ by 0 or 1. Since all the pixel values in $\hat{X}_{min}$ have already decreased in the forward embedding by the value of 0 or 1, the backward embedding thus can partially counterbalance the effect of the forward embedding resulting in lower distortion in the embedded image. On the contrary, for applying the backward embedding to the pixel-pairs in $\hat{X}_{max}$, the lower pixel value $\hat{x}_{\sigma(1)}$ is predicted from $\hat{x}_{\sigma(2)}$. Thereby, the set of expanded pixels, $\{\hat{\hat{x}}_{\sigma(2)}\}$ for $\hat{X}_{min}$ and $\{\hat{\hat{x}}_{\sigma(1)}\}$ for $\hat{X}_{max}$ are computed to generate the expanded minimum-and maximum groups, $\hat{\hat{X}}_{min}$ and $\hat{\hat{X}}_{max}$, respectively. The final embedded image, $\hat{\hat{I}}$ is obtained by updating $\hat{I}$ with $\hat{\hat{X}}_{min}$ and $\hat{\hat{X}}_{max}$.

3.3.3 Data extraction and image recovery

Data extraction and image recovery are inverse of embedding of the proposed RDH scheme. This means, data is first extracted with the inverse of backward embedding followed by the inverse of forward embedding. The input image to the decoder is partitioned into a non-overlapping block of size 1×3, and each block pixels are sorted in either ascending or descending order. From the reserved pixels, the location map, LM is extracted. From the sorted pixels of each block and using LM, the sets of maximum and minimum expanded pixels, $\hat{\hat{X}}_{max}$ and $\hat{\hat{X}}_{min}$, respectively are determined

using Algorithm 4.

Algorithm 4: $dPVO \cdot decode\,(\cdot)$

Input: $\hat{\hat{I}}$
Output: $\hat{\hat{X}}_{max}, \hat{\hat{X}}_{min}$
1: $\hat{\hat{X}}_{min} \leftarrow empty$
2: $\hat{\hat{X}}_{max} \leftarrow empty$
3: $LM \leftarrow LMext\left(\hat{\hat{I}}\right)$
4: $(M, N) \leftarrow size\left(\hat{I}\right)$
5: **for** all $k = 1$ to $\frac{M \times N}{3}$ **do**
6: $\quad[\hat{\hat{x}}^k_{\sigma(1)}, \hat{\hat{x}}^k_{\sigma(2)}, \hat{\hat{x}}^k_{\sigma(3)}] \leftarrow sort\left([\hat{\hat{x}}^k_1, \hat{\hat{x}}^k_2, \hat{\hat{x}}^k_3] \in \hat{\hat{I}}\right)$
7: $\quad$**if** $\hat{\hat{x}}^k_{\sigma(2)} - \hat{\hat{x}}^k_{\sigma(1)} > 1 \;\;\&\;\; LM\,(1,k) = 1$ **then**
8: $\quad\quad\hat{\hat{X}}_{min} \leftarrow append\left(\hat{\hat{X}}_{min}, \hat{\hat{x}}^k_{\sigma(1)}\right)$
9: $\quad$**end if**
10: $\quad$**if** $\hat{\hat{x}}^k_{\sigma(3)} - \hat{\hat{x}}^k_{\sigma(2)} > 1 \;\;\&\;\; LM\,(2,k) = 1$ **then**
11: $\quad\quad\hat{\hat{X}}_{max} \leftarrow append\left(\hat{\hat{X}}_{max}, \hat{\hat{x}}^k_{\sigma(3)}\right)$
12: $\quad$**end if**
13: **end for**
14: **return** $\hat{\hat{X}}_{max}, \hat{\hat{X}}_{min}$

Extraction of the embedded data bits, and recovery of $\hat{X}_{min}$ and $\hat{X}_{max}$ from $\hat{\hat{X}}_{min}$ and $\hat{\hat{X}}_{max}$, respectively are carried out using (3.11) and (3.12). The errors from each pixel-pair in $\hat{\hat{X}}_{min}$ and $\hat{\hat{X}}_{max}$ are first computed using (3.11a) and (3.11b), respectively. Embedded bits are extracted using the error-values and conditions in (3.11c). The higher pixel, $\hat{\hat{x}}_{\sigma(2)}$ of each pixel-pair in $\hat{\hat{X}}_{min}$ are restored to $\hat{x}_{\sigma(2)}$ using (3.12a). Similarly, $\hat{x}_{\sigma(1)} \in \hat{X}_{max}$ is restored from $\hat{\hat{x}}_{\sigma(1)} \in \hat{\hat{X}}_{max}$ using (3.12b). Thereby, $\hat{X}_{max}$ and $\hat{X}_{min}$ from $\hat{\hat{X}}_{max}$ and $\hat{\hat{X}}_{min}$, respectively are obtained to finally compute $\hat{I}$ from $\hat{\hat{I}}$.

$$\hat{\hat{e}}_{xmin} = \hat{\hat{x}}_{\sigma(2)} - \hat{\hat{x}}_{\sigma(1)} | (\hat{\hat{x}}_{\sigma(1)}, \hat{\hat{x}}_{\sigma(2)}) \in \hat{\hat{X}}_{min} \tag{3.11a}$$

$$\hat{\hat{e}}_{xmax} = \hat{\hat{x}}_{\sigma(1)} - \hat{\hat{x}}_{\sigma(2)} | (\hat{\hat{x}}_{\sigma(1)}, \hat{\hat{x}}_{\sigma(2)}) \in \hat{\hat{X}}_{max} \tag{3.11b}$$

$$b = \begin{cases} \hat{\hat{e}}_{xmin} - 1, & \text{if } 1 \leq \hat{\hat{e}}_{xmin} \leq 2 \\ -\hat{\hat{e}}_{xmax} - 1, & \text{if } -2 \leq \hat{\hat{e}}_{xmax} \leq -1 \end{cases} \tag{3.11c}$$

For all $\hat{x}_{\sigma(2)} \in \hat{X}_{min}$ & $\hat{\hat{x}}_{\sigma(2)} \in \hat{\hat{X}}_{min}$:

$$\hat{x}_{\sigma(2)} = \begin{cases} \hat{\hat{x}}_{\sigma(2)}, & \text{if } \hat{\hat{e}}_{xmin} = 0 \\ \hat{\hat{x}}_{\sigma(2)} - b, & \text{if } 1 \leq \hat{\hat{e}}_{min} \leq 2 \\ \hat{\hat{x}}_{\sigma(2)} - 1, & \text{if } \hat{\hat{e}}_{min} > 2 \end{cases} \tag{3.12a}$$

For all $\hat{x}_{\sigma(1)} \in \hat{X}_{max}$ & $\hat{\hat{x}}_{\sigma(1)} \in \hat{\hat{X}}_{max}$:

$$\hat{x}_{\sigma(1)} = \begin{cases} \hat{\hat{x}}_{\sigma(1)}, & \text{if } \hat{\hat{e}}_{xmax} = 0 \\ \hat{\hat{x}}_{\sigma(1)} + b, & \text{if } -2 \leq \hat{\hat{e}}_{max} \leq -1 \\ \hat{\hat{x}}_{\sigma(1)} + 1, & \text{if } \hat{\hat{e}}_{max} < -2 \end{cases} \tag{3.12b}$$

The original image is finally restored with the inverse of the first phase embedding, which follows the extraction principle of Jung's scheme (see Section 2.4). This extraction phase starts with partitioning $\hat{I}$ into non-overlapping blocks. For each block, block pixels $[\hat{x}_1, \hat{x}_2, \hat{x}_3]$ are then sorted to $[\hat{x}_{\sigma(1)}, \hat{x}_{\sigma(2)}, \hat{x}_{\sigma(3)}]$. Respective errors, $\hat{e}_{max}$ and $\hat{e}_{min}$ are computed using (3.13) followed by the data-bit extraction and pixel recovery using (2.8b) to (2.8d). Upon the extraction of all data-bits, they are merged to the data-bits extracted in earlier phase. Similarly, the original image I is obtained by updating $\hat{I}$ with the restored values of $\hat{x}_{\sigma(3)}$ and $\hat{x}_{\sigma(1)}$ of each block.

$$\hat{e}_{max} = \hat{x}_{\sigma(3)} - \hat{x}_{\sigma(2)} \tag{3.13a}$$

$$\hat{e}_{min} = \hat{x}_{\sigma(1)} - \hat{x}_{\sigma(2)} \tag{3.13b}$$

3.4 Chapter Summary

The technical details of the proposed RDH scheme with new kernels and dPVO based embedding are presented in this chapter. The proposed scheme has utilized the mixed neighborhood (*i.e.*, horizontal, vertical and diagonal) pixels to define an image block of size 3×2 (or 2×3) with a greater possible correlation among the pixels. Additionally, a dPVO-based embedding is introduced with the prediction error expansion (PEE) and employed in the proposed RDH scheme to counterbalance the distortion made in the first phase of embedding. With this two-phase embedding and a more suitable kernel, the proposed scheme is expected to improve both the embedding rate and visual quality of the embedded image, which will be presented in the next chapter.

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Introduction

This chapter presents the evaluation of the new RDH scheme presented in Chapter 3 for its efficiency and better embedding rate -distortion performance. The experiment is carried out for a set of standard USC-SIPI gray scale test-images [50] of size $512 \times 512 \times 8$, and the Kodak [51] images of sizes $768 \times 512 \times 8$ and $512 \times 768 \times 8$.

The performance of the proposed scheme is validated with high capacity and better image quality compared to the prominent RDH schemes. In Section 4.3, the performance improvement due to the new kernels are evaluated, and in Section 4.4, the overall embedding rate-distortion performance of the proposed RDH scheme is evaluated and compared with a number of popular and recent RDH schemes: He *et al*. (2018) [38], Jung (2017) [36], Ou *et al*. (2016) [34], Wang *et al*. (2015) [33], Qu & Kim (2015) [32], Peng *et al*. (2014) [30] Li *et al*. (2013) [29] and Sachnev *et al*. (2009) [19].

4.2 Evaluation Metrics

The embedded image quality is evaluated in terms of two popular objective visual quality metrics, peak signal to noise ratio (PSNR) defined in (4.1) and structural similarity (SSIM) [49] defined in (4.2). Here, $M \times N$ is the image size, and $I(i,j)$ and $\hat{I}(i,j)$ are the pixel values of position (i,j) in an original image and its embedded version, respectively. In (4.2), μ_x and $\mu_{\hat{x}}$ are the average values of x and $\hat{x}$, where $x \in I$ and $\hat{x} \in \hat{I}$ are the pixels of original and embedded images, respectively. Similarly, σ_x^2 and $\sigma_{\hat{x}}^2$ are the variance of x and $\hat{x}$, respectively; $\sigma_{x\hat{x}}$ is the covariance of x and $\hat{x}$; c_1 and c_2

are two regularization constants, and L is the dynamic range of the pixel values.

$$\text{MSE} = \frac{\sum_{j=1}^{N}\sum_{i=1}^{M}\left(\hat{I}(i,j) - I(i,j)\right)^2}{MN} \tag{4.1a}$$

$$\text{PSNR} = 10\log\frac{L^2}{\text{MSE}} \tag{4.1b}$$

$$\text{SSIM} = \frac{(2\mu_x\mu_{\hat{x}} + c_1)(2\sigma_{x,\hat{x}} + c_2)}{(\mu_x^2 + \mu_{\hat{x}}^2 + c_1)(\sigma_x^2 + \sigma_{\hat{x}}^2 + c_2)} \tag{4.2}$$

The proposed RDH scheme and the existing schemes [19, 29, 30, 32–34, 36, 38] were implemented using MATLAB R2016b with a 1.3 GHz Intel Core i5 CPU, 4 GB memory. The proposed scheme developed in this research can also be re-implemented with the technical details given in Chapter 3. However, it is to note that with the programming skills and platforms, the optimization of the implemented code may vary without losing the embedding rate-distortion performance reported in this thesis.

4.3 Performance of the New Kernels for PVO

The performance of the proposed PVO embedding has been evaluated and compared with the Jung's PVO-based scheme [36]. An example of a set of original test images are shown in Fig. 4.1. Both the embedding-capacity and embedding-rate are determined in terms of total embedded bits and bit per pixels (bpp), respectively. The visual quality of the embedded images is evaluated in terms of the PSNR and SSIM defined in the previous section. For embedding, a set of pseudo-random bits is generated as *data*.

A better embedding rate-distortion performance is observed for PVO embedding with mixed neighborhood pixels. This means that inter-pixel correlations can be better utilized in PVO embedding with blocks of size 2×3 or 3×2 resulting in better embedding rate-distortion performance as illustrated in Table 4.1. For example, the

(a) Airplane (b) Boat (c) Peppers (d) Goldhill

Figure 4.1: Example of original test images (USC-SIPI database [50]).

total embedding capacity of Jung's Scheme is 44992 bits (or 0.1716 bpp) for Airplane image, which is increased to 46547 bits, 46612 bits and 46762 bits (or 0.1776 bpp, 0.1778 bpp and 0.1784 bpp) for 3×1, 2×3 and 3×2 size of image blocks, respectively for the proposed embedding.

Additionally, visual quality of the embedded images has remained at the similar level or slightly improved as evident in Table 4.1. For example, the PSNR and SSIM values of Airplane embedded image are 51.576 dB and 0.9759, respectively. In contrast, for the proposed embedding with 3×1, 2×3 and 3×2, respective PSNR and SSIM values are 51.617 dB and 0.9756, 51.639 dB and 0.9760, and 51.629 dB and 0.9759.

With another example of Peppers image in Table 4.1, embedding capacity is improved from 33.5 kbits to 33.8 kbits by using the kernel of size 3×2 over the Jung's scheme. The PSNR and SSIM values, however, remained almost similar in this case. The images have also demonstrated similar results in improving the embedding rate-distortion performance with the proposed kernels.

The comparison of the rate-distortion performance is summarized in Table 4.2. Like the performance of the proposed scheme for an individual image, the proposed kernels demonstrated similarly better embedded image quality for a given embedding rate and for all the test images and block sizes. In particular, the average embedding capacity achieved with 3×2 size block is 32191 bits and that with a block of size 2×3 is 32228 bits, whereas the capacity is 31223 bits and 31387 bits for the blocks of size

Table 4.1: Comparison of rate-distortion performance

Images	**Metric**	**Jung [36]**	**Proposed**		
		$(\mathbf{1 \times 3})$	$(\mathbf{3 \times 1})$	$(\mathbf{2 \times 3})$	$(\mathbf{3 \times 2})$
Airfield	Capacity (bits)	27307	29414	30333	30104
	bpp	0.1042	0.1122	0.1157	0.1148
	PSNR (dB)	50.756	50.842	50.864	50.862
	SSIM	0.9941	0.9943	0.9943	0.9943
Airplane	Capacity (bits)	44992	46547	46612	46762
	bpp	0.1716	0.1776	0.1778	0.1784
	PSNR (dB)	51.576	51.617	51.639	51.629
	SSIM	0.9759	0.9756	0.9760	0.9759
Baboon	Capacity (bits)	13226	14046	14090	14087
	bpp	0.0505	0.0536	0.0537	0.0537
	PSNR (dB)	50.263	50.283	50.282	50.286
	SSIM	0.9977	0.9977	0.9977	0.9977
Boat	Capacity (bits)	26588	25521	26224	26338
	bpp	0.1014	0.0973	0.1000	0.1005
	PSNR (dB)	50.681	50.6485	50.660	50.666
	SSIM	0.9926	0.9926	0.9925	0.9925
Couple	Capacity (bits)	34494	34968	34882	34596
	bpp	0.1316	0.1334	0.1331	0.1320
	PSNR (dB)	51.016	51.008	50.996	50.985
	SSIM	0.9916	0.9915	0.9915	0.9915
Elaine	Capacity (bits)	23306	23997	24392	24304
	bpp	0.0889	0.0915	0.0930	0.0927
	PSNR (dB)	50.595	50.612	50.633	50.629
	SSIM	0.9929	0.9926	0.9928	0.9928
Goldhill	Capacity (bits)	27021	29365	28280	28573
	bpp	0.1031	0.1120	0.1079	0.1090
	PSNR (dB)	50.688	50.759	50.719	50.730
	SSIM	0.9922	0.9924	0.9923	0.9923
Peppers	Capacity (bits)	33483	31933	33423	33802
	bpp	0.1277	0.1218	0.1275	0.1289
	PSNR (dB)	50.923	50.869	50.914	50.916
	SSIM	0.9887	0.9885	0.9886	0.9886
Tiffany	Capacity (bits)	41750	38807	41864	41680
	bpp	0.1593	0.1480	0.1597	0.1590
	PSNR (dB)	51.316	51.183	51.305	51.303
	SSIM	0.9829	0.9826	0.9829	0.9829

1×3 and 3×1, respectively. This improved embedding capacity also comes with improved average PSNR and SSIM values in case of 2×3 block size. This is because, with these block sizes, all three neighborhood (*i.e.*, horizontal, vertical and diagonal) pixels are present and more correlated than the other blocks (*i.e.*, proposed 3×1 and Jung's 1×3).

Table 4.2: Comparison of average rate-distortion performance

Metric	**Jung [36]**	**Proposed**		
	$(\mathbf{1 \times 3})$	$(\mathbf{3 \times 1})$	$(\mathbf{2 \times 3})$	$(\mathbf{3 \times 2})$
Capacity (bits)	31223	31387	32228	32191
bpp	0.1191	0.1197	0.1229	0.1228
PSNR (dB)	50.921	50.916	50.948	50.944
SSIM	0.9891	0.9890	0.9891	0.9891

With example of original test images and their embedded versions, the improvement of image quality may also be justified. For example, the Airplane image in Fig. 4.2 is embedded with the proposed kernels. The images, while appeared in their full size appearance during experimentation, helped distinguish between their visual quality for the existing technique and that of the proposed one. However, the Fig. 4.2 captured the images with their shrunk sizes due to the space constraints, where the visual quality improvements may not be visually inspected as observed in the result of quantitative performance in Tab. 4.1 and 4.2. Similar observations also hold for the Fig. 4.3 and Fig. 4.4 that are shown to compare the visual image qualities of the embedded images, Peppers and Goldhill, obtained from the Jung's scheme and the proposed scheme, respectively.

A more convincing illustration of the improvement is seen in Fig. 4.5. It is observed that, while performance of the proposed scheme with 3×1 block-size slightly improves over the Jung's scheme, this improvement becomes more noticeable for the

(a) (b) (c) (d)

Figure 4.2: Example of embedded images for the test image 'Airplane' with blocks of different neighborhood pixels: (a) 1×3, (b) 3×1, (c) 2×3 and (d) 3×2.

(a) (b) (c) (d)

Figure 4.3: Example of embedded images for the test image 'Peppers' with blocks of different neighborhood pixels: (a) 1×3, (b) 3×1, (c) 2×3 and (d) 3×2.

(a) (b) (c) (d)

Figure 4.4: Example of embedded images for the test image 'Goldhill' with blocks of different neighborhood pixels: (a) 1×3, (b) 3×1, (c) 2×3 and (d) 3×2.

other proposed block-sizes (*i.e.*, 2×3 and 3×2). For different images including the Airplane, Baboon, Elaine and Goldhill, all the proposed kernels improve the kernel used in the Jung's scheme. Among the proposed kernels, 2×3 and 3×2 generally show a better PSNR over the others for a certain embedding capacity. This means that our proposed embedding scheme with either 2×3 or 3×2 can have better embedding

rate-distortion performance than that of the Jung's scheme.

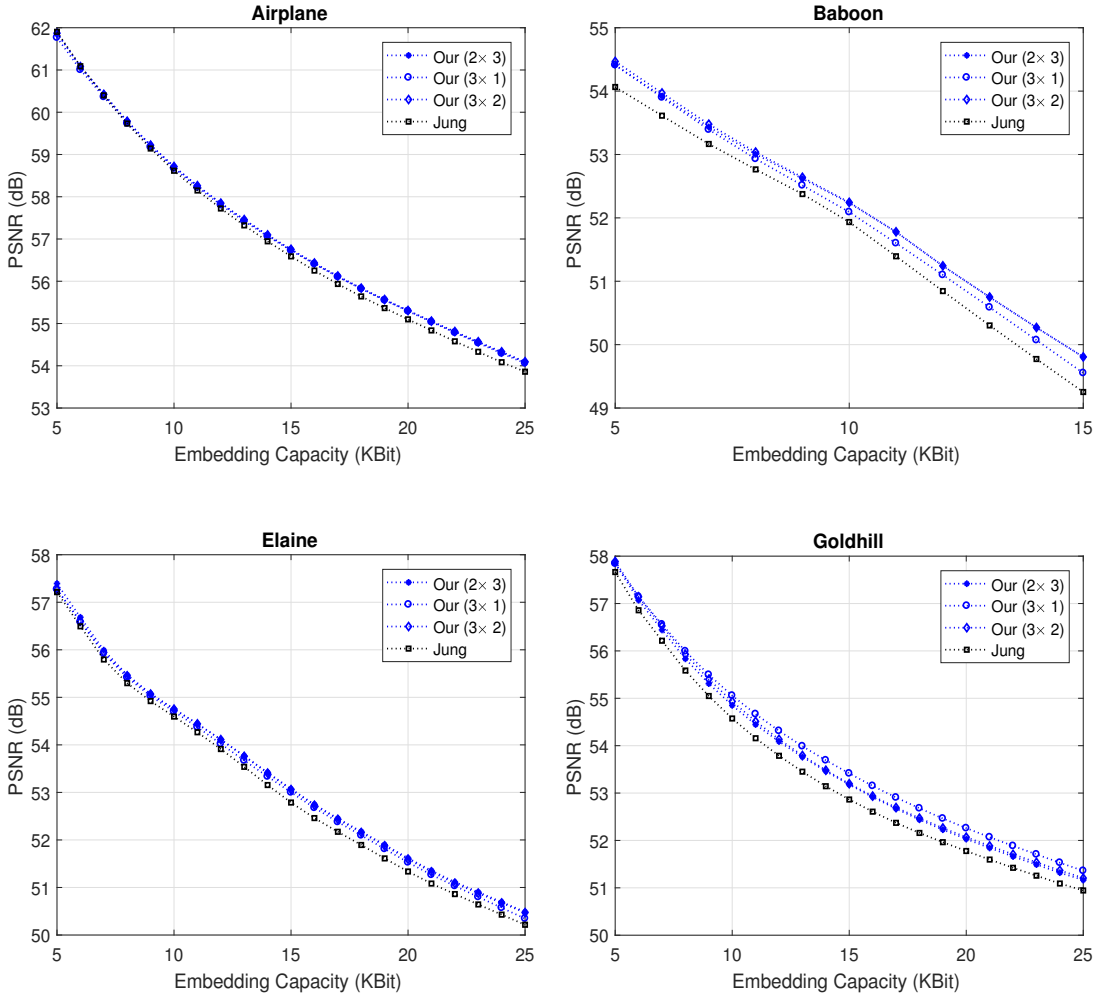


Figure 4.5: Embedding rate-distortion performance comparison

It is also noted that the trend of improvements in rate-distortion performance of the proposed PVO embedding discussed above for a few test images also holds for the other test images used for the performance evaluation.

4.4 Performance of the dPVO-based Embedding

The performance of the proposed RDH scheme with dPVO is now presented for its analysis and validation. For this performance evaluation, both the image databases, USC-SIPI and Kodak are used. As mentioned in the previous section, both the em-

bedding-capacity and embedding-rate are determined in terms of total embedded bits and bit per pixels (bpp), respectively. The visual quality of the embedded images is assessed in terms of PSNR and SSIM.

Overall embedding rate-distortion performance of the proposed scheme is evaluated and presented in Table 4.3 for the USC-SIPI images. Average PSNR (dB) value of 53.57 with embedding rate (bpp) of 0.14 is found for USC-SIPI test-images. Additionally, an image with smoother region like the Airplane, Peppers and Lena offered higher embedding rate than the other images having much non-smooth regions. This is due to the PVO based embedding property that only embeds in the pixels which is closer by only a single bit to the center pixel in a kernel.

Table 4.3: Performance of the proposed scheme for USC-SIPI images

Image	**Embedding capacity**		**PSNR (dB)**	**SSIM**
	Total (Kbits)	**Rate (bpp)**		
Airplane	55.67	0.21	53.99	0.9994
Baboon	17.83	0.07	53.09	0.9998
Barbara	33.88	0.13	53.30	0.9996
Boat	33.62	0.13	53.20	0.9996
Elaine	30.27	0.12	53.26	0.9996
Lake	34.68	0.13	53.30	0.9996
Lena	40.55	0.15	53.48	0.9995
Peppers	38.77	0.15	53.34	0.9995
Tiffany	30.83	0.12	53.40	0.9995
Average	**35.12**	**0.14**	**53.37**	**0.9996**

Similarly, the overall embedding rate-distortion performance of the proposed scheme is also evaluated and presented in Table 4.4 for the Kodak image-sets. For Kodak test-images, an average PSNR (dB) value of 54.40 with embedding rate (bpp) of 0.21 is obtained. Due to relatively larger image size, Kodak images show higher

Table 4.4: Performance of the proposed scheme for Kodak images

Image	**Embedding capacity**		**PSNR**	**SSIM**
	Total (bits)	**Rate (bpp)**	**(dB)**	
kodim01	60453	0.15	53.80	0.9996
kodim02	99987	0.25	54.58	0.9992
kodim03	96353	0.25	55.51	0.9995
kodim04	87435	0.22	54.15	0.9993
kodim05	59261	0.15	53.76	0.9997
kodim06	81545	0.21	54.14	0.9995
kodim07	90632	0.23	55.72	0.9996
kodim08	56815	0.14	53.57	0.9997
kodim09	109870	0.28	54.97	0.9994
kodim10	102318	0.26	54.78	0.9994
kodim11	84092	0.21	54.29	0.9995
kodim12	104905	0.27	55.10	0.9994
kodim13	44104	0.11	53.33	0.9996
kodim14	73000	0.19	53.95	0.9995
kodim15	88828	0.23	54.45	0.9994
kodim16	91394	0.23	55.06	0.9995
kodim17	90313	0.23	54.41	0.9995
kodim18	64441	0.16	53.62	0.9995
kodim19	87826	0.22	54.82	0.9995
kodim20	105883	0.27	54.25	0.9993
kodim21	95517	0.24	54.22	0.9995
kodim22	80679	0.21	53.77	0.9993
kodim23	100599	0.26	55.25	0.9994
kodim24	68978	0.18	54.10	0.9996
Average	**84384.50**	**0.21**	**54.40**	**0.9995**

embedding rate with a better image quality. This means that with a higher number of pixels in an input image, the number of counterbalanced pixels with the proposed dPVO and backward embedding in the second phase of embedding becomes higher, which can improve embedded image quality and embedding rate.

A case by case analysis has also been carried out for different size of payloads of 10 Kbit and 20 Kbit and for two test image databases: USC-SIPI and Kodak. Firstly, the embedded image quality for embedding of 10 Kbit in the test-images is evaluated and compared with the relevant RDH schemes [29–32, 36] with PVO and PEE in Table 4.5. Generally, in contrast to those schemes, the PSNR values of the proposed scheme remain higher for most of the images. In other words, while for some images, the PSNR values are similar or slightly lower, for the other images, it remains significantly higher. Considering the average performance of all the USC-SIPI test images, a better value of PSNR for the proposed scheme is observed than that of the Jung's scheme. Particularly, an average PSNR value of the proposed scheme remains about 7.5% higher than the Jung's scheme as shown in Table 4.5.

Table 4.5: PSNRs (dB) for embedding 10,000 bits in the USC-SIPI images.

Schemes	**Li *et al.* [29]**	**Peng *et al.* [30]**	**Ou *et al.* [31]**	**Qu & Kim [32]**	**Jung** [36]	**Proposed**
Lena	60.3	60.47	60.46	60.31	56.99	60.5
Baboon	53.52	53.55	54.16	54.21	51.62	55.19
Barbara	59.81	60.54	60.15	59.77	55.69	58.96
Airplane	62	62.96	63.14	63.68	58.63	61.65
Peppers	58.87	58.98	59.16	58.78	55.86	59.57
Boat	58.11	58.27	58.06	58.42	54.61	58.9
Elaine	56.81	57.36	57.36	58.72	54.65	58.56
Lake	58.21	58.87	59.23	59.76	55.75	59.11
Average	**58.45**	**58.88**	**58.97**	**59.21**	**55.48**	**59.6**

In another case of embedding of 20 Kbit of data in the USC-SIPI image-sets is evaluated and compared with the relevant RDH schemes [29–32, 36] with PVO and PEE in Table 4.6. Generally, in contrast to those schemes, in both cases of embedding 20 Kbit data, PSNR values of the proposed scheme remain higher. In other words, while for some images, the PSNR values are similar or slightly lower, for the other images, it remains significantly higher. This results in a better average value of PSNR for the proposed scheme than that of the Jung's scheme, which the proposed scheme is directly built on. Particularly, an average PSNR value of the proposed scheme remains about 7% higher than the Jung's scheme in case of 20 Kbit of embedding for USC-SIPI image set.

Table 4.6: PSNRs (dB) for embedding 20,000 bits in the USC-SIPI images.

Schemes	**Li *et al.* [29]**	**Peng *et al.* [30]**	**Ou *et al.* [31]**	**Qu & Kim [32]**	**Jung** [36]	**Proposed**
Lena	56.21	56.54	56.6	56.67	53.40	56.92
Baboon	–	–	–	–	–	49.95
Barbara	54.69	56.2	55.89	55.63	51.99	55.75
Airplane	58.13	59.07	59.26	59.91	56.16	58.97
Peppers	54.72	54.77	54.93	54.98	52.67	56.23
Boat	53.34	53.84	53.72	54.21	51.69	55.11
Elaine	52.41	52.61	52.71	53.71	51.33	54.95
Lake	53.44	53.6	54.28	54.69	52.08	56.17
Average	**54.92**	**55.23**	**55.34**	**55.69**	**52.76**	**56.30***

*Average value is calculated without Baboon image.

Embedding performance is also evaluated for the Kodak image sets. Firstly, the embedded image quality for embedding of 10 Kbit is evaluated and compared with the relevant RDH schemes considered above. For embedding 10 Kbit, the proposed scheme generally shows an improved image quality in terms of PSNR value over the

other scheme as illustrated in Table 4.7. While for some images, the PSNR values are similar or slightly lower, for the other images, it remains significantly higher. An average PSNR value of the proposed scheme remains about 5% higher than the Jung's scheme for all the test images of Kodak set.

Table 4.7: PSNRs (dB) for embedding 10,000 bits in Kodak images.

Image	**Li *et al*. [29]**	**Peng *et al*. [30]**	**Ou *et al*. [31]**	**Qu & Kim [32]**	**Jung [36]**	**Proposed**
kodim01	57.95	61.58	61.45	64.23	58.10	61.38
kodim02	62.83	64.07	61.97	64.49	60.13	63.12
kodim03	63.84	65.39	63.57	65.12	64.70	66.86
kodim04	62.08	63.62	63.46	64.17	57.20	60.45
kodim05	59.64	61.61	54.62	62.51	59.87	63.05
kodim06	61.36	65.02	60.62	66.05	64.93	67.05
kodim07	63.67	65.12	63.07	64.83	64.29	66.48
kodim08	52.73	56.23	63.50	57.83	58.37	61.58
kodim09	62.79	63.72	62.68	63.34	63.20	65.08
kodim10	61.90	63.11	60.32	62.68	60.86	63.45
kodim11	62.46	65.19	62.66	65.43	64.82	66.36
kodim12	63.00	64.51	62.96	64.75	64.06	66.50
kodim13	52.16	54.54	53.96	58.20	57.72	61.44
kodim14	59.51	61.35	62.86	62.62	59.22	62.30
kodim15	62.35	64.46	62.06	62.86	60.92	63.89
kodim16	63.11	64.98	63.42	65.06	65.71	67.71
kodim17	62.06	63.76	60.36	64.52	62.43	64.84
kodim18	59.63	60.74	63.96	61.02	57.48	60.65
kodim19	62.53	63.36	62.57	63.19	62.29	64.83
kodim20	61.14	65.54	60.74	57.76	63.01	65.48
kodim21	62.54	63.72	63.22	63.58	65.65	67.19
kodim22	61.36	62.59	63.74	63.04	63.10	65.53
kodim23	63.21	64.58	56.74	64.35	64.72	67.17
kodim24	58.80	62.20	62.88	62.24	58.09	61.24
Average	**60.94**	**62.96**	**61.56**	**63.08**	**61.70**	**64.32**

Table 4.8: PSNRs (dB) for embedding 20,000 bits in Kodak images.

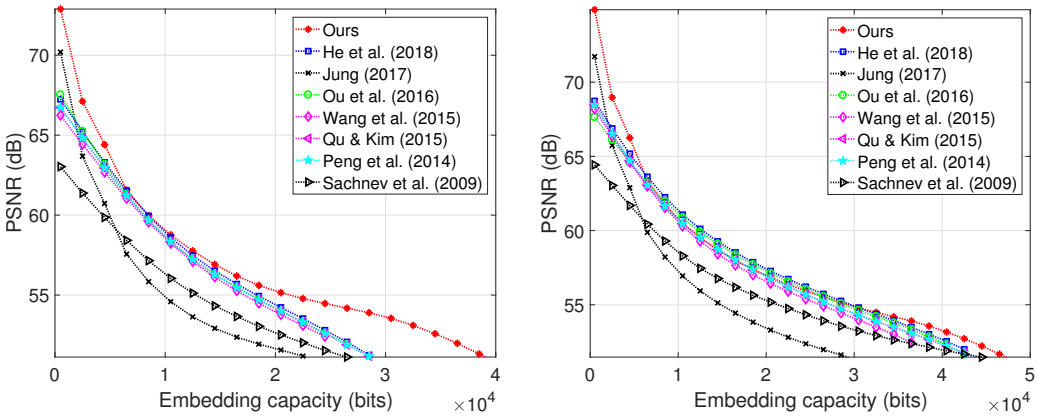
Image	Li *et al*. [29]	Peng *et al*. [30]	Ou *et al*. [31]	Qu & Kim [32]	Jung [36]	Proposed
kodim01	53.36	56.37	55.52	60.47	54.32	58.04
kodim02	58.76	60.52	58.57	61.37	57.17	60.26
kodim03	60.16	62.00	59.04	62.42	60.47	62.98
kodim04	58.02	59.99	59.89	60.85	54.62	57.63
kodim05	55.10	57.53	–	58.94	55.76	59.23
kodim06	56.47	61.45	56.27	63.45	61.56	63.88
kodim07	60.05	61.81	59.90	62.03	60.33	62.87
kodim08	–	51.98	59.91	55.10	54.67	58.14
kodim09	59.09	60.23	58.89	60.50	59.95	61.99
kodim10	58.24	59.71	56.24	59.98	57.72	60.38
kodim11	57.38	61.23	58.93	62.23	59.16	61.86
kodim12	58.98	61.03	59.47	61.80	60.18	62.93
kodim13	–	–	–	53.76	53.74	57.40
kodim14	55.28	57.27	59.09	58.68	56.51	59.67
kodim15	58.30	61.32	58.22	61.99	58.12	61.06
kodim16	58.55	61.52	60.10	62.22	62.48	64.60
kodim17	58.07	59.99	56.87	61.17	58.08	60.96
kodim18	54.82	56.47	60.56	57.48	54.12	57.67
kodim19	58.71	59.82	58.90	60.14	59.33	61.86
kodim20	56.96	62.40	56.42	58.15	59.96	62.41
kodim21	58.66	60.17	58.50	60.87	62.42	64.10
kodim22	57.15	58.66	60.23	59.56	58.76	61.73
kodim23	59.68	61.12	53.06	61.61	61.18	63.71
kodim24	54.81	58.59	59.23	60.49	55.86	58.89
Average	**57.57**	**59.62**	**58.36**	**60.22**	**58.19**	**61.01**

In a case of higher embedding rate, *i.e*., embedding of 20 Kbit of data, the embedded image quality is also evaluated and compared with the relevant RDH schemes. This is illustrated in Table 4.8. Similar improvement of embedded image quality is demonstrated over the other schemes as found in case of the USC-SIPI image sets.

Particularly, while for some images, the PSNR values are similar or slightly lower, for the other images, it remains significantly higher. An average PSNR value of the proposed scheme remains about 5% higher than the Jung's scheme , respectively.

The above cases of embedding 10 Kbit and 20 Kbit of data in both test-image sets and respective embedded quality comparison over the relevant schemes suggest that the proposed scheme can have better image quality for relatively higher embedding rate.

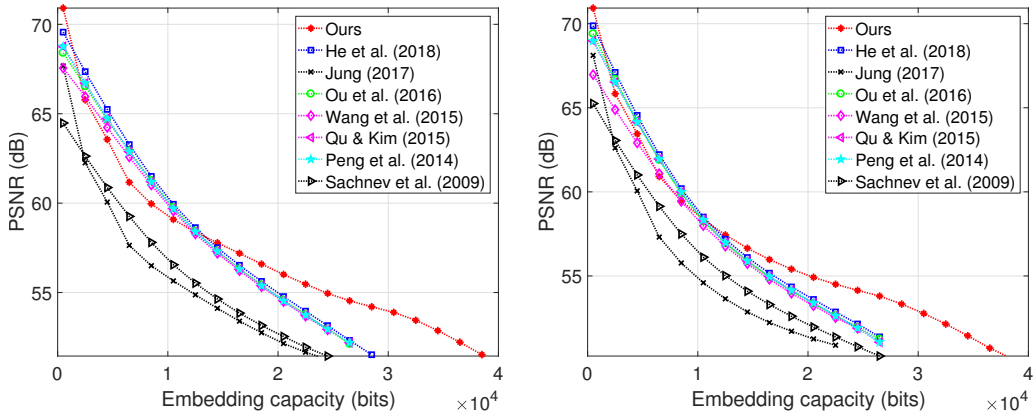
Moreover, an embedding rate-distortion curve is usually more helpful to visualize the improvement of a new scheme over the existing schemes. To determine the trend of the overall performance of the proposed scheme, embedding rate-distortion curve is, therefore, compared with a few recent and popular RDH schemes [19, 29, 30, 32–34, 36, 38]. For example, Fig. 4.6 illustrates the trends of PSNR over the increasing embedding capacity for the Boat and Lena images. Over the existing schemes, our proposed scheme has offered a better image quality, particularly at the higher embedding rate.



(a) Boat (b) Lena

Figure 4.6: Overall embedding rate-distortion performance comparison with the Boat and Lena images over the popular and recent RDH schemes of He *et al*. (2018) [38], Jung (2017) [36], Ou *et al*. (2016) [34], Wang *et al*. (2015) [33], Qu & Kim (2015) [32], Peng *et al*. (2014) [30] and Sachnev *et al*. (2009) [19] .

With another example of the Lake and Elaine test images in Fig. 4.7, the embedding



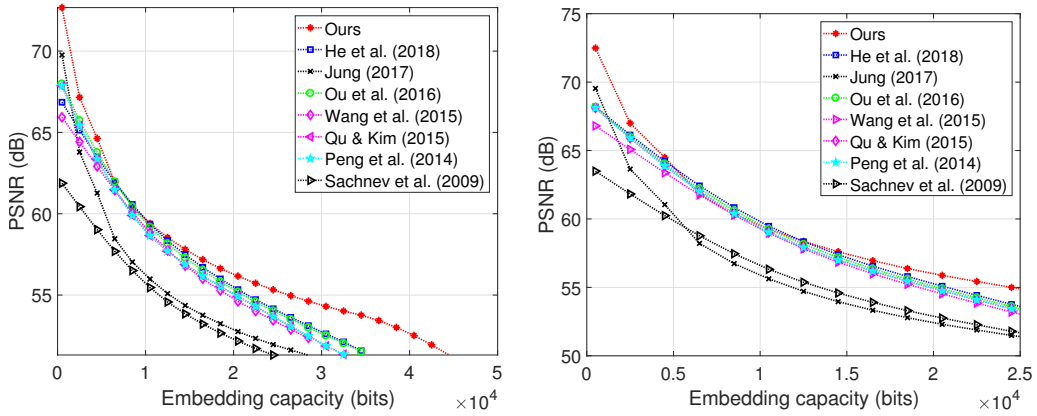
(a) Lake (b) Elaine

Figure 4.7: Overall embedding rate-distortion performance comparison with the Lake and Elaine images over the popular and recent RDH schemes of He *et al*. (2018) [38], Jung (2017) [36], Ou *et al*. (2016) [34], Wang *et al*. (2015) [33], Qu & Kim (2015) [32], Peng *et al*. (2014) [30] and Sachnev *et al*. (2009) [19] .

rate-distortion curve of the proposed scheme is found to improve over the existing schemes. Particularly, a relatively higher trend of PSNR over the increasing embedding capacity for the images is evident for our proposed scheme over the other schemes.

The proposed scheme has demonstrated a trend to improve embedded image quality over the higher embedding rate. This is because a higher rate of embedding requires a higher number of pixels to be embedded. Once the number of pixels becomes higher, the sizes of X_{min} and X_{max} in the second phase of embedding tend to be higher resulting in more counterbalanced pixels and beter-embedded image quality. The average performance in Fig. 4.8 also holds the trend of improvement. This trend of improvements in the rate-distortion performance of the proposed RDH scheme discussed above and illustrated for a few test images also holds for the other test images.

In addition, no significant visual difference can be noticed in the embedded image and its decoded image. The decoded image is the same as input image by definition of the RDH scheme. This means that the distortion in the embedded image may be acceptable as also verified in the quantitative evaluation mentioned above.



(a) Peppers (b) Average

Figure 4.8: Overall embedding rate-distortion performance comparison with the Peppers image and average values over the popular and recent RDH schemes of He *et al*. (2018) [38], Jung (2017) [36], Ou *et al*. (2016) [34], Wang *et al*. (2015) [33], Qu & Kim (2015) [32], Peng *et al*. (2014) [30] and Sachnev *et al*. (2009) [19] .

Example of the embedded images and their decoded versions for the USC-SIPI image sets are shown in Fig. 4.9 and Fig. 4.10. Similarly, the embedded and decoded images for the Kodak images are shown in Fig. 4.11. The decoded image is found identical to the input image, which is verified for all the test images and can be roughly observed with the given examples.

(a) embedded (b) decoded

Figure 4.9: Example of embedded and decoded versions of Boat image of size $512 \times 512 \times 8$. (Original test images are from USC-SIPI database [50].)

(a) embedded (b) decoded

Figure 4.10: Example of embedded and decoded versions of Lake image of size $512 \times 512 \times 8$. (Original test images are from USC-SIPI database [50].)

(a) embedded (b) decoded

(c) embedded (d) decoded

Figure 4.11: Example of embedded and decoded versions of test images (a, b) *kodim19* of size $768 \times 512 \times 8$ and (c, d) *kodim24* of size $512 \times 768 \times 8$. (Original test images are from Kodak database [51].)

On the other hand, the reversibility of the proposed embedding function has been verified by comparing the difference between the input image of the embedding function and the output image of the decoding function. A null difference is found, which establishes that the proposed embedding function is reversible.

Nevertheless, considering the overall rate-distortion performance, the proposed RDH scheme outperforms its baseline schemes: Jung (2017) [36], Peng *et al*. (2014) [30] and Li *et al*. (2013) [29]. The proposed scheme is based on the classic PVO and uses a simple scenario of image partitions of size 1×3, and thus those schemes for baseline comparison are reasonably considered. However, a promising performance of the proposed scheme is also observed, while it is compared to the state-of-the-art PVO based RDH schemes [32–34, 38].

4.5 Chapter Summary

The performance of the proposed RDH scheme is evaluated and validated for USC-SIPI and Kodak test images. The embedding capacity and rate are evaluated in terms of total bits embedded and bpp, respectively. The image quality is expressed in terms of PSNR and SSIM. The performance improvement due to the new kernels are evaluated as well as the overall embedding rate-distortion performance of the proposed RDH scheme is evaluated and compared with a number of popular and recent RDH schemes. Improvements in terms of visual quality and embedding rate are explained and analytical notes are given. Based on the observations presented in this chapter, concluding remarks and future works of the research are presented in the following chapter.

CHAPTER 5

CONCLUSIONS AND FUTURE WORKS

5.1 Conclusions

The research presented in this thesis aimed to develop a PVO-based RDH scheme for digital image applications. Development of a new data hiding scheme is steered by its better embedding rate-distortion performance. Since a higher data embedding rate naturally causes higher embedding distortion, achieving better rate-distortion performance is a challenging task. Existing PVO-based RDH schemes have demonstrated impressive performance. However, none of them has considered a more correlated pixel blocks and counterbalancing the distortion in an additional embedding, which can improve both the embedding rate and embedded image quality. The research presented in this thesis therefore contributes to the development of a new RDH scheme. Research contributions and its significance are summarized in the sections below.

5.1.1 Research outcomes

This thesis contributes to the development of a new PVO-based RDH scheme with a new PVO-kernel and backward embedding. Firstly, a new triangular kernel is proposed that captures the pixels correlated in the horizontal, vertical and diagonal directions simultaneously. The proposed kernel is employed in a prominent PVO-based RDH scheme and is verified for a better (or occasionally similar) rate-distortion performance than the existing schemes that rely on only the column- or row-kernel.

Additionally, an improved PVO-based RDH scheme is reported in this thesis with a new backward embedding technique to counterbalance the distortion caused in a forward embedding phase. Particularly, the proposed scheme embeds in two phases: (*i*) forward embedding, and (*ii*) backward embedding. In the forward embedding, PEE

is applied with PVO to every non-overlapping image block of size 1×3. In the backward embedding, *minimum-set* and *maximum-set* of pixels are determined from the pixels predicted in the first phase. The pixels predicted in the first phase is investigated to partially restore them to their original values resulting in both a higher embedding rate and a better-embedded image quality.

The computational modeling, evaluation, analysis and validation of the new PVO-based RDH scheme are presented in the thesis. Given experimental results has demonstrated a promising performance of the proposed scheme and its improvement over the popular and state-of-the-art PVO-based RDH schemes. A significantly better rate-distortion performance is obtained at the higher embedding rate, which means the proposed scheme is more promising to the applications that usually require high embedding capacity like electronic patient record hiding in medical images.

The contributions and findings that are discussed in this thesis have been presented in several reputable conferences, published in the IEEE conference proceedings and submitted for possible publications in two reputable journals (see the list of publications in Page 53). The PVO-based embedding approach has also been appreciated with a best paper award in EICT 2017.

5.1.2 Research significance

This thesis dissertation advances knowledge in the area of reversible data hiding and its application to digital images. Particularly, the proposed new kernel for PVO can effectively use the pixels' correlation resulting in lower possible embedding distortion in the embedded images. The proposed backward-embedding introduces a new approach of counterbalancing the prediction errors. More data, therefore, can be embedded with an improved image quality than the existing RDH schemes.

Moreover, the new RDH scheme developed using the proposed new kernel and backward-embedding would create a new paradigm of RDH and open several oppor-

tunities for data hiding research. The presented experimental results demonstrated the effectiveness of the proposed scheme for better visual quality with higher embedding capacity compared to the prominent RDH schemes.

5.2 Future Works

Some possible avenues for future research have been identified. In addition to the study of its specific application scenarios, future investigation on the proposed dPVO based principle of backward embedding may be worthwhile, for example, in the following areas: (*i*) its information theoretic analysis, (*ii*) optimization of its computational requirements for multi-level embedding, and (*iii*) developing its generalized framework for multi-level embedding and dynamic image-block partitioning.

A further development of computational efficiency with more concrete foundation of the proposed PVO-based embedding may be studied with information theoretic analysis. Both the new kernels and dPVO techniques presented in this thesis may be investigated further for their combined employment to further develop an RDH scheme in future. Additionally, multilevel embedding with the proposed scheme may require higher resources. The forward and backward embedding may be streamlined using their equivalent simultaneous operations to optimize their efficiency in terms of both the computational resources and their rate-distortion performance.

For the attainment of reasonably higher embedding rate, an interpolation based embedding may be combined with the proposed embedding approach. Study of a proper sequence of application of the embedding techniques to determine the embedding capacity and respective distortion may also open up a new scope for making a positive research contribution in this area.

Moreover, extension of the application of the proposed scheme to other specialized images like medical and military images can be an interesting area of investigation.

Determining the requirements of those applications and study of the suitability of the proposed scheme may create another avenue for further development of the proposed scheme. Development of a generalized framework of the proposed PVO-based scheme may therefore also be investigated.

LIST OF PUBLICATIONS

Journal Papers:

(i) **Hasib, S. A.** and Nyeem, H., "Reversible data hiding with dual pixel value ordering and prediction error expansion,"*An International Journal of Engineering Science and Technology (JESTECH)*, Elsevier, 2019. (under review)

(ii) **Hasib, S. A.** and Nyeem, H., "Pixel grouping of digital images for reversible data hiding," *ACTA Cybernetica*, 2019. (under review)

Conference Papers:

(iii) **Hasib, S. A.** and Nyeem, H., "Can the Expansion of Prediction Errors be Counterbalanced in Reversible Data Hiding?." *Proceedings of International Joint Conference on Computational Intelligence*, pp. 97-109. Springer, Singapore, 2020.

(iv) **Hasib, S. A.** and Nyeem, H., " Developing a pixel value ordering based reversible data hiding scheme," *Proceedings of the 3rd International Conference on Electrical Information and Communication Technology (EICT), 2017*, Khulna, Bangladesh, IEEE, 2017, pp. 1-6.

(v) **Hasib, S. A.**, and Nyeem, H., "Rate-Distortion Analysis of an Improved Pixel Value Ordering based Reversible Embedding," *Proceedings of the 2nd International Conference on Electrical & Electronic Engineering (ICEEE)*, Rajshahi, Bangladesh, IEEE, 2017, pp. 1-4.

BIBLIOGRAPHY

[1] A. Khan, A. Siddiqa, S. Munib, and S. A. Malik, "A recent survey of reversible watermarking techniques," *Information Sciences*, vol. 279, pp. 251–272, 2014.

[2] Y.-Q. Shi, X. Li, X. Zhang, H.-T. Wu, and B. Ma, "Reversible data hiding: advances in the past two decades," *IEEE Access*, vol. 4, pp. 3210–3237, 2016.

[3] I. J. Cox, M. L. Miller, J. A. Bloom, J. Fridrich, and T. Kalker, *Models of Watermarking*. Burlington: Morgan Kaufmann, 2008, pp. 61–103.

[4] H. Nyeem, W. Boles, and C. Boyd, "Digital image watermarking: its formal model, fundamental properties and possible attacks," *EURASIP Journal on Advances in Signal Processing*, vol. 2014, no. 1, pp. 1–22, 2014.

[5] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, pp. 890–896, 2003.

[6] A. M. Alattar, "Reversible watermark using the difference expansion of a generalized integer transform," *IEEE Trans. Image Processing*, vol. 13, pp. 1147–1156, 2004.

[7] H. J. Kim, V. Sachnev, Y. Q. Shi, J. Nam, and H.-G. Choo, "A novel difference expansion transform for reversible data embedding," *IEEE Transactions on Information Forensics and Security*, vol. 3, pp. 456–465, 2008.

[8] Y. Hu, H.-K. Lee, and J. Li, "DE-based reversible data hiding with improved overflow location map," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, pp. 250–260, 2009.

[9] L. Kamstra and H. J. Heijmans, "Reversible data embedding into images using wavelet techniques and sorting," *IEEE Trans. Image Processing*, vol. 14, pp. 2082–2090, 2005.

[10] D. M. Thodi and J. Rodríguez, "Expansion embedding techniques for reversible watermarking," *IEEE Transactions on Image Processing*, vol. 16, pp. 721–730, 2007.

[11] Y. Hu, H.-K. Lee, K. Chen, and J. Li, "Difference expansion based reversible data hiding using two embedding directions," *IEEE Transactions on Multimedia*, vol. 10, no. 8, pp. 1500–1512, 2008.

[12] C.-C. Lee, H.-C. Wu, C.-S. Tsai, and Y.-P. Chu, "Adaptive lossless steganographic scheme with centralized difference expansion," *Pattern Recognition*, vol. 41, no. 6, pp. 2097–2106, 2008.

[13] Z. Ni, Y.-Q. Shi, N. Ansari, and W. Su, "Reversible data hiding," *IEEE Trans. CSVT*, vol. 16, pp. 354–362, 2006.

[14] C.-C. Lin, W.-L. Tai, and C.-C. Chang, "Multilevel reversible data hiding based on histogram modification of difference images," *Pattern Recognition*, vol. 41, pp. 3582–3591, 2008.

[15] W.-L. Tai, C.-M. Yeh, and C.-C. Chang, "Reversible data hiding based on histogram modification of pixel differences," *IEEE Transactions on Circuits and Systems for Video technology*, vol. 19, no. 6, pp. 906–910, 2009.

[16] K.-S. Kim, M.-J. Lee, H.-Y. Lee, and H.-K. Lee, "Reversible data hiding exploiting spatial correlation between sub-sampled images," *Pattern Recognition*, vol. 42, pp. 3083–3096, 2009.

[17] X. Li, W. Zhang, X. Gui, and B. Yang, "Efficient reversible data hiding based on multiple histograms modification," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 9, pp. 2016–2027, 2015.

[18] J. Wang, J. Ni, X. Zhang, and Y.-Q. Shi, "Rate and distortion optimization for reversible data hiding using multiple histogram shifting," *IEEE Transactions on Cybernetics*, vol. 47, no. 2, pp. 315–326, 2017.

[19] V. Sachnev, H. J. Kim, J. Nam, S. Suresh, and Y. Q. Shi, "Reversible watermarking algorithm using sorting and prediction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 7, pp. 989–999, 2009.

[20] D. Coltuc, "Improved embedding for prediction-based reversible watermarking," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 873–882, 2011.

[21] I.-C. Dragoi and D. Coltuc, "Local-prediction-based difference expansion reversible watermarking," *IEEE Transactions on image processing*, vol. 23, no. 4, pp. 1779–1790, 2014.

[22] G. Coatrieux, W. Pan, N. Cuppens-Boulahia, F. Cuppens, and C. Roux, "Reversible watermarking based on invariant image classification and dynamic histogram shifting," *IEEE Transactions on Iinformation Forensics and Security*, vol. 8, no. 1, pp. 111–120, 2013.

[23] X. Li, B. Yang, and T. Zeng, "Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection," *IEEE Transactions on Image Processing*, vol. 20, pp. 3524–3533, 2011.

[24] W. Hong, "Adaptive reversible data hiding method based on error energy control and histogram shifting," *Optics Communications*, vol. 285, no. 2, pp. 101–108, 2012.

[25] X. Wang, X. Li, B. Yang, and Z. Guo, "Efficient generalized integer transform for reversible watermarking," *IEEE Signal Processing Letters*, vol. 17, no. 6, pp. 567–570, 2010.

[26] H. Y. Leung, L. M. Cheng, F. Liu, and Q. Fu, "Adaptive reversible data hiding based on block median preservation and modification of prediction errors," *Journal of Systems and Software*, vol. 86, no. 8, pp. 2204–2219, 2013.

[27] B. Ou, X. Li, Y. Zhao, R. Ni, and Y.-Q. Shi, "Pairwise prediction-error expansion for efficient reversible data hiding," *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 5010–5021, 2013.

[28] X. Li, W. Zhang, X. Gui, and B. Yang, "A novel reversible data hiding scheme based on two-dimensional difference-histogram modification," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 7, pp. 1091–1100, 2013.

[29] X. Li, J. Li, B. Li, and B. Yang, "High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion," *Signal Processing*, vol. 93, no. 1, pp. 198–205, 2013.

[30] F. Peng, X. Li, and B. Yang, "Improved PVO-based reversible data hiding," *Digital Signal Processing*, vol. 25, pp. 255–265, 2014.

[31] B. Ou, X. Li, Y. Zhao, and R. Ni, "Reversible data hiding using invariant pixel-value-ordering and prediction-error expansion," *Signal Processing: Image Communication*, vol. 29, no. 7, pp. 760–772, 2014.

[32] X. Qu and H. J. Kim, "Pixel-based pixel value ordering predictor for high-fidelity reversible data hiding," *Signal Processing*, vol. 111, pp. 249–260, 2015.

[33] X. Wang, J. Ding, and Q. Pei, "A novel reversible image data hiding scheme based on pixel value ordering and dynamic pixel block partition," *Information Sciences*, vol. 310, pp. 16–35, 2015.

[34] B. Ou, X. Li, and J. Wang, "High-fidelity reversible data hiding based on pixel-value-ordering and pairwise prediction-error expansion," *Journal of Visual Communication and Image Representation*, vol. 39, pp. 12–23, 2016.

[35] I.-C. Dragoi and D. Coltuc, "Adaptive pairing reversible watermarking," *IEEE Transactions on Image Processing*, vol. 25, no. 5, pp. 2420–2422, 2016.

[36] K.-H. Jung, "A high-capacity reversible data hiding scheme based on sorting and prediction in digital images," *Multimedia Tools and Applications*, vol. 76, no. 11, pp. 13 127–13 137, 2017.

[37] W. He, K. Zhou, J. Cai, L. Wang, and G. Xiong, "Reversible data hiding using multi-pass pixel value ordering and prediction-error expansion," *Journal of Visual Communication and Image Representation*, vol. 49, pp. 351–360, 2017.

[38] W. He, G. Xiong, S. Weng, Z. Cai, and Y. Wang, "Reversible data hiding using multi-pass pixel-value-ordering and pairwise prediction-error expansion," *Information Sciences*, 2018.

[39] C.-C. Chang, W.-L. Tai, and C.-C. Lin, "A reversible data hiding scheme based on side match vector quantization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 10, pp. 1301–1308, 2006.

[40] C. Qin, C.-C. Chang, and Y.-P. Chiu, "A novel joint data-hiding and compression scheme based on smvq and image inpainting," *IEEE transactions on image processing*, vol. 23, no. 3, pp. 969–978, 2014.

[41] C.-N. Yang, S.-C. Hsu, and C. Kim, "Improving stego image quality in image interpolation based data hiding," *Computer Standards & Interfaces*, vol. 50, pp. 209–215, 2017.

[42] M. A. Wahed and H. Nyeem, "Reversible data hiding with interpolation and adaptive embedding," *Multimedia Tools and Applications*, pp. 1–25, 2018.

[43] W. Zhang, K. Ma, and N. Yu, "Reversibility improved data hiding in encrypted images," *Signal Processing*, vol. 94, pp. 118–127, 2014.

[44] X. Cao, L. Du, X. Wei, D. Meng, and X. Guo, "High capacity reversible data hiding in encrypted images by patch-level sparse representation," *IEEE Transactions on Cybernetics*, vol. 46, no. 5, pp. 1132–1143, 2016.

[45] B. Ma and Y. Q. Shi, "A reversible data hiding scheme based on code division multiplexing," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 9, pp. 1914–1927, 2016.

[46] N. Muhammad, N. Bibi, Z. Mahmood, T. Akram, and S. R. Naqvi, "Reversible integer wavelet transform for blind image hiding method," *PloS one*, vol. 12, no. 5, p. e0176979, 2017.

[47] H.-W. Tseng and C.-P. Hsieh, "Prediction-based reversible data hiding," *Information Sciences*, vol. 179, no. 14, pp. 2460–2469, 2009.

[48] B. Ou, X. Li, and J. Wang, "Improved pvo-based reversible data hiding: A new implementation based on multiple histograms modification," *Journal of Visual Communication and Image Representation*, vol. 38, pp. 328–339, 2016.

[49] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[50] USC-SIPI, "Image database," http://sipi.usc.edu/database/, [Online; last accessed 23-Nov-2013].

[51] "Kodak lossless true color image suit," http://www.r0k.us/graphics/kodak/, [Online; last accessed 23-Jan-2017].

APPENDIX A
MATLAB CODES

A.1 The Proposed dPVO-based Embedding

```
clc
clear all
close all

%% Define the path of the test image for batch run
path = strcat(cd, '\TestImage\');
pathresult = strcat(cd, '\sResult\SIPIfull\');
contents = dir(path);

Method = [];

for F=1:numel(contents)-2
    filename = contents(F+2).name;
    I= imread(filename);

    load eqdata                 % the data bits to be embedded
    Io= double(I(:,:)); % original test input image
    Io(Io==0)=1;
    Io(Io==255)=254;

    [M, N] = size(Io);
    pxs =0;
    pdx =1;

    tic
    Iw = Io;
    x=1;
    for i=1:1:M
        for j=1:3:N-2
            if pdx +1 ≤ numel(eqdata)
                [ Iw(i,j:j+2), Iems(x:x+2, :), unusedbit, pshifted] = ...
                    jungembnew(Io(i,j:j+2), eqdata, pdx);
                pxs = pxs + pshifted;
                pdx = pdx+2-unusedbit;
                x=x+3;
            end
        end
    end

    Cjung=pdx-1
    x=1;
    y=1;
    minset=[];
```

```
maxset=[];
LM =[];
for i=1:3:numel(Iems(:,1))
    if Iems(i+1,1)-Iems(i,1)>1
        minset(x,1) = Iems(i,1);
        minset(x,2) = Iems(i,2);
        minset(x,3) = i;
        x=x+1;
    else LM= [LM, i];
    end
    if Iems(i+2,1)-Iems(i+1,1)>1
        maxset(y,1) = Iems(i+2,1);
        maxset(y,2) = Iems(i+2,2);
        maxset(y,3) = i+2;
        y=y+1;
    else LM= [LM, i+2];
    end
end

Iemse = Iems;
%% MIN embedding
if numel(minset(:,1))>1
    dminset = minset;
    for i = 1:2:numel(minset(:,1))-1
        temp = minset(i:i+1, 1)';
        [dminset(i:i+1, 1), unusedbit, pshifted] = dminemb(temp, eqdata, pdx);
        pxs = pxs + pshifted;
        pdx = pdx+1-unusedbit;
    end

    x=1;
    for i=1:3:numel(Iems(:,1))
        if x < length(dminset)+1
            if dminset(x,3)==i
                Iemse(i,1) = dminset(x, 1);
                x=x+1;
            end
        end
    end
end


%% MAX embedding
if numel(maxset(:,1))>1
    dmaxset = maxset;
    for i = 1:2:numel(maxset(:,1))-1
        temp = maxset(i:i+1, 1)';
        [dmaxset(i:i+1, 1), unusedbit, pshifted] = dmaxemb(temp, eqdata, pdx);
        pxs = pxs + pshifted;
        pdx = pdx+1-unusedbit;
    end
```

```
94
95          x=1;
96          for i=3:3:numel(Iems(:,1))
97             if x < length(dmaxset)+1
98              if dmaxset(x,3)==i
99                  Iemse(i,1) = dmaxset(x, 1);
100                 x=x+1;
101             end
102            end
103         end
104     end
105
106
107     %% Reshaping embedded image
108      x=1;
109      Inew= Iw;
110      for i=1:1:M
111          for j=1:3:N-2
112              temp = Iemse(x:x+2,:);
113              temp = sortrows(temp, 2);
114              Inew(i,j:j+2) = temp(:,1)';
115              x=x+3;
116          end
117      end
118    Ctot = pdx-1
119    Pjung= PSNR(Io, Iw)
120    Pimp= PSNR(Io, Inew)
121
122   embtime(F) = toc;
123   Ctot(F)= pdx-2+unusedbit-1;
124   bpp(F)= Ctot(F)/(M*N);
125   psnrval(F)= PSNR(Iw,Io);
126   ssimval(F)= ssim(Iw,Io);
127
128   [pathx, fname, extx]=fileparts(strcat(path, filename));
129
130   imwrite(uint8(Io),strcat(pathresult, fname,'_orig','.tif'));
131   imwrite(uint8(Iw),strcat(pathresult, fname,'_sPVO_1x3','.tif'));
132   Temp = string(strcat(fname,'_sPVO_1x3'));
133
134   % mat file writing
135   mfname = [pathresult, fname,'sPVO_1x3','.mat' ];
136   save(mfname)
137
138   if F==1
139          Method = Temp;
140   else
141          Method = [Method;Temp];
142   end
143  end
144
```

```
 psnrval = psnrval';
 ssimval = ssimval';
 Ctot  = Ctot';
 bpp = bpp';
 embtime = embtime';

 % xl file writing
 xlfname = [pathresult, 'sPVO_1x3','.csv' ];
 Tprop = table(Method, Ctot, bpp, psnrval,ssimval, embtime);
 writetable(Tprop, xlfname, 'WriteVariableNames', 1);

 fprintf('%s completed \n', fname);
```

A.1.1 *jungembnew* $(\cdot)$

```
function [Iwfull, Ctot] = jungembfull(Io, eqdata)
Io = double(Io);

depth = ceil(log2(double(max(Io(:)+1))));
MAX = 2^depth -1;
Io(Io==0) = Io(Io==0)+1;
Io(Io==MAX) = Io(Io==MAX)-1;
k=mod(size(Io),3);
Iop = Io(1:end-k(1), 1:end-k(2));
pdx =1;
[M, N] = size(Iop);

Idx = double(reshape(1:M*N, [M,N]));
%Block-wise zigzag scanning
Izdx = im2col(Idx,[3 1],'distinct');
Iopz = Iop(Izdx);

pdx =1;
[R, C] = size(Iopz);
Iw = double(zeros(R, C));
for j = 1:C
        pix(1:3) = Iopz(:,j);
        [Iw(:,j), unusedbit] = jungemb(pix, eqdata, pdx);
        pdx = pdx+2-unusedbit;
end

temp=sortrows([Iw(:), Izdx(:)],2);
Iwnew= reshape(temp(:,1), [M, N]);

Iwfull = uint8(Io);
Iwfull(1:end-k(1), 1:end-k(2))=Iwnew(:,:);
Ctot= pdx-2+unusedbit-1; %new line
end
```

A.1.2 *dminemb* $(\cdot)$

```
function [wpixnew, k, pxs] = dminemb(pix, payloadData, pdx)
%pix is an input array containing three pixels
%idx is an input array containing indeces of the input three pixels
%bits is an array of embedding bits.

pxs=0;
idx = [1,2];
pd= [pix;idx]';         % two columns, 1st for pix and 2nd for its idx
spd = sortrows(pd,1);    % sorting pix with idx. For example, 1st row contain ...
    the lowest pix and its location in an image
emax = spd(2,1)-spd(1,1);
k=1;
        if emax == 0
            nemax = emax;
        elseif emax == 1 && (spd(1,1)+emax+payloadData(pdx))<256
            nemax = emax + payloadData(pdx);
            pdx = pdx+1;
            k = k-1;
        elseif emax > 1 && (spd(1,1)+emax+1)<256
            pxs = pxs+1;
            nemax = emax+1;
        else
            nemax = emax;
        end

        wpix = spd;
        wpix(2,1) = spd(1,1)+ nemax;

        %de-sorting using original indeces of input pixels
        temp = sortrows(wpix, 2);
        wpixnew = temp(:,1)';
end
```

A.1.3 *dmaxemb* $(\cdot)$

```
function [wpixnew, k, pxs] = dmaxemb(pix, payloadData, pdx)
%pix is an input array containing three pixels
%idx is an input array containing indeces of the input three pixels
%bits is an array of embedding bits.

pxs=0;
idx = [1,2];
pd= [pix;idx]';         % two columns, 1st for pix and 2nd for its idx
spd = sortrows(pd,1);    % sorting pix with idx. For example, 1st row contain ...
    the lowest pix and its location in an image
```

```
emin = spd(1,1)-spd(2,1);
k=1;
        if emin == 0
            nemin = emin;
        elseif emin == -1 && (spd(2,1)+emin-payloadData(pdx))>-1
            nemin = emin - payloadData(pdx);
            k = k-1;
        elseif (emin < -1) && (spd(2,1)+emin-1)> -1
            nemin = emin - 1;
            pxs = pxs+1;
        else
            nemin = emin;
        end

        %embedded pixels
        wpix = spd;
        wpix(1,1) = spd(2,1)+ nemin;

        %de-sorting using original indeces of input pixels
        temp = sortrows(wpix, 2);
        wpixnew = temp(:,1)';
end
```

A.2 The Jung's PVO-based Embedding

```
clc
clear all
close all

% define the directory for test images
    path = strcat(cd, '\TestImage\');
    pathresult = strcat(cd, '\Result\');
    contents = dir(path);
    path = strcat(cd, '/SipiImg/');
    pathresult = strcat(cd, '/Result/');
    contents = dir(path);

    load eqdata

Method = [];

for F=1:numel(contents)-2
    filename = contents(F+2).name;
    I= imread(filename);
    Io= double(I); %Original Test input image

    [M, N] = size(Io);
    pxs =0;
    pdx =1;

    % eqdata = eqdata(1:20000);
    tic
    Iw = Io;

    for i=1:1:M
        for j=1:3:N-2
            if pdx +2 ≤ numel(eqdata)
                [ Iw(i,j:j+2), unusedbit, pshifted] = jungemb( Io(i,j:j+2), ...
                    eqdata, pdx);
                pxs = pxs + pshifted;
                pdx = pdx+2-unusedbit;
            end
        end
    end
    embtime(F) = toc;

    Ctot(F)= pdx-1; %new line
    bpp(F)= Ctot(F)/(512*512);
    psnrval(F)= PSNR(Iw,Io);    %****
    ssimval(F)= ssim(Iw,Io);    %****


    [pathx, fname, extx]=fileparts(strcat(path, filename));
```

```
48
49      imwrite(uint8(Io),strcat(pathresult, fname,'_orig','.tif'));
50      imwrite(uint8(Iw),strcat(pathresult, fname,'_PVO_1x3','.tif'));
51      Temp = string(strcat(fname,'_PVO_1x3'));
52
53      % mat file writing
54      mfname = [pathresult, fname,'_PVO_1x3','.mat' ];
55      save(mfname)
56
57      if F==1
58         Method = Temp;
59      else
60         Method = [Method;Temp];
61      end
62
63  end
64
65  psnrval = psnrval';
66  ssimval = ssimval';
67  Ctot  = Ctot';
68  bpp = bpp';
69  embtime = embtime';
70
71  % xl file writing
72  xlfname = [pathresult, 'jungPVO','.xlsx' ];
73  Tprop = table(Method, Ctot, bpp, psnrval,ssimval, embtime);
74  writetable(Tprop, xlfname, 'WriteVariableNames', 1);
75
76  fprintf('%s completed \n', fname);
```

A.2.1 $jungemb(\cdot)$

```
1  function [wpixnew, k, pxs] = jungemb(pix, payloadData, pdx)
2  %pix is an input array containing three pixels
3  %idx is an input array containing indeces of the input three pixels
4  %bits is an array of embedding bits.
5
6  pxs=0;
7  idx = [1,2,3];
8  pd= [pix;idx]';         % two columns, 1st for pix and 2nd for its idx
9  spd = sortrows(pd,1);    % sorting pix with idx. For example, 1st row contain ...
       the lowest pix and its location in an image
10
11 emax = spd(3,1)-spd(2,1);
12 emin = spd(1,1)-spd(2,1);
13 k=2;
14          if emax == 0
15              nemax = emax;
```

```
        elseif emax == 1 && (spd(2,1)+emax+payloadData(pdx))<256
            nemax = emax + payloadData(pdx);
            pdx = pdx+1;
            k = k-1;
        elseif emax > 1 && (spd(2,1)+emax+1)<256
            pxs = pxs+1;
            nemax = emax+1;
        else
            nemax = emax;
        end

        if emin == 0
            nemin = emin;

        elseif emin == -1 && (spd(2,1)+emin-payloadData(pdx))>-1
            nemin = emin - payloadData(pdx);
            k = k-1;
        elseif (emin < -1) && (spd(2,1)+emin-1)> -1
            nemin = emin - 1;
            pxs = pxs+1;
        else
            nemin = emin;
        end

        %embedded pixels
                wpix = spd;
                wpix(3,1) = spd(2,1)+ nemax;
                wpix(1,1) = spd(2,1) + nemin;

        %de-sorting using original indeces of input pixels
        temp = sortrows(wpix, 2);
        wpixnew = temp(:,1)';
end
```

A.3 Generating Curves of Rate-Distortion Performance

```
clc
clear all
close all

%% for windows
 path = strcat(cd, '\selected plot data\');
 pathresult = strcat(cd, '\Plot_EC\');
 contents = dir(path);

%% for unix
%path = strcat(cd, '/sPlot/');
%pathresult = strcat(cd, '/FiguresPlot/');
%contents = dir(path);

allnames = {'Peppers', 'Airplane', 'Baboon', 'Lena', 'Barbara', 'Boat', ...
    'Elaine','Lake', 'Peppers'};
lmt = [500, 55000];

for i=1:numel(allnames)
    imname = allnames{i}

    %load data of other schemes
    fname = strcat(imname, '-data.mat');
    load(fname)

    %sPVO
    fname1 = strcat(imname, '_sPVO_1x3.csv');
    temp1 = csvread(fname1);
    cts= temp1(:,1)/10000;
    pts = temp1(:, 2);
    cs = 0.1:0.2:max(cts);
    ps=smooth(interp1(cts, pts, cs,'linear', 'extrap'));

    %PVO by Jung 2016
    fname2 = strcat(imname, '_PVO_1x3.csv');
    temp2 = csvread(fname2);
    ctj= temp2(:,1)/10000;
    ptj = temp2(:, 2);
    cj = 0.1:0.2:max(ctj);
    pj=smooth(interp1(ctj, ptj, cj, 'linear', 'extrap'));

    % He et al. (2018) InfoSc
    fname3 = strcat('He18', imname);
    temp3 = eval(fname3);
    cth = temp3(:,1);
    pth = temp3(:, 2);
    ch = 0.1:0.2:max(cth);
    ph=smooth(interp1(cth, pth, ch, 'linear', 'extrap'));
```

```
48
49      % Ou et al. (2016) JVCI
50      fname4 = strcat('Ou22', imname);
51      temp4 = eval(fname4);
52      cto = temp4(:,1);
53      pto = temp4(:, 2);
54      co = 0.1:0.2:max(cto);
55      po=smooth(interp1(cto, pto, co, 'linear', 'extrap'));
56
57      % Wang et al. (2015) InfoSc
58      fname5 = strcat('Wang15', imname);
59      temp5 = eval(fname5);
60      ctw = temp5(:,1);
61      ptw = temp5(:, 2);
62      cw = 0.1:0.2:max(ctw);
63      pw=smooth(interp1(ctw, ptw, cw, 'linear', 'extrap'));
64
65      % Qu and Kim (2015) SigProc
66      fname6 = strcat('Qu', imname);
67      temp6 = eval(fname6);
68      ctq = temp6(:,1);
69      ptq = temp6(:, 2);
70      cq = 0.1:0.2:max(ctq);
71      pq=smooth(interp1(ctq, ptq, cq, 'linear', 'extrap'));
72
73      % Peng (2015) SigProc
74      fname7 = strcat('Qu', imname);
75      temp7 = eval(fname7);
76      ctp = temp7(:,1);
77      ptp = temp7(:, 2);
78      cp = 0.1:0.2:max(ctp);
79      pp=smooth(interp1(ctp, ptp, cp, 'linear', 'extrap'));
80
81      % Sachnev (2009) IEEE CSVT
82      fname8 = strcat('Sachnev', imname);
83      temp8 = eval(fname8);
84      ctsv = temp8(:,1);
85      ptsv = temp8(:, 2);
86      csv = 0.1:0.2:max(ctsv);
87      psv=smooth(interp1(ctsv, ptsv, csv, 'linear', 'extrap'));
88
89      % Ou et al (2014) SPIC
90      fname9 = strcat('Ou20', imname);
91      temp9 = eval(fname9);
92      cto2 = temp9(:,1);
93      pto2 = temp9(:, 2);
94      co2 = 0.1:0.2:max(cto2);
95      po2=smooth(interp1(cto2, pto2, co2, 'linear', 'extrap'));
96
97  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
98  %% plot for imname
```

```
99      figure('pos',[1 1 900 1200])
100     h=plot(...
101             cs*10000, ps,     '*:r',...
102             ch*10000, ph,     's:b',...
103             cj*10000, pj,     'x:k',...
104             co*10000, po,     'o:g',...
105             cw*10000, pw,     'd:m',...
106             cq*10000, pq,     'd:m',...
107             cp*10000, pp,     'p:c',...
108             csv*10000, psv,   '>:k',...
109             'LineWidth', 2, 'MarkerSize', 8); grid on,
110     legend({'Ours', 'He et al. (2018)', 'Jung (2017)', 'Ou et al. (2016)', ...
111             'Wang et al. (2015)', 'Qu & Kim (2015)', 'Peng et al. (2014)', ....
112             'Sachnev et al. (2009)'},'FontSize',13);
113     ylabel('PSNR (dB)','fontsize',16); xlabel('Embedding capacity (bits)', ...
            'fontsize',16); title(imname, 'fontsize',16)
114     ylim([min(pp(:)) max(max(ps(:)), max(ph(:)))]);
115     set(gca,'fontsize',14)
116     print (strcat('ec',imname, '.eps'), '-depsc')
117 end
```