Institute of Parallel and Distributed Systems (IPVS)
Universität Stuttgart
Universitätsstraße 38
70569 Stuttgart
Germany

Master thesis Nr. 3359

LOCMIC: LOW COMPLEXITY MULTI-RESOLUTION IMAGE COMPRESSION

Haytham W. Hijazi

Course of study:	INFOTECH
Examiner:	Prof. Dr.-Ing. Sven Simon
Supervisor:	M.Sc. Zhe Wang
Commenced:	May 18, 2012
Completed:	November 15, 2012
CR-Classification:	E.4, I.4.1, I.4.2, I.4.10, I.6

ABSTRACT

Image compression is a well-established and extensively researched field. The huge interest in it has been aroused by the rapid enhancements introduced in imaging techniques and the various applications that use high-resolution images (e.g. medical, astronomical, Internet applications). The image compression algorithms should not only give state-of-art performance, they should also provide other features and functionalities such as progressive transmission. Often, a rough approximation (thumbnail) of an image is sufficient for the user to decide whether to continue the image transmission or to abort; which accordingly helps to reduce time and bandwidth. That in turn necessitated the development of multi-resolution image compression schemes. The existed multi-resolution schemes (e.g., Multi-Level Progressive method) have shown high computational efficiency, but with a lack of the compression performance, in general. In this thesis, a LOw Complexity Multi-resolution Image Compression (LOCMIC) based on the Hierarchical INTerpolation (HINT) framework is presented. Moreover, a novel integration of the Just Noticeable Distortion (JND) for perceptual coding with the HINT framework to achieve a visual-lossless multi-resolution scheme has been proposed. In addition, various prediction formulas, a context-based prediction correction model and a multi-level Golomb parameter adaption approach have been investigated.

The proposed LOCMIC (the lossless and the visual lossless) has contributed to the compression performance. The lossless LOCMIC has achieved a 3% reduced bit rate over LOCO-I, about 1% over JPEG2000, 3% over SPIHT, and 2% over CALIC.
The Perceptual LOCMIC has been better in terms of bit rate than near-lossless JPEG-LS (at NEAR=2) with about 4.7%. Moreover, the decorrelation efficiency of the LOCMIC in terms of entropy has shown an advance of 2.8%, 4.5% than the MED and the conventional HINT respectively.

ACKNOWLEDGMENTS

I would like to express my sincere appreciation for my thesis supervisor, M.Sc. Zhe Wang, for his guidance and support over this six-month journey. I would like to thank him for the valuable comments and the significant hints he has given to me. The fruitful technical discussions with him helped me a lot overcome many difficulties encountered and they have been crucial in making the work successful.

I would like to thank Professor Sven Simon for giving me the chance to work on this thesis. I have been always inspired by his character and his achievements since I knew him.
Special thanks to the German Academic Exchange Service (**DAAD**) for their financial and inspirational support through the two years of my study.

I am profoundly grateful to my parents Wail & Suhair Hijazi for their continuous encouragement and advice.
I'm very thankful to all those who made this thesis possible and an unforgettable experience for me.

Finally, to all my friends in Germany and Palestine, thank you for the unconditional support!

Table of Contents

List of Figures

List of Tables

List of Listings

1 Introduction

1.1 Motivation

With the ever tighter weave of today's interactive multimedia and Internet applications, new strategies of information processing have emerged in recent years. In particular, image compression techniques have become very essential to achieve efficient storage and transmission of digital data. Despite of the significant improvements in media storage, e.g., DVD (Digital Versatile Disk), and transmission performance, e.g. ADSL (Asymmetric Digital Subscriber Line), the need for larger storage capacity and faster transmission speeds will continue to be a serious demand and outstrip the available capacity. Accordingly, the image compression algorithms should not only give state-of-art performance, they should also provide other features and functionalities such as progressive transmission in terms of image resolution [9]. In progressive transmission, image information is transmitted in many stages starting from a low resolution version of the image until the full restoration of it. At each stage, the restored image progressively improves as more information is transmitted [10]. Progressive scheme is very significant in multimedia and telebrowsing applications where the user can abort the transmission at an intermediate stage, if the image is not of interest. Thus, using the above scheme leads to significant savings in time and bandwidth. These requirements provoked the need for multi-resolution image compression algorithms to be used, and hence, well-established and extensive studies have been conducted in this field. Nevertheless, one of the paradoxes of technologies evolution is that despite of the above mentioned advances, there is still room for further research in image compression, in general [11].

The fundamental goal of this work in this thesis is to explore the potentials of the multi-resolution framework in the field of image compression at reasonable complexity. Complexity and power-constraints of embedded systems, e.g., digital cameras put a significant challenge on the design of a low complexity and high performance image compression algorithm. In accordance, a **Lo**w **C**omplexity **M**ulti-resolution **I**mage **C**ompression (LOCMIC) algorithm, based on the **H**ierarchical **INT**erpolation framework [12, 18] is proposed. On the other hand, the higher demands of a good reconstructed image quality while minimizing the file size, necessitated the development of the so-called near-lossless concept, in which an additional block, i.e., quantization is added to the lossless algorithm. As a consequence, a novel integration of the proposed algorithm with the Just Noticeable Distortion (JND) [20] for perceptual coding, i.e., adaptive near lossless compression based on the Human Visual System (HVS), is introduced. JND profile can dynamically adjust the quantization step size based on human perception.

1.2 Image compression

The weave of communication systems, computing, networking and social web has a dominant role in one’s life. In the last decade, the world has been witnessing a transformation in the way we communicate. This transformation included the ever-growing Internet and the development of mobile and video communications, particularity the multimedia revolution. However, despite of the emerging growth studies in mass-storage and bandwidth optimization, the evolution of image-intensive applications, e.g., web applications, on the other hand, have been increasing. Storing the image, especially with high resolution, without any kind of compression, consumes a lot of storage space, time, and bandwidth, particularly, in image acquisition and transmission applications. To imagine that, the storage or the transmission of a one-second video (625 lines per frame, 720 samples per line) without compressing it, would require over 20 MB, i.e., 70 GB for a two-hour video. In recent years, image compression techniques have introduced impressive progress e.g., [1, 4, 13, and 18]. These techniques aim to represent the image efficiently, i.e., at lower bit rates. In general, image compression is defined as the processes of observing regularities, i.e., redundancy in the image and trying to eliminate “unnecessary” information about it. One can roughly classify image compression algorithms into two main categories: 1) Lossy image compression 2) Lossless or error free compression. Lossless image compression aims at the exact reconstruction of the original image from compressed representation. However, due to the statistical properties of the spatial distributions in natural images which cannot be well predicted, the lossless scheme seldom reaches significant compression performance. On the other hand, the error-free reconstruction of the image has been an attractive propriety to various applications, e.g., medical applications, where the correctness and the quality of the image are of main concern.

Lossless Image compression adopts two distinct and independent phases: 1) Modeling. 2) Coding. The modeling can be interpreted as an inductive interference problem in which the image is processed in a predefined order, e.g., raster scan. At each time instant of scanning the past image data, an inference is made on the next sample values. The inferred value which is considered redundant information is subtracted from the real pixel value, and only the difference or the residual is encoded using any of the variable length coders, e.g. Huffman, Golomb or Arithmetic.

In lossy or near lossless algorithm, every pixel value in the reconstructed picture is guaranteed to differ from the corresponding original value by a small amount δ. Thus, the lossless mode is a special case when δ=0. To achieve this, the algorithm quantizes regions of image to get higher compression ratios and in consequence sacrificing in the reconstructed image quality. The near lossless perceptual coding will be discussed later in this thesis.

1.3 Background of some lossless image compression algorithms

1.3.1 CALIC: Context-Based Adaptive Lossless Image Codec

In 1996, CALIC has been described, even counted today, among the best, considering only the compression performance. However, the algorithm is computationally complex to implement in a standard. To better understand the CALIC complexity, consider the following table which illustrates the mathematical operations and how many times executed.

Table 1. CALIC complexity [29]

Operation	*Times*
ADD/MINUS	4
MUL/DIV	1
SHIFT	3
ABS	1
Comparison	11

CALIC gives an average lossless bit rate of 2.99 bits per pixel on 18 8-bit tested images in [4] verses 3.89 bits per pixel for lossless JPEG. The prediction in CALIC is based on 12 nearby pixels, which are encoded before the current pixel. Using these pixels, four gradients are calculated in four directions (vertical, horizontal, both diagonals). These gradients can be used as a context for the adaptive prediction formula. In other words, they can determine one of eight different cases of prediction, whether there are sharp edges in the previously mentioned directions or flat regions. Thereafter, the correction factor which is based on six neighboring pixels is added to the prediction. Finally, the prediction error, i.e., residual is Huffman or Arithmetic coded with conditioning one of the eight cases [12].

1.3.2 FELICS: Fast Efficient Lossless Image Compression System

FELICS is a lossless raster scan based coding algorithm proposed in 1993. It has presented a novel idea of using two neighboring pixels for both the prediction and the error modeling. The nearest neighbors (N1, N2) of the pixel N, as shown in Figure 1, are used to obtain an approximate probability distribution for its intensity.

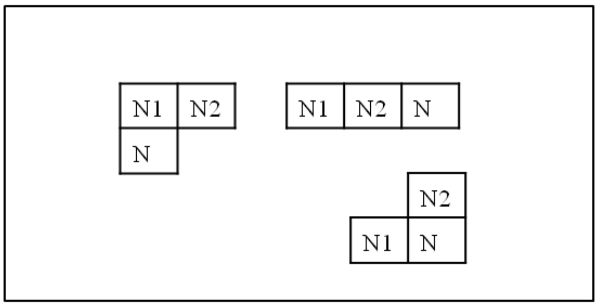


Figure 1 Neighborhood definition in FLICS algorithm

FELICS algorithm assumes the probability distribution of N in its region relative to the nearest neighbors, and coding this region using Huffman coder. For example, the probability of N to be in between N1 and N2 is 50%. This fact is coded using only one bit. Considering other cases where N lies above the maximum of N1, N2 and below their minimum. Each of which occurs 25% and hence coded with two bits. There has been an extension to use Golomb coder recently. This algorithm is very comparable to the lossless JPEG with about five times the throughput [13], but the huge growth in multimedia and telebrowsing applications necessitated the development of a much faster, progressive version of FELICS [13]. The algorithm will be illustrated in the next Chapter.

1.3.3 JPEG-LS

JPEG-LS is a lossless/near lossless image compression standard for continuous-tone images. Like all lossless compression schemes, it consists of two distinct parts: prediction part and coding part. According to JPEG-LS [1], the image is read and processed in raster scan. It follows the structure pioneered by Sunset algorithm, includes [1]:

- The prediction of the value xt+1 is based on a causal template of the past encoded data.
- Determination of the context in which they symbol xt+1 occurs and it is also a function of that causal template.
- The selection of the probabilistic model for encoding the prediction error conditioned with the context.

The JPEG-LS predictor consists of fixed and adaptive part. The fixed part must have some prior knowledge and capable of detecting edges in the image. Specifically, it predicts:

$$P = \begin{cases} \max(a,b) \text{ if } c1 \le \min(a,b) \\ \min(a,b) \; if \; c1 \ge \max(a,b))) \\ a+b-c, \qquad otherwise \end{cases} \tag{1}$$

Where P is the predicted value and (a, b, c) are the pixels available to predict the next symbol as shown in Figure 2 which illustrates the functional block of JPEG-LS.

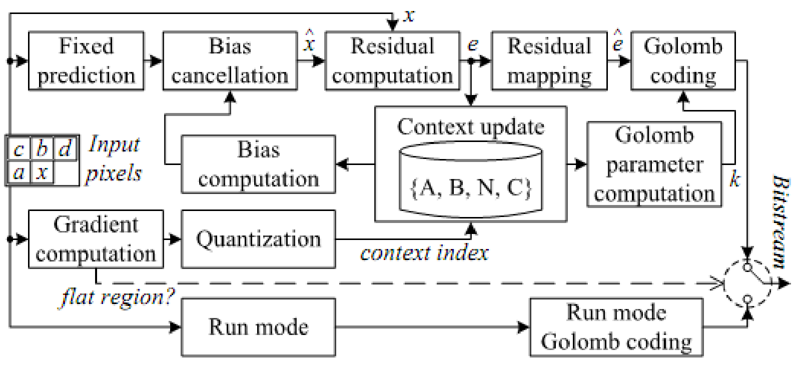


Figure 2 JPEG-LS functional block [15]

Although JPEG-LS ranks among the best in terms of compression performance and complexity, but it has some drawbacks, for example, the neighborhood of the past values, called the causal template is too restricted, i.e., only four pixels can be used in prediction. In consequence, the prediction may not consider higher order complexities of the nearby pixels without employing context modeling in some cases. Furthermore, with the accelerating advances in web technologies, some web applications use a lower resolution of the image, i.e., thumbnail, to save bandwidth and time in case that image is unwanted by the user. Whereas in JPEG-LS and the above schemes, one could not find this important feature.

1.4 Thesis scope

In this thesis, a **Lo**w **C**omplexity **M**ulti-resolution **I**mage **C**ompression (LOCMIC) approach, based on the **H**ierarchical **Int**erpolation (HINT) framework is to be investigated. Lossless compression and near-lossless compression are both in the scope of this thesis. For lossless compression, the performance of various variations of proposed prediction methods is to be investigated. Furthermore, a context based predication correction that is adaptively computed is to be investigated. In addition, a multi-level Golomb coder and a multi-level prediction error correction scheme are in the scope of the thesis as well. For the near-lossless compression, new functional blocks i.e., perceptual quantization will be added to the lossless model. The encoder and the decoder are fully implemented using MATLAB as the developing tool.

1.5 Thesis structure

The thesis will be organized in the following structure:

Chapter 2- Multi-resolution image compression and progressive transmission: This Chapter illustrates the theoretical background of multi-resolution scheme, especially the HINT framework, in addition to a brief illustration of the progressive transmission and its importance.

Chapter 3- LOCMIC Hierarchical Interpolation: This Chapter illustrates the proposed prediction schemes and formulas applied on the proposed algorithm.

Chapter 4- Context based prediction correction: A full explanation of the contexts being used to correct the prediction, and conditional Golomb coding are described. Furthermore, a justification of using different context formulations and components is explained.

Chapter 5- LOCMIC perceptual coding: A superior combination between the HINT framework and the JND perceptual scheme to achieve near lossless compression based on adaptive quantization has been illustrated in this chapter.

Chapter 6- Multi-Level Golomb coding schemes: Three schemes of multi-level Golomb coding have been presented here.

Chapter 7- Comparative results: The compression performance, the time complexity and the image quality are the contents here. In addition, a comparison will be held among different coding algorithms in terms of performance, time and quality and in each level (Prediction, context corrected, and coded).

Chapter 8- Conclusions and future work: This chapter aims to build some conclusions and comments further improvements and future work.

2 Multi-resolution image compression and progressive transmission

2.1 Background of the multi-resolution scheme (HINT)

Multi-resolution image model represents the image with varying spatial resolutions, i.e., pyramid like representation. Each layer serves as a prediction model for the layer lies immediately below. One of the unique proprieties of the multi-resolution image compression algorithms, e.g., Laplacian Pyramid, is the progressive transmission that allows an early visual inspection of the image at different stages. Many important image applications use some processing (e.g., filtering, adjustment, contrast enhancements) that is applied to the transmitted or the stored images. Lossy compression, if used, can add some artifacts that may lead to erroneous interpretation. Particularly, the user in such applications needs to have control over the represented pixels precision and preferably encodes them without any loss or discard. Images in this scheme can be visually inspected and simultaneously fully recovered, but only when necessary. These unique proprieties can only be achieved if a multi-resolution scheme is used. One more advantage of the multi-resolution schemes is related to the coding efficiency. The single resolution image schemes build some assumptions on the image statistical models, e.g., stationary for coding. These models are seldom encountered in real images and hence may lead to higher complexity. Whereas the multi-resolution representation has a unique recursive structure which serves to do the higher complex transformations as a sequence of simple and low order transformations. One of the popular frameworks of multi-resolution model is known as **H**ierarchical **INT**erpolation (HINT), suggested by *Endoh and Yamakazi* [12, 18, 31] on which the basics of this work have been done. The HINT algorithm starts by reducing the original image horizontally and vertically to its half size, and then encodes that low-resolution version. In other words, the process starts by obtaining the residual corresponding to pixels labeled (Δ) in Figure 3 using any lossless algorithm, e.g., DPCM. Then, the intermediate pixels labeled (○) are estimated by the reconstructed labeled pixels (Δ), using a linear prediction formula. Only the resulted residual is sent or achieved. Then, using the available (○), (Δ) pixels, (X) labeled pixels can be interpolated. Finally, from the known pixels, the labeled (*) and (●) pixels can be interpolated as well. The reconstruction process proceeds in the same manner. In practice, an 8×8 window size is used for HINT rather than 5×5 window size as shown in Figure 3.

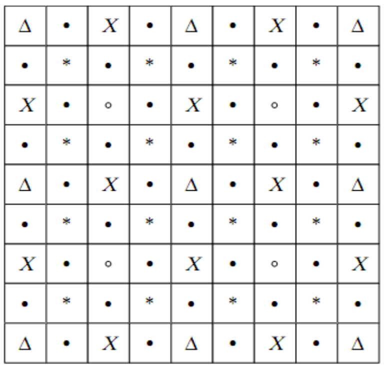


Figure 3 the HINT scheme for hierarchical prediction [12]

The HINT implementation results in [31] have been achieved using the DPCM. Based on [31] also, the HINT has relatively poor compression performance. That is partly due to the five-pass interlaced sampling transversal scheme. In this scheme, the first three passes have to encode sub-sampled images of little correlation. In this work, some enhancements, e.g., proposed interpolation formulas, have been introduced to overcome the compression performance problem. The decorrelation efficiency of the proposed work has been better in about ***4.5 %*** in terms of the first order entropy than the conventional HINT described here.

2.2 Progressive transmission

The amount of information stored as images has been rapidly increasing, especially the remotely sensed images (e.g. weather images, satellite images) and the medical images (e.g. Computed Tomography CT scan, Mammograms). Suppose a user wants to pass over a number of images in a remote database and connected via a 1 Mbps modem. Suppose also that the images size is 1024×1024, and that user browses more than 30 images before finding the image he is looking for. If the image is grayscale i.e., 8 bit per pixel, this process would take some time and consumes bandwidth. Thus, this case is not very practical even with some sort of compression. A good solution is to send an approximation of each image where the approximated version does not need too many bits and sufficiently gives the user a clear idea about the image. In case the user finds it to be of interest, further refinements will be sent or the complete image. To achieve that, the image is divided into blocks and a representative pixel for the block is sent. In other words, an image of 1024×1024 can be approximated with 128×128 sub-sampled image using a block size of 8×8. If this approximation is not sufficient, the user can ask for further refinements, i.e., divide the 8×8 block into four 4×4 blocks.
Figure 4 illustrates that difference between the raster scan transmission and the progressive scheme.

Recently, the world has witnessed a rapid growth of social networks, e.g., Facebook, MySpace, alongside with the great progress in the digital cameras resolution. Hence, the interest in progressive transmission algorithms is expected to increase in accordance.

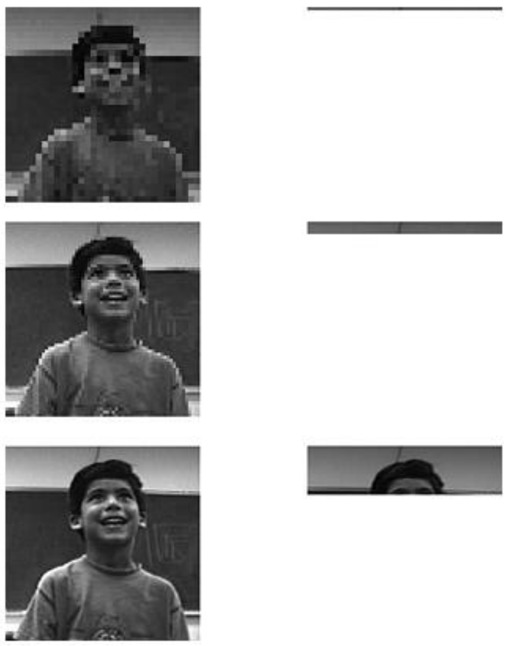


Figure 4. The progressive transmission verses raster scan transmission [12]

The widely known multi-Resolution algorithms are: Multi Level Prediction (MLP) [16], Laplacian and Gaussian Pyramids [17], Wavelet transformation [9] and the Hierarchical INTerpolation algorithm (HINT) [12, 18]. The following sections illustrate these algorithms in brief.

2.3 MLP: Multi-Level Progressive method

In this lossless progressive multi-resolution method [14, 16], the prediction model is based on using nearby pixels in all directions. At the first level, as shown in Figure 5 (a), the known pixels form a square grid. The small dots (midpoints) will be interpolated at this level. After coding these small dots, as shown in Figure 5 (b), the known pixels form a check-board. After scaling the pixel values by $\sqrt{2}$, they are rotated by 45 degrees as shown in Figure 5(c). The known pixels form a square grid again as in (a). In the next level, the midpoints are interpolated and coded, namely all the remaining pixels.

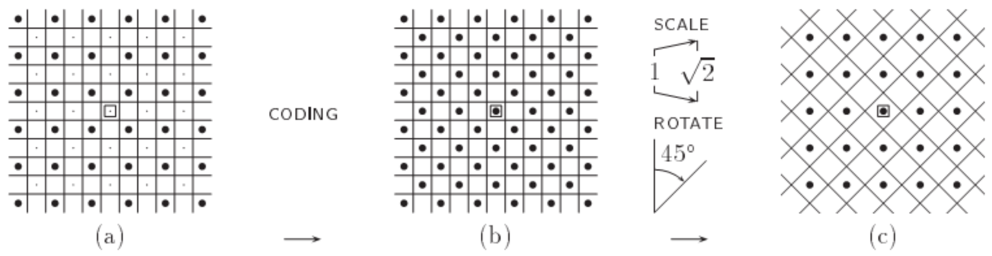


Figure 5. MLP last level prediction neighborhood and coefficients [14]

The algorithm for each successive level L can be briefly illustrated as following:

1. For all pixels P∈ L, P' is predicted and the residual is computed in accordance.
2. For all pixels P∈ L, the variance is computed to be used in encoding the residual.
3. For all pixels P∈ L, the residual is finally encoded by any variable length coders, e.g., Rice.
4. Rotation of the pixels by 45 degrees and scaling them by $\frac{1}{\sqrt{2}}$ as shown in Figure 5 (c).

The predication model proceeds in levels. At each level, pixels are interpolated using specific number of nearby pixels which have been already predicted in the previous levels. The last level uses a 4×4 nearby group for the interpolation. Typically, the predicted value will be the weighted sum of this pixel group. The pixels near the edges have some missing neighbors; hence, the weights of the missing neighbors will be set to zero.

Clearly this method is progressive, where the values of pixels at each level are uniformly selected from the entire image. Suppose that the last level is left without being encoded, the decoder has already half of the original image pixels, and hence can interpolate the remaining pixels using any prediction formula. In practice, skipping the last two levels and letting the decoder interpolate, will end up with an indistinguishable reconstructed image in comparison with the original image.

It has been shown in [32], that MLP scan order is generally worse than the raster scan pixel order that uses all nearby pixels up to distance 2 in terms of the first order entropy in about 1.35% on four selected natural images and one MRI. Using thee of these images, namely, Lena, Goldhill, Barb. The proposed LOCMIC has shown an 8.1% better decorrelation efficiency in terms of the first order entropy.

2.4 Laplacian and Gaussian Pyramids algorithm

The Gaussian-Laplacian image pyramid is a simple multi-resolution pyramid scheme, used for image processing and in particular for image compression. It stores successive low-pass filtered and down-sampled versions of an image to allow a simple access to specific image features of certain frequencies. Suppose that the original image as shown in Figure 6 is represented by g0. This image becomes the bottom of the Gaussian pyramid (level 0). Level has the reduced or low-pass filtered image gL as shown in Figure 6. The pixel values of level 1 are computed using a weighted 5×5 averaging window in level 0. Similarly, each pixel in level 2 is computed using a weighted 5×5 averaging window in level 1, and so forth until reaching the lowest resolution needed (the top of the Gaussian pyramid). On the other hand, the Laplacian image pyramid is an extension to the Gaussian pyramid. It was suggested by (Burt and Adelson 1983). The reduced image gL may serve as prediction for pixels value in the original image g0 by expanding it using an interpolation method, e.g., bilinear. To obtain a compressed representation of the image, the difference, i.e., error between the original image g0 and the expanded image is encoded.

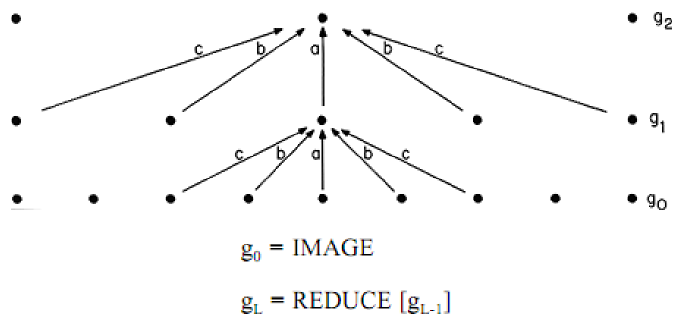


Figure 6 a graphical representation of the process which generates the Gaussian Pyramid [17].

Hence, the Laplacian pyramid is a sequence of error images between two successive levels of the Gaussian. In practice, the pyramid algorithm is based on two complementary functions: EXPAND and REDUCE. As shown in Figure 6, REDUCE generates the lower resolution level of Gaussian pyramid while EXPAND interpolates the lower resolution image up-sample it to its original size.

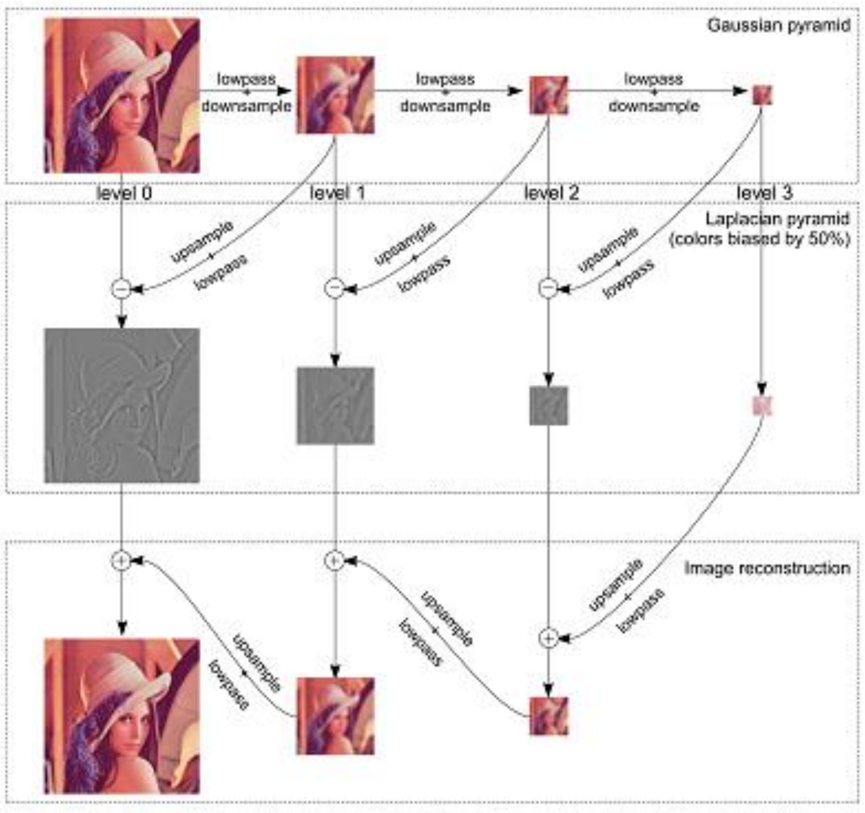


Figure 7 the generation and reconstruction of Laplacian Pyramid [30].

The Laplacian-Gaussian pyramid encompasses a low, band-pass filter respectively, in other words, the Gaussian pyramid can be viewed as a set of low-passed filter copies of the original image, while the Laplacian pyramid can be viewed as a set of band-pass filtered copies of the image. The computational complexity of this scheme is simple because it performs the computations locally and may be performed in parallel. Furthermore, the same computation is iterated to build each level from the predecessor level.

2.5 Progressive FELICS: Fast, Efficient, Lossless Image Compression System

This Algorithm retains the same hierarchical pixel sequence of MLP (refer to Section 2.3) to obtain a progressive coding. The difference is in the image prediction model. The prediction is based on just four nearby pixels. Two of the four nearest known pixels are selected. Coding of the pixels is then realized using single bits, adjusted binary codes, and simple prefix code, e.g., Rice.

Selecting the two pixels is done using any of three ways: 1) maximum and minimum values 2) the two middle values 3) a selected pair of spatially adjacent pixels. Practically, it has been shown in [14] that using the middle values give the best compression results in general.
The context model in progressive FELICS uses the absolute difference between the selected two predicting pixel values. One bit is used to indicate whether the predicted pixel is in the range. Accordingly, an adjusted binary code for the exact value within the range is used. The coding parameter is separate for each context. As the difference between the nearby pixels is correlated with the variance of the image, it has been found that the context that is based on such small differences leads to more sharply-peaked distribution. Progressive FELICS coding is based on the original FELICS algorithm as shown if Figure 8.

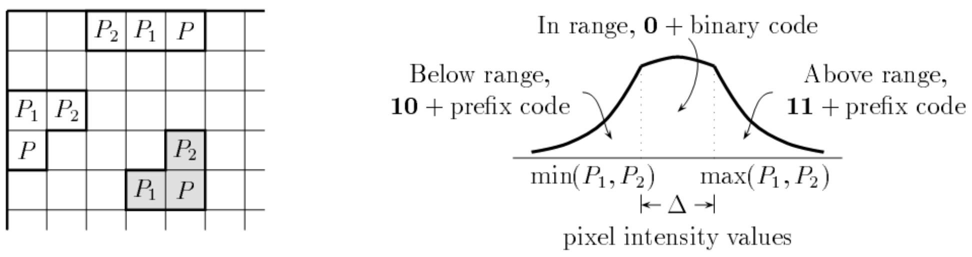


Figure 8 the coding process of FELICS [14].

The left depict of the above Figure shows the context coding is consisting of two nearest neighbors P1, P2. The right depict shows the coding for different intensity ranges relative to the maximum and the minimum of the two context values .The probability of an in-range event equals to that of an out-range event. The same is with the above, below range, they are equally probable.
The progressive FELICS has shown a better compression performance about ***1%*** better than the normal FELICS (refer to [13], [14] for more details).

3 LOCMIC hierarchical interpolation

3.1 Overview

The conventional Hierarchical Interpolation (HINT) is a multi-resolution image compression algorithm that has been found to be very efficient to compress images especially the medical, e.g. X-Ray [12, 18]. The LOCMIC algorithm uses different pixel order selection. Typically it starts by reducing the original image to its half size by choosing every odd pixel (circles in Figure 9 and 10) these pixels is then encoded using JPEG-LS algorithm. JPEG-LS has been used in this work because of its high compression performance and low complexity [1]. In the second pass, the encoded (circle) image is enlarged to its original value. The circle pixels are then used to interpolate the intermediate pixels (stars) by averaging the four corner pixels or taking the median value. Subsequently, the (pyramids) pixels can be interpolated from the available (circle) and (stars) pixels which exist as four direct neighbors. Finally, the available (circles, stars, pyramids) pixels contribute to interpolating the remaining pixels (rectangles) as shown in Figure 10. To ensure the reversibility, the interpolation results are rounded to the nearest integer.

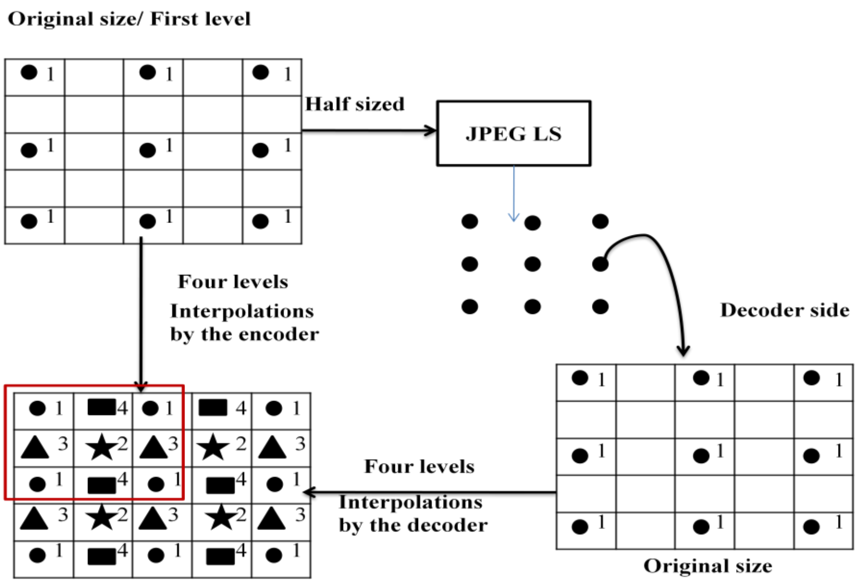

Original size/ First level
Half sized
JPEG LS
Four levels Interpolations by the encoder
Decoder side
Four levels Interpolations by the decoder
Original size

Figure 9. A general depict of the HINT algorithm used in this work

At the decoder side, the reconstruction process proceeds in the same manner. Upon reconstructing the (circle) pixels, the same interpolation algorithm is used to interpolate the (stars) pixels and adding the received residual. Subsequently, all other pixels are reconstructed in the same fashion.

Figure 10 shows a detailed view of the pixels available at each level. A different coding order is of course possible, e.g., encoding the (pyramids) pixels in the 2nd step, the (stars) pixels in the 3rd step and the (stars) pixels in the 4th step. It has been found by our extensive simulations, however, that the encoding order as shown in Figure 10 leads to good compression performance in general.

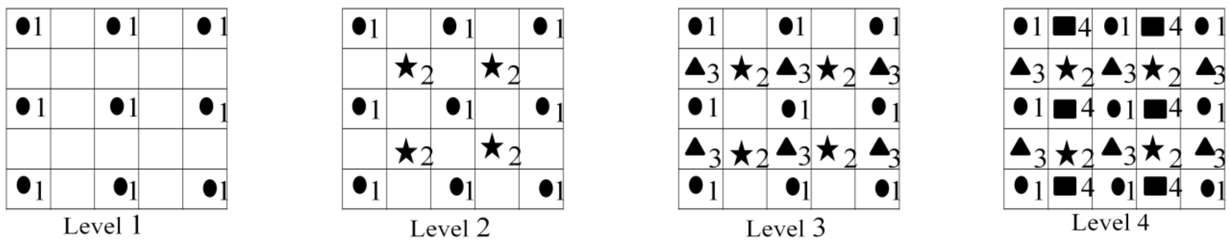


Figure 10. The hierarchical depict of the LOCMIC

As discussed in Chapter 2, the conventional HINT algorithm uses in practice, an 8×8 window size to interpolate the midpoint pixels. In this work, only a 3×3 window size has been used in the proposed schemes. Empirically, the proposed schemes have shown better decorrelation results in about 4.5% than the conventional HINT in terms of the first order entropy. The experiments have been on different known grayscale natural images (see chapter 7).

3.2 Proposed prediction schemes for levels 1, 2 and 3

The first level (circles) is entirely encoded using JPEG-LS [1] algorithm because of its high compression performance and low complexity. The prediction scheme at the second level (stars) proceeds as follows:

1. The average of the four nearby pixels (circles), as shown in Figure 11, is calculated.
2. The median value of these pixels is calculated.
3. In step (2), two median values are resulted, thus their average is calculated.
4. Taking the average of step (1) and (2) as the final predicted value.

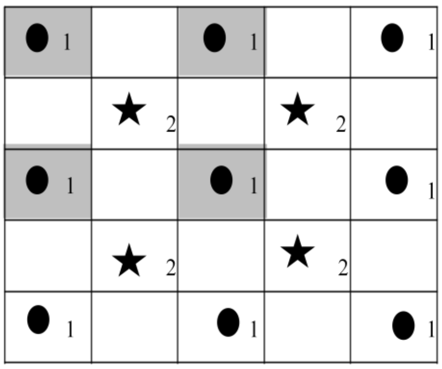


Figure 11 The second level prediction map

Calculating the median value has an advantage of excluding the noise pixel, if it is encountered in the four nearby pixels. Moreover, the calculation of median is computationally simple.

The third (pyramids) level applies the same prediction scheme used in level 2 (stars). But thanks to the direct close nearby pixels available at this level, as shown in Figure 12, the pixels average has also been found to deliver precise prediction values in general.

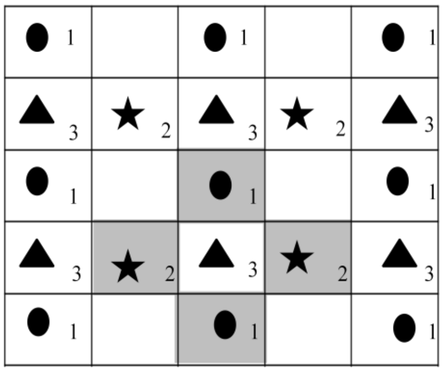


Figure 12 the third level prediction map

Of course, other schemes and interpolation formulas have been investigated as well, but the best among them which give the best compression results, have been illustrated here.

3.3 Proposed novel prediction schemes for level 4

A wide spectrum of prediction formulas and schemes has been proposed and applied on the last level (rectangles) level, thanks to the availability of ¾ pixels of the original image. The following sections discuss the proposed schemes briefly.

3.3.1 “K” tuned average LOCMIC (KLOCMIC)

The idea of this method is inspired from removing the grain from images [19]. Removing grain was primarily used as denoiser[1] and over time it has evolved into a general plug-in which manipulates the pixels in terms of their nearby pixels. In this method, the first step is to calculate the average of the four neighboring pixels as shown in Equation (2).

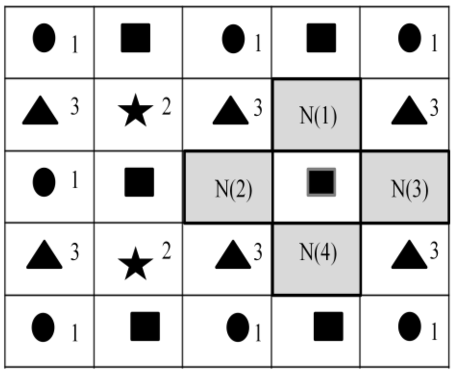


Figure 13. The last level neighborhood pixels map

$$\text{Average}=0.25\times N(1)+0.25\times N(2)+0.25\times N(3)+0.25\times N(4) \quad (2)$$

$$P=\max(\min(\text{Average}(N(1)\text{-}N(4)), N(5\text{-}k)), N(K)), k=1, 2, 3, 4 \quad (3)$$

Where P is: the predicted value, average is: average of the pixels N(1), N(2)…N(4), K: is the equation parameter and takes the values from 1-4. The condition here is: N(1) ≤ N(2) ≤N(3) ≤N(4), i.e., they are in an ascending order.

The calculated average is then clipped by two values, namely N(5-k) and N(K), as shown in Equation (3) and Figure 13, using the minimum and the maximum function. An empirical value of K that leads to good compression results has been found to be 2. In other words, the four pixels average is clipped by their median value.

To see the effectiveness of this method, consider the following example: Suppose that the real value of the pixel to be interpolated is 163, as shown in Figure 14, and suppose that the right most pixels is a noise value and equals to 244.

[1] Denoiser is the process of removing noise from pictures (e.g., Median filter)

x	163	x
162	**163**	244
x	164	x

Figure 14 . A noise value case

If the average is considered only, the interpolation result will be 183.25; the value is rounded to nearest integer to become 183. Accordingly, the error value is equal to -20. However, if Equation (3) is applied, the result should be like this:

P=max (min (183, 164), 163), where the min(183, 164) is 164 and the max(164, 163) is 164. The resulted prediction value P is 164, and the error value is -1 which is nearer to zero than the -20 and hence better prediction is achieved. This process resembles the functionality of a filter, but fortunately without needing two passes of manipulation.

3.3.2 The Two Median Edge Detectors (MED) LOCMIC (TMLOMIC)

JPEG-LS [1] encodes the pixels in raster scan, thereby the MED predictor can only use the pixels a1, b1, c1 as shown in Figure 15 to predict the next value to be encoded. However, the proposed TMLOCMIC applies the MED algorithm two times. The first time, it performs as if it encodes in raster scan using only the pixels a1, b1, c1 (the light gray shaded area in Figure 15). The second time, the pixels a2, b2, c2 (the dark grey shaded area in Figure 15) are exploited to predict the same pixel value as shown in Equations (4) and (5).

$$\mathrm{P1}=\begin{cases} \max(\mathrm{a1},\mathrm{b1}) \text{ if } \mathrm{c1} \le \min(\mathrm{a1},\mathrm{b1}) \\ \min(a1,b1)\ if\ c1 \ge \max(a1,b1) \\ a1+b1-c1\ otherwise \end{cases} \tag{4}$$

$$\mathrm{P2}=\begin{cases} \max(\mathrm{a2},\mathrm{b2}) \text{ if } \mathrm{c2} \le \min(\mathrm{a2},\mathrm{b2}) \\ \min(a2,b2)\ if\ c2 \ge \max(a2,b2) \\ a2+b2-c2\ otherwise \end{cases} \tag{5}$$

Finally, the average of p1 and p2 is calculated and rounded to the nearest integer.

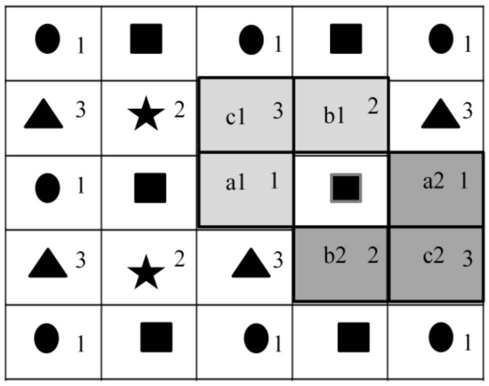


Figure 15. The Two Median Edge Detector HINT (TMHINT)

This method takes the advantage of the original MED [13] in performing a primitive test for both the horizontal and the vertical edges existed in the image. For example, if a vertical edge exists, P1 will pick b1, and P2 will pick b2 using Equations (4) and (5) respectively. Similarly, in case a horizontal edge exists, P1 will pick a1 and p2 will pick a2 using Equations (4) and (5) respectively. If no edge is detected P1 chooses a1+b1-c1, p2 chooses a2+b2-c2.

3.3.3 Median value LOCMIC (MLOCMIC)

As discussed in section 3.2, calculating the median value has been found to be a good choice because it is computationally simple and in some cases, as shown in Figure 16, has an advantage of omitting the noise value from the prediction. To illustrate that, consider the following example:

155	x	151
x	**160**	x
266	x	159

Figure 16 the second level (stars) pixel values

If the original value of the pixel to be predicted is 160. The two median values are 155, 159. Taking the average and rounding the result to the nearest integer, the predicted value is 157. Accordingly the error value will be (160-157) =3. If only the average of the nearby pixels was considered in this example, the error value would be -23 which is for sure worse. However, this method may encounter two or more noise pixel values. In such case, it would not perform well and give the expected precise prediction.

3.3.4 The Comprehensive LOCMIC (CLOCMIC)

Each of the above schemes has performed well on some kind of images while better on others. Therefore, a comprehensive method has been introduced. The idea is to encompass the advantages of the proposed methods and assign certain scores to each of which. In addition, gradients in four directions, as shown in Figure 17, are calculated to capture the activity of the pixels, e.g., edginess, smoothness. The realization of this scheme is as follows:

1. The gradients in four directions are calculated (two diagonals and two perpendicular gradients) as illustrated in Equations (6-9).
2. Calculate the minimum gradient value among the four gradients as given by Equation (10).
3. The components of the minimum gradient are averaged as shown in Equation (11).
4. Use Equation (3) which is the "K" tuned average to determine PK in Equation (12).
5. Assign weighted scores and average the used values as shown in Equation (12).

The notation here expresses the pixel value by I located at (x, y).

$$\mathrm{Grad1}=I(x-1, y-1)-I(x+1, y+1) \tag{6}$$

$$\mathrm{Grad2}=I(x-1, y+1)-I(x+1, y-1) \tag{7}$$

$$\mathrm{Grad3}=I(x, y-1)-I(x, y+1) \tag{8}$$

$$\mathrm{Grad4}=I(x-1, y)-I(x+1, y) \tag{9}$$

$$\mathrm{GradMinimum}=\mathrm{Min}(|\mathrm{Grad1}|,|\mathrm{Grad2}|,|\mathrm{Grad3}|,|\mathrm{Grad4}|, |\mathrm{Grad4}|) \tag{10}$$

$$\mathrm{GradAverage}= \mathrm{average}\ (\mathrm{GradMinimum}) \tag{11}$$

$$P=w1\times \mathrm{Median}+w2\times \mathrm{Average}+w3\times \mathrm{GradAverage} +w4\times PK \tag{12}$$

Where P is the predicted pixel value, Median is the median of the four neighbors; GradMinimum calculates the minimum absolute value of the gradients. GradAverage takes the result of Equation (10) and calculates the average of the gradient components, i.e., the gradient pixels. Finally, a weighted sum is used to combine the computed values.

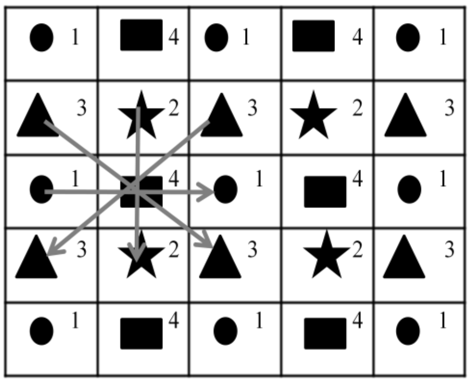


Figure 17 the gradients directions

Experimentally, the appropriate values of scores have been empirically found to be W4=0.85, W1, W2, W3= 0.05.

This method has been among the best formulas presented in terms of compression performance. However, it lacks of the simplicity found in pervious proposed methods.

3.4 Summary

The Hierarchical Interpolation (HINT) is a multi-resolution image compression scheme which supports the progressive transmission scheme. Based on its framework, the proposed LOCMIC has been proposed.

Different and new interpolation methods have been proposed such as, “K” tuned average HINT (KGHINT), The two Median Edge Detectors (MED) HINT (TMHINT), Median value HINT (MHINT), and the Comprehensive HINT (CHINT) among which the KGHINT and the CHINT have been found to give good compression results in general.

As reported in Chapter 7, the experimental results have shown an advance in compression performance in terms of the first order entropy over the conventional and the known prediction schemes such as: MED, bilinear interpolation, MLP.

4 Context based prediction correction

The prior knowledge of the image structure such as texture pattern, edginess and smoothness can be exploited by fitting such patterns to the parametric distributions, e.g. Laplacian [3]. That in turn allows a large number of contexts to capture the activity of the image. The basic idea behind context modeling is that the currently processed pixel may depend on previously processed pixels taken as parameters. Hence, the selection of symbols to estimate the conditional probability such that the code length is minimized is of vital importance [2]. However, if the number of these parameters is too large with respect to the coded image, the count statistics may not mange to have good samples to estimate the conditional probability. This problem is called “sparse of context” or “context dilution” [4]. The statistical modeling of the source has revealed [3] that the prediction error i.e., the residual, of the continuous-tone images can be described by a Two Sided Geometric Distribution (TSGD) centered at zero, as shown in Figure 18.

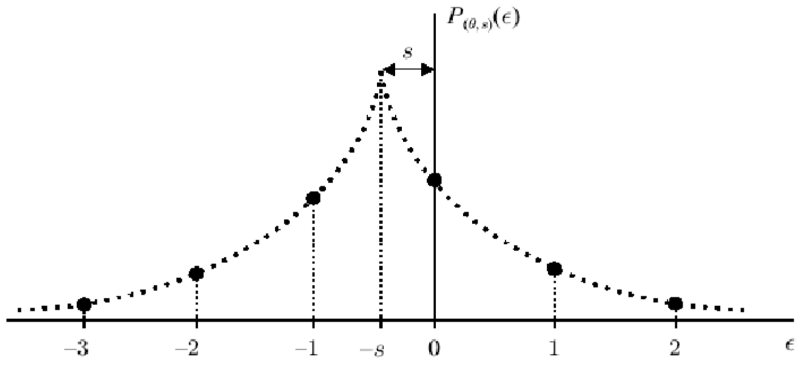


Figure 18 the Two Sided Geometrical Distribution (TSGD) [5]

The context Q that conditions the encoding of a pixel in JPEG-LS [1] is determined by the gradients or the differences of the neighboring pixels. The context is supposed to capture the image activity surrounding the current pixel. The bias of “Q” is estimated, to be added or subtracted from error modeler from the initial prediction [5]. The bias is totally specified from the accumulated magnitudes of prediction error “A”, and the count “N” of encoded pixels belong to a specific context “Q”. The context variables (A, B, N) are then updated. Finally, the residual is mapped to non negative integers to be encoded.

Because the model cost is very important and needed, the gradient values are further quantized. In principle, the number of regions into which the context difference is quantized has been adaptively selected and optimized. The regions are: -T,…0,1,…T. where T has been selected by JPEG-LS [1] to be 4. Thus, the number of contexts is in total of $(2T+1)^3=729$. A further reduction could be introduced to the number of contexts due to the symmetry in the regions to be in total of $((2T+1)^3+1)/2=365$. In practice, the context is falling into the following default quantization regions: {0}, ∓{1, 2}, ∓{3, 4, 5, 6}, ∓ {7, 8…20} ∓{e|e>20}.

4.1 LOCMIC context formulation

In practice, it has been observed [1, 19], that the fixed prediction does not fully omit the statistical redundancy in the image. Therefore, a prediction correction module based on the image context has been presented in the proposed LOCMIC. The framework, i.e. HINT allows to use the nearby pixels in different directions as discussed in Chapter 2, 3, and hence, that should capture the statistical image proprieties in a better way. As discussed in Chapter 3, the first level (circles) is entirely encoded by JPEG-LS, thus, the same context formulation at this level will apply. The second level (starts) has no sufficient correlation between pixels. Thereby, this level has not shown great performance after introducing the context model. For the remaining levels, however, the efficiency after applying the context based prediction correction has been about 2.2%. The next sections illustrate the presented context formulation which is mainly based on the JPEG-LS context model [1], but with different components formulations.

4.1.1 Second level context formulation

Thanks to the relatively low correlation of the available pixels at this level, the formulation of a context that could capture higher dependencies between these pixels has been a bit challenging. Figure 19 shows the gradient directions used to form the context.

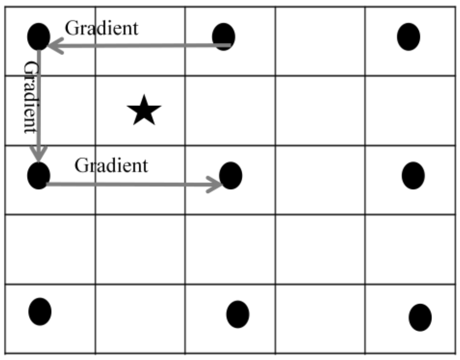


Figure 19 the context formulation of the second level

The model cost is paid an attention here; thus, at the first and the second levels, uses only three context components. Based on the default quantization regions used by JPEG-LS [1], each component will fall within an index of range –T, …, -1, 0, 1, … T, where T=4. An overall contexts will be of $(2T+1)^3$=729. Further minimization could be achieved by merging the contexts which have opposite sign. The number of contexts the is of $((2T+1)^3+1)/2$=365 contexts. Equations 13-15 show the proposed context formulation at the second level (starts):

$$C2=I(x-1, y+1) - I(x-1, y-1) \tag{13}$$

$$C3=I(x-1, y-1) - I(x+1, y-1) \tag{14}$$

$$C3=I(x+1, y-1) - I(x+1, y+1) \tag{15}$$

Where Ci: the context component number. i=1, 2, 3. I(x, y) is the grayscale value of the pixel located at (x, y).

4.1.2 Third level context modeling

At this level, both the circles, stars symbols are available. The idea is to make a two-pass context formulation, where in the first pass; the average of the nearby pixels is calculated and preserved for the second pass. In the second pass, the calculated average is included in forming the context components. The used components are illustrated in Figure 20 and Equations (16-19).

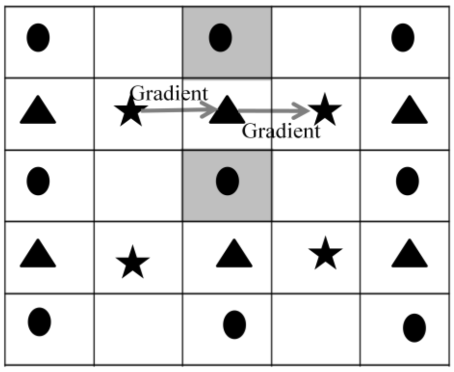


Figure 20 the context formulation of the third level

$$C1=I(x-1,y) \tag{16}$$

$$C2=\lceil 0.5 \times \text{Errval}(x-1,y) + 0.5 \times \text{Errval}(x+1,y) \rceil \tag{17}$$

$$C3=I(x,y-1) - P(x,y) \tag{18}$$

$$C4=P(x,y) - I(x,y+1) \tag{19}$$

Where x, y are the indices of the current pixel, I(x y) is the grayscale value of the pixel located at (x, y), Errval is the error value, i.e., residual, P(x, y) is the predicted value from the first pass.

Using the predicted value is expected to help in gathering much statistics on the available pixels to form a good context. Empirically, at this level the efficiency of applying the prediction correction is about 1.7%. However, the number of contexts in this level is higher than the first and the second level that is of $T\times(1+T)^3+(2\times T+1)^3/2= 865$. Where T=4 which represents the number of regions used in the quantization.

4.1.3 Fourth level context modeling

Likewise, this level follows a similar fashion as the third level, but with very minor changes. Particularly, in gradients directions. Figure 21 and Equations (20-23) illustrate the approach.

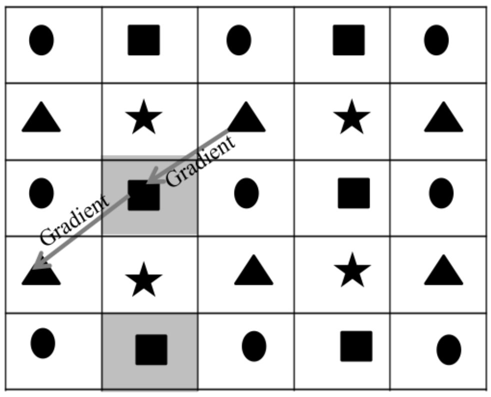


Figure 21 the context formulation of the fourth level

As shown in Figure 21, the two pass approach has been used here as well. To better understand the formulations, Equations (20-23) illustrate the approach.

$$C1=I(x-1, y) \tag{20}$$

$$C2=[0.5 \times Errval(x-1, y) + 0.5 \times Errval(x+1, y)] \tag{21}$$

$$C4=P(x, y) - I(x+1, y-1) \tag{22}$$

$$C3=I(x-1, y+1) - P(x, y) \tag{23}$$

Where, (x, y) are the spatial coordinates of a specific pixel, I(x y) is the grayscale value of the pixel located at (x, y), Errval is the error value, i.e., residual, P(x, y) is the predicted value from the first pass.

Similarly, the number of contexts in this level, however, is higher than the first and the second level that is of $T\times(1+T)^3+(2\times T+1)^3/2= 865$. Where T=4 which represents the number of regions used in the quantization.

4.2 Prediction correction

As discussed in this Chapter, the fixed prediction formulas cannot adequately capture the complex relationship between the predicted pixel value and its surrounding activity. In other words, it cannot fully omit the redundant statistics in the image. Therefore, the context formed in the previous section has been used to adaptively refine the prediction considering the higher-order structures, such as the texture patterns for further compression gain. In practice, the mechanism for the correction depends on the SIGN variable and the cumulative correction values stored in C(Q) [6] as shown in Listing 1. The SIGN follows the context sign after quantization.

```
If (SIGN==-1)
    P(x, y) = P(x, y) - C (Q(x, y));
Else
    P(x, y) = P(x, y) + C (Q(x, y));
```

Listing 1. Prediction correction from the distribution bias

Thereafter, the corrected prediction value shall be clamped to the range [0, maximum] as shown in Listing 2; in this case, the maximum value is equal to 255.

```
If P(x, y)>255
P(x, y) = 255;
Else if P(x, y) <0
P(x, y) = 0;
```

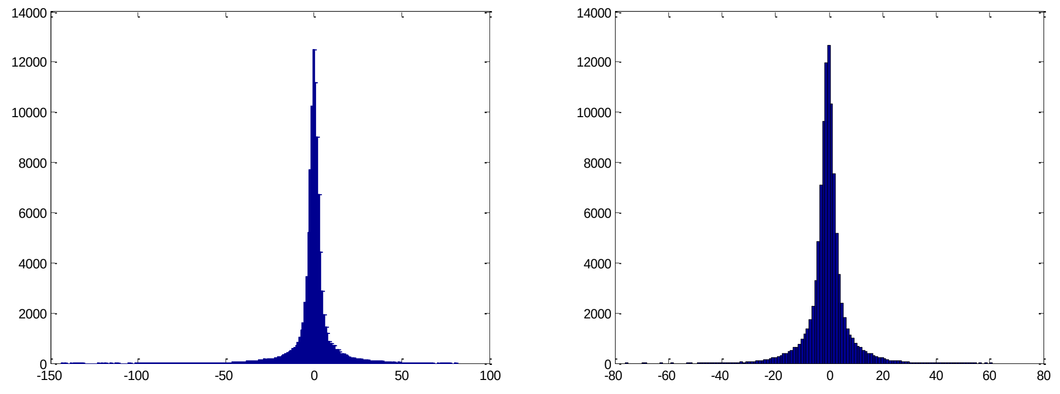
Listing 2. Prediction clamping

Finally, the prediction error is recomputed using the corrected prediction value and the sign flipping process takes place as demonstrated in Listing 3.

```
Errval(x, y) = I(x, y) – P(x, y);
 If SIGN(x, y) <0
Errval(x, y) = -Errval(x, y);
```

Listing 3. Residual computation and flipping the sign

To better understand the prediction correction effect, Figure 22 shows a histogram of the third level (pyramids) residual without and with the correction applied on the known Barb image.



(a) (b)

Figure 22 a) The histogram of the barb residual at the third level without correction (Entropy 4.82 bpp). b) The histogram of barb residual at the third level with correction (Entropy 4.53 bpp)

As shown in Figure 22, the residual pixel values have fallen inside a narrower range after the correction, thus the residual has become more peaked around zero with less variance value. This level has empirically achieved about ***2.7%*** better decorrelation after the adaptive refinement.

4.3 Summary

The context modeling has been a crucial part in any image encoding system. It can adaptively correct the fixed interpolation value based on image activity and depending on higher order collected statistics.

JPEG-LS [1] uses the three differences between the nearby pixels i.e., gradients and quantize their values. In HINT scheme, the same approach applies, but with some changes to adapt with the multi-resolution framework.

The first level is entirely modeled and encoded using JPEG-LS, whereas the second level uses three gradient components also but in different directions, to exploit the 360 degree neighborhood availability.

The third and the last level uses four components, they are in general: 1) Real pixel value 2) Prediction Error 3) Two gradients.

The context variables are updated and manipulated as similar as JPEG-LS [1] algorithm does.

5 LOCMIC perceptual coding

5.1 Perceptual coding

Perceptual coding reduces the image file size by analyzing certain parts and activities in the image such as: brightness, texture, fixation point and so on, to which the Human Visual System (HVS) is sensitive. The HVS is a description of the eye, receptors, and psychological processing of images in the human brain. Based on this definition, different models have been proposed to characterize the HVS. An example of such model is the Just Noticeable Distortion (JND) which is superiorly applied on the Hierarchal Interpolation scheme in this work.

In general, lossy or near lossless image compression allows a certain error or loss in the reconstructed image quality, and accordingly, achieves higher compression ratios. In particular, the visual lossless or the perceptual-based coding attempts to distinguish between signal components in terms of their visibility and non visibility to the human receiver [7]. In other words, perceptual coding allows further removal of perceptual redundant information by adapting the coding scheme to the human perception [8]. The target is minimizing the bit rate for a desired perceptual goal distortion. To achieve that, the masking proprieties of the HVS shall be exploited by establishing a dynamic threshold of JND and Minimally Noticeable Distortion (MND), based on psychophysical proprieties. That means, the Quantization Step Size (QSS) is not fixed here as the conventional near lossless schemes, e.g., near-lossless JPEG-LS. In JPEG-LS near lossless algorithm [1] the non adaptive QSS is given by Equation (24).

$$\mathrm{QSS}=2\times \mathrm{NEAR}+1 \tag{24}$$

In Equation (24), NEAR is the parameter that decides the maximum allowed deviation between the original image and the reconstructed image.

In JND, the QSS calculation replaces the NEAR in Equation (24) by the calculated JND values for each pixel, i.e., QSS=2×JND(x, y).

The JND computation will be illustrated with equations in the next section.

In this work, a novel attempt has been presented to apply the visual lossless JND profile on the HINT framework to achieve a perceptual coding version of LOCMIC.

5.2 JND profile

The JND profile provides a dynamic visibility threshold of distortion for images being studied [8]. JND profile has two main factors that affect the error visibility threshold. 1) The background luminance 2) The texture masking effect. According to Weber's law [7, 20], the HVS is more sensitive to luminance contrast than absolute luminance value. More precisely, "If the luminance of an eye stimulus is just noticeable from the surrounding luminance, the ratio of just noticeable luminance difference is approximately constant", said Weber [20]. Based on that, the high spatial heterogeneity in the image background makes the error less visible, i.e., spatial masking. The calculation of the spatial masking is performed by weighting each pixel as the sum of the horizontal, vertical and diagonal slopes at the neighboring pixels. Because of the different HINT framework, some modifications have been introduced to the original JND profile. The mathematical description of the JND profile is discussed in the next subsections: For the first level (circles) the conventional JPEG-LS near lossless algorithm is applied with different NEAR values, (e.g., 1, 2, and 3). The other levels have been individually handled as follows.

5.2.1 Second level (stars) JND

As discussed in Chapter 3, for each pixel to be predicted, four nearby pixels contribute to the prediction. The modified JND profile starts by calculating the average gray level bg(x, y) of these four neighbors using a 3×3 window as given by Equation (25). Then the maximum weighted gradient (mg) is calculated using four operators G1, G4, G11, and G44 as shown in Figure 23. The gradient operators are different from the original JND profile to fit with the HINT framework used in this work. Where G1, G11 detect the horizontal edges. G4, G44 detect the vertical edges. Then the JND calculation proceeds, f1 and f2 are calculated, where f1, f2 represents the spatial masking effects and the visibility threshold due to the background luminance respectively. To realize that, the following Equations and operators are computed:

$$bg(x,y)=0.25\times I(x-1,y-1)+0.25\times I(x-1,y+1)+0.25\times I(x+1,y-1)+0.25\times I(x+1,y+1) \quad (25)$$

$$Gk = 1/f\times\sum_{i=-1}^{i=1}\sum_{j=-1}^{j=1} I(x-i,y-j)\times Gk,\ k=1, 2, 11, 22 \quad (26)$$

* f is the summation of the positive operator weights

$$\text{mg}=\text{Max}\{|(G1)|,\ |(G2)|, |(G11)|,\ |(G22)|\} \tag{27}$$

$$\text{f1}(\text{bg}(x,y), \text{mg}(x,y)) = \text{mg}(x, y) \times \alpha\ (\text{bg}(x, y)) + \beta\ (\text{bg}(x, y)) \tag{28}$$

$$\text{f2}(\text{bg}(x, y) = \begin{cases} T0 \times \left(1 - \left(\frac{\text{bg}(x,y)}{127}\right)\right)^{1/2} + 3 & \text{if } \text{bg}(x, y) \le 127 \\ \gamma \times (\text{bg}(x, y) - 127) + 3 & \text{if } \text{bg}(x, y) > 127 \end{cases} \tag{29}$$

$$\alpha\big(bg(x,y)\big) = bg(x,y) \times 0.0001 + 0.115 \tag{30}$$

$$\beta\ (\text{bg}(x, y) = \lambda\text{-bg}(x, y)\ \times\ 0.01 \tag{31}$$

Where T0 in f2 masks the visibility threshold when background gray level equals to zero, and the slope λ of the linear function relating the background luminance to the visibility threshold when bg(x, y) is greater than 127. λ represents the effect of the average amplitude of visibility threshold due to the spatial masking effects. The initial values of these parameters are as follows: T0=17, $\gamma = \frac{3}{128}$, $\lambda = \frac{1}{2}$.

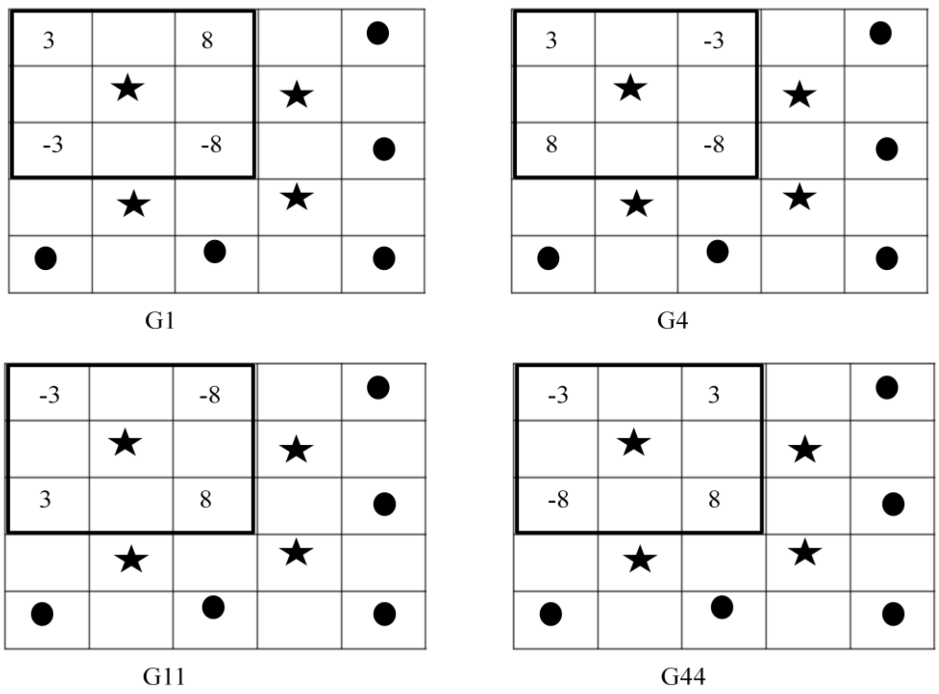


Figure 23 the gradient operators-Level2 (stars)

G1 calculates the average luminance changes giving the priority to the upper horizontal direction. G4 calculates the average luminance giving the priority to the left vertical direction. While G11 calculates the average luminance changes giving the priority to the lower horizontal direction. G44 calculates the average luminance giving the priority to the right vertical direction. Empirically, we could claim that using the 3×3 window used in detecting the edges has delivered good compression results with a perceptually accepted image quality (see chapter 7). This claim is based on extensive experiments. The experiments have shown, particularly in the HINT framework that the active area from which the image texture is to be detected could be within the 3×3 window in natural images.

5.2.2 Third level (pyramids) JND

Likewise, the algorithm at this level starts also by calculating the background gray average bg(x, y) as given by Equation (32).

$$Bg(x,y)=0.25\times I(x,y-1)+0.25\times I(x,y+1)+0.25\times I(x+1,y)+0.25\times I(x-1,y) \qquad (32)$$

The operators here G2, G3 are also different from the original JND as shown in Figure 24.

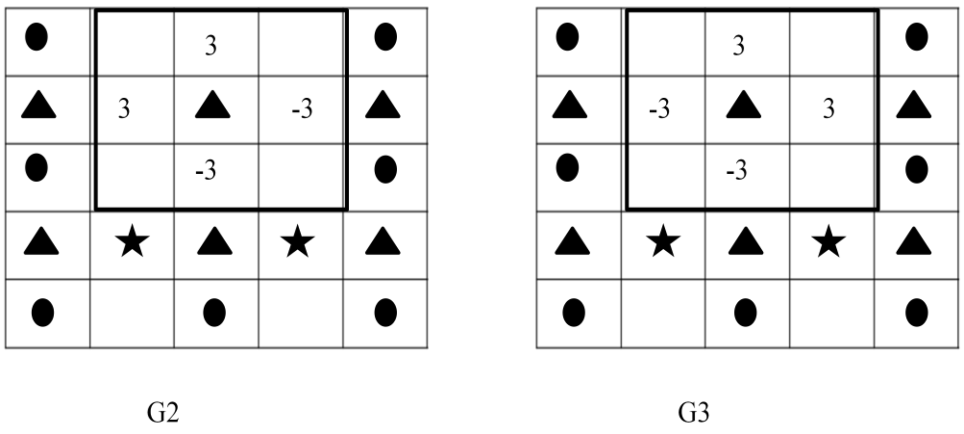


Figure 24 . The gradient operators-Level3 (pyramids)

The rest of the formulations follow a similar fashion as in the second level and as illustrated again here:

$$\text{Gk} = 1/\text{f}\times\sum_{i=-1}^{i=1}\sum_{j=-1}^{j=1}\text{I}(\text{x}-\text{i},\text{y}-\text{j})\times\text{Gk},\ \text{k=2, k=3} \tag{33}$$

* f is the summation of the positive operator weights

$$\text{mg=Max}\{|(\text{G2})|,\ |(\text{G3})|\} \tag{34}$$

$$\text{f1}(\text{bg(x,y), mg(x,y)})=\text{mg(x, y)}\times\alpha\,(\text{bg(x, y)})+\beta\,(\text{bg(x, y)}) \tag{35}$$

$$\text{f2}(\text{bg(x, y)}=\begin{cases} T0\times\left(1-\left(\frac{\text{bg(x,y)}}{127}\right)\right)^{1/2}+3 & \text{if bg(x, y)} \le 127 \\ \gamma\times(\text{bg(x, y)}-127)+3 & \text{if bg(x, y)} > 127 \end{cases} \tag{36}$$

$$\alpha\big(bg(x,y)\big)=bg(x,y)\times 0.0001+0.115 \tag{37}$$

$$\beta\,(\text{bg(x, y)}=\lambda\text{-bg(x, y)}\times 0.01 \tag{38}$$

5.2.3 Fourth level (rectangles) JND

Similarly, the algorithm at this level starts by calculating the average gray level using Equation (39).

$$\text{Bg(x,y)}=0.25\times\text{I(x, y-1)}+0.25\times\text{I(x , y+1)}+0.25\times\text{I(x+1, y)}+0.25\times\text{I(x-1, y)} \tag{39}$$

Four gradient operators, namely G1, G2, G3, and G4, have been used to capture the spatial activity directions as shown in Figure 25.

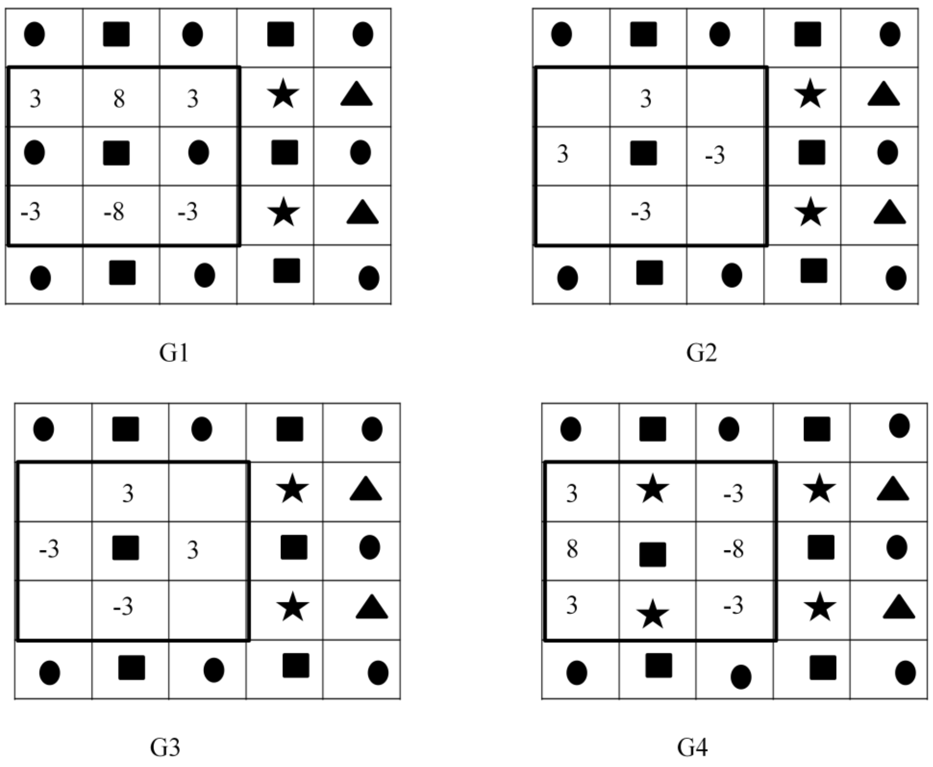


Figure 25 the gradient operators-Level4 (rectangles)

The mathematical formulations are the same with considering different gradient operators directions:

$$\mathrm{Gk}=1/\mathrm{f}\times\sum_{i=-1}^{i=1}\sum_{j=-1}^{j=1}\mathrm{I}(\mathrm{x}-\mathrm{i},\mathrm{y}-\mathrm{j})\times\mathrm{Gk},\ \ \mathrm{k}=1,2,3,4 \tag{40}$$

* f is the summation of the positive weights

$$\mathrm{mg}=\mathrm{Max}\{|(\mathrm{G1})|,\ |(\mathrm{G2})|,|(\mathrm{G3})|,|(\mathrm{G4})|\} \tag{41}$$

$$\mathrm{f1}(\mathrm{bg}(x,y),\mathrm{mg}(x,y))=\mathrm{mg}(x,y)\times\alpha\,(\mathrm{bg}(x,y))+\beta\,(\mathrm{bg}(x,y)) \tag{42}$$

$$\mathrm{f2}(\mathrm{bg}(x,y))=\begin{cases}T0\times\left(1-\left(\frac{\mathrm{bg}(x,y)}{127}\right)\right)^{1/2}+3 & \text{if } \mathrm{bg}(x,y)\le 127\\ \gamma\times(\mathrm{bg}(x,y)-127)+3 & \text{if } \mathrm{bg}(x,y)>127\end{cases} \tag{43}$$

$$\alpha\big(bg(x,y)\big)=bg(x,y)\times 0.0001+0.115 \tag{44}$$

$$\beta\,(\mathrm{bg}(x,y)=\lambda\text{-}\mathrm{bg}(x,y)\ \times\ 0.01 \tag{45}$$

5.3 Summary

The JND is one of the known perceptual or visually near lossless compression models. The aim is to have an adaptive quantization step size based on HSV and user perception of the noise. In JND, there are two factors that affect the visibility error threshold: 1) The spatial texture mask 2) The average background luminance.

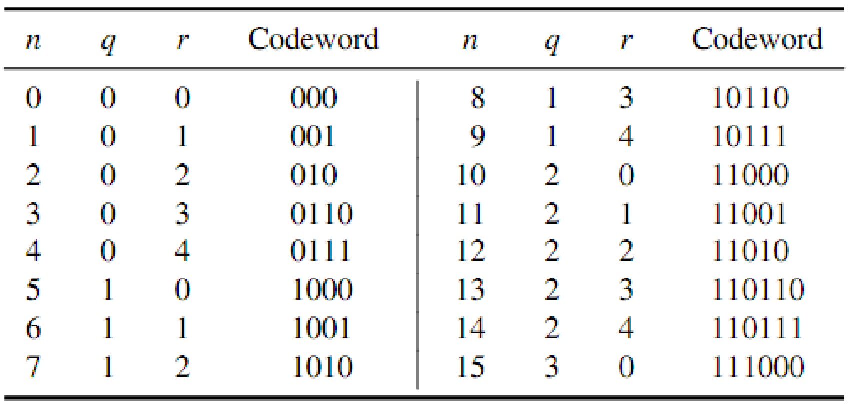
Considering the framework used in this work, i.e., HINT, slight modifications have been introduced to the JND original profile to fit with the structure of the algorithm at each level. In particular, the medications should fit with the availability of symbols, i.e., pixels at each level and accordingly the gradient operators have been changed as shown in the Chapter. The first level used the JPEG-LS [1] near lossless algorithm with varying the NEAR parameter. In the second and the third level the original JND has been applied with using only two modified gradient operators. Finally, the last level (rectangles) has used the four gradients operators but with modifications already justified.

6 Multi-level Golomb coding schemes

6.1 Golomb coder

The high construction cost of Huffman code and the large computation resources needed if adaptive coding algorithm is used have made Golomb coder an interesting alternative. Golomb coder was originally proposed in [22]. It is a powerful realization of the run-length coding and considered nearly optimal for coding of geometrically distributed non-negative integers. The integer n is represented in terms of quotient q=$\left\lfloor \frac{n}{m} \right\rfloor$, i.e., prefix and reminder (r) r=n-q×m, i.e., suffix. For simplicity the divisor (m) is chosen to be a power of two 2k and hence the Golomb coder is parameterized by (k). The quotient is unary represented (e.g. an integer 3 is represented by 1110) and the remainder is given by a binary representation using $\lfloor \log m \rfloor$ bits. Consider the following example when m=5, the code word of Golomb will be as shown in Table 2.

Table 2 Golomb code for m=5 [12]



n	*q*	*r*	Codeword	*n*	*q*	*r*	Codeword
0	0	0	000	8	1	3	10110
1	0	1	001	9	1	4	10111
2	0	2	010	10	2	0	11000
3	0	3	0110	11	2	1	11001
4	0	4	0111	12	2	2	11010
5	1	0	1000	13	2	3	110110
6	1	1	1001	14	2	4	110111
7	1	2	1010	15	3	0	111000

In general, when the image is being encoded, the input symbol is the mapped residual e- and accordingly the code length is:

$$\text{Length}=\left\lfloor \frac{e^{-}}{2^{k}} \right\rfloor+k+1 \tag{46}$$

The effectiveness of using Golomb-Rice codes is the calculation of the parameter k for a given sample or a block of samples. In JPEG-LS [1], the parameter k is estimated on the fly for each error value using techniques discussed in the following sections.

In this work, three Golomb-coding schemes have been used: 1) JPEG-LS based method. 2) Multi-dimensional scheme using previous prediction error values. 3) Hybrid scheme between the previous two approaches.

6.2 JPEG-LS based coding scheme (Scheme A)

The first approach for encoding prediction error values in this work is based on the original JPGEG-LS Golomb coding approach [1]. In JPEG-LS, before encoding the residual value, a mapping procedure is applied as illustrated in Equations (47), (48).

$$M(e) = 2|e| - u(e) \tag{47}$$

$$\text{u(e)}=\begin{cases} 1 \; if \; e < 0 \\ 0 \; if \; e \geq 0 \end{cases} \tag{48}$$

Where 47 and 48 map the residual to its index as a sorted sequence in non-decreasing order of probability as shown in Table 3.

Table 3. Mapping the residual values

Prediction residual	Mapped residual
0	0
-1	1
1	2
-2	3
2	4

For each context in JPEG-LS (refer to Chapter 4), the accumulated sum of magnitudes of the residual is stored in a register, A. While the accumulation of the corrected prediction residual is stored in a register B. To cancel the bias, a context counter for each occurrence of a context is kept in a register, N. The following procedure illustrates how to select a code for the prediction residual et+1:
Compute the parameter k as shown in Equation (49)

$$\text{k} = min\{k' | 2^{k'} \, N(Q) > A(Q)\} \tag{49}$$

a) If k >0, the code $Ѓ_k$ is chosen. If k=0 and 2B>-N, code $Ѓ_0$ is chosen. Else, code $Ѓ''_0$ is chosen.

b) Referring to [1], and using the C code to compute k empirically, Equation (50) shows that:

$$\mathrm{for(\ k=0;\ (N(Q)<<k)<A;\ k{+}{+})} \tag{50}$$

6.3 Multi-level/dimensional Golomb coding (Scheme B)

In this approach, the Golomb parameter k is adapted based on a context which collected the image statistics from the previously encoded error values, i.e., from upper levels of the hierarchy. Because of the availability of the nearby pixels in all directions, the context is expected to have good prior knowledge about the currently processed pixel, hence higher probability will be assigned during the coding process, i.e., a shorter code length is achieved. Each level has its own adaption method. The first level (circles) -as discussed in Chapter 3- is entirely encoded using JPEG-LS, thus, the same JPEG-LS adaption rules apply. The rest of the levels adaption rules are illustrated in the following sections.

6.3.1 Second level (stars) adaption

As shown in Figure 25, this level could use the error values of the four nearby pixels to adapt the Golomb parameter by that context. However, the error values, at this level, my not collect sufficient statistics due to the low correlation between the pixel and its neighbors. Thus, an attempt has been done to exploit all of the available four pixels to achieve a 4D adaption rule. Equation (51) which is considered appropriate to geometric sources and works perfectly under the assumption of a Laplacian-like residual. This Equation is based on the statistical mean of the source to determine the parameter k. The mapping of residuals to non-negative integers obeys Equations (47, 48). Because the natural images are considered locally stationary in these four dimensions, i.e., corner pixels, the k value resulted from each of the four pixels would give the same compression performance [15].

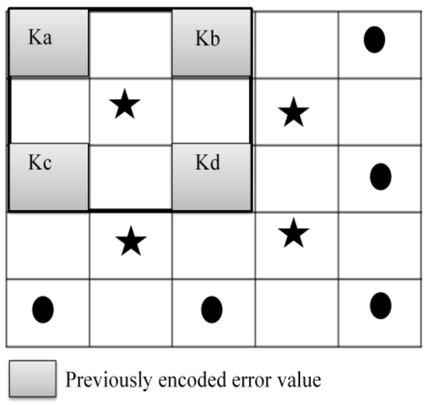


Figure 26 the second level (stars) adaption map

As a result, the average of the four (k) values is computed as shown in Equation (51)

$$ki = \max\left\{0, \left\lceil\frac{\log2(E(e))}{2}\right\rceil\right\} \tag{51}$$

Where E(e) is the expectation value of the mapped error prediction value. From Figure 26, the resulted K’s are: Ka, Kb, Kc, Kd. Using Equation (52), the average value is computed to determine K.

$$k = \left\lceil\frac{Ka+Kb+Kc+Kd}{4}\right\rceil \tag{52}$$

6.3.2 Third level (pyramids) adaption

At this level, also four nearby error values could be used (i.e., left, right, upper, lower) of the two last level (circles and stars). Figure 26 illustrates the used pixels.

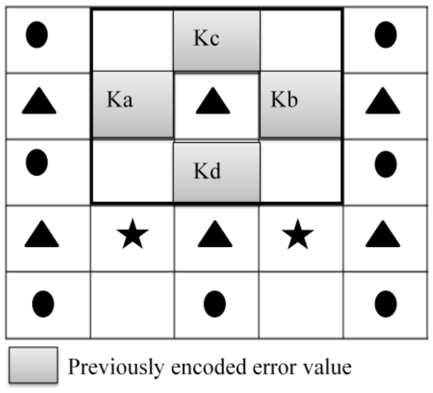


Figure 27 the third level (pyramids) adaption map

Apparently, the collected statistics from these nearby error values should capture higher dependency thanks to the higher correlation. The same k computation approach has been applied at this level as shown in Equation (53).

$$\mathrm{ki} = \max\left\{0, \left\lceil \frac{\log2(\mathrm{E(e)})}{2} \right\rceil\right\} \tag{53}$$

Where E(e) again is the expectation value of the mapped error prediction value. From Figure 27, the resulted ki values are: ka, kb, kc, kd. Using Equation (54), the average value is computed to determine k.

$$\mathrm{k} = \left\lceil \frac{Ka+Kb+Kc+Kd}{4} \right\rceil \tag{54}$$

6.3.3 Fourth level (rectangles) adaption

As discussed in Chapter 3, this level contains ¾ of the original image pixel values. Thus, the parameter k adaption could exploit the available nearby error values from all previous levels (i.e., circles, stars and pyramids). The higher order statistics is typically more achieved at this level. Therefore, the eight nearby error values have been used to form an 8D adaption method. Figure 28 shows the used error values in the adaption process.

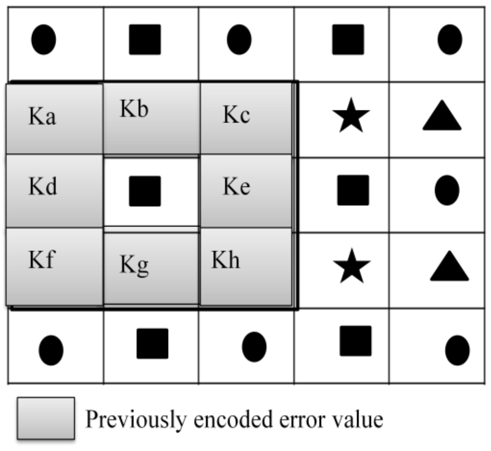


Figure 28 the fourth level (rectangles) adaption map

Likewise, the same k computation approach has been applied at this level as shown in Equation (55).

$$\mathrm{ki} = \max\left\{0, \left\lceil \frac{\log 2(\mathrm{E(e)})}{2} \right\rceil\right\} \tag{55}$$

Where E(e) again is the expectation value of the mapped error prediction value. From Figure 27, the resulted k's are: ka, kb, kc, kd, ke, kf and kh. Using Equation (52), the average value is computed to determine k.

$$\mathrm{k} = \left\lceil \frac{\mathrm{Ka+Kb+Kc+Kd+Kb+Kc+Kd+Ke+Kf+Kg+Kh}}{8} \right\rceil \tag{56}$$

6.4 The hybrid scheme

Scheme A has achieved a gain about 1.5% in compression performance over scheme B, in general. While in some pictures (e.g., Cameraman, Mandrill), the scheme B has shown better compression performance. Unfortunately, the used error values from previous levels have not collected sufficient statistics to achieve much gain in compression in comparison with scheme A in general. However, the good compression performance of scheme B on some images (e.g., cameraman) has led to another adaption approach, namely the hybrid approach. The hybrid approach aims to encompasses the advantages of both approaches by assigning scores to each k determined from the above two schemes, (i.e., scheme A, scheme B) as given by Equation (57).

$$\mathrm{Kavg} = \lceil 0.75 \times \mathrm{schemeA} + 0.25 \times \mathrm{schemeB} \rceil \tag{57}$$

The scores assigned have been empirically found to give the best compression results.

This approach has shown relatively similar compression performance with (scheme A)

but suffers from higher execution time in comparison with (scheme A) and (scheme B).

The performance and the time analysis will be discussed in Chapter 7.

6.5 Summary

The last step in LOCMIC algorithm is coding the residuals, i.e., assigning a codeword. Golomb coder has been chosen because it can achieve an optimal variable length coding for the geometric-like sources [24]. Moreover, it is well suited to the adaptive coding since only one parameter, i.e., k, needs to be adapted. The higher complexity of the Arithmetic and Huffman coding has made Golomb an interesting alternative.

In this work, three different coding approaches based on Golomb coder have been presented: 1) JPEG-LS based method. 2) Multi-dimensional scheme using previous prediction error values. 3) Hybrid scheme between the previous two approaches.

The first approach has been found to give the best compression performance among the three schemes.

7 Experimental results

7.1 Lossless LOCMIC comparative evaluation

To evaluate the proposed algorithm compression performance in terms of bit rate, a comparison with other representative lossless image compression algorithms in the literature has been made. The evaluation is based on a known 8-bit gray-scale image set. Table 4 shows this compression results in bits per pixel (bpp). The selected algorithms are: 1) LOCO-I: A low complexity, context-based, lossless image compression algorithm. 2) JPEG2000. 3) JPEG-LS. 4) SPIHT: A fast and efficient image codec based on **S**et **P**artitioning **I**n **H**ierarchical **T**rees. 5) CALIC: Context-based, adaptive, lossless image coding. 6) KLOCMIC: "K" tuned average LOCMIC. 7) CLOCMIC: Comprehensive LOCMIC. The proposed algorithms results have been achieved using Scheme A in coding the images (refer to chapter 5). The compression performance results of schemes 1-5 are from [24, 25].

Table 4. Comparative compression performance evaluation (bpp)

Image	LOCO-I	JPEG2000	JPEG-LS	SPIHT	CALIC	KLOCMIC	CLOCMIC
Lena	4.25	4.30	4.24	4.17	4.05	4.27	4.26
Goldhill	4.73	4.60	4.47	4.75	4.67	4.60	4.60
Balloon	2.90	3.0	2.90	2.98	2.78	3.00	2.98
Airplane	4.59	4.01	3.80	4.50	4.85	3.98	3.97
Peppers	4.50	4.62	4.51	4.54	4.47	4.56	4.56
Mean	**4.194**	**4.106**	**3.984**	**4.188**	**4.164**	**4.083**	**4.073**

As shown in Table 3, the two proposed algorithms KLOCMIC and CLOCMIC have achieved gain about ***3%*** in compression performance over LOCO-I, about ***1%*** over JPEG2000, ***3%*** over SPIHT, and ***2%*** over CALIC.

7.2 Decorrelation evaluation

To evaluate the prediction efficiency of the listed proposed prediction schemes in Chapter 3, the first order entropy is used as a measurement. Table 5 shows the comparison of three different proposed algorithms with the Median Edge Detector (MED) fixed predictor. The proposed algorithms are: 1) MLOCMIC: Median value LOCMIC. 2) KLOCMIC: "K" tuned average LOCMIC. 3) CLOCMIC: the Comprehensive LOCMIC.

Table 5 First order entropy comparison (bpp) on known image set

Image	MED	MLOCMIC Proposed	KLOCMIC Proposed	CLOCMIC Proposed
Lena	4.90	4.75	4.75	4.75
Lennagrey	4.56	4.41	4.41	4.40
Balloon	3.12	3.06	3.08	3.06
Mandrill	6.28	6.30	6.30	6.30
Peppers	4.95	4.64	4.65	4.64
Noisesequare	5.73	5.56	5.55	5.55
Mean	**4.92**	**4.79**	**4.79**	**4.78**

The comparison has also been with the conventional HINT [18] on another known image set. To see the decorrelation efficiency of the proposed prediction model over the original HINT, look at Table 6.

Table 6 First order entropy comparison (bpp) with multi-resolution schemes

Image	Conventional HINT [31]	CLOCMIC Proposed
Barb	5.41	5.26
Barb2	5.47	5.26
Balloon	3.29	3.06
Gold	4.90	4.74
Hotel	5.04	4.74
Mean	**4.82**	**4.61**

As shown in Table 5, 6, the compression performance gain of the proposed algorithms over MED is about ***2.8% and*** about ***4.5%*** over the conventional HINT.

7.3 The context model effectiveness

To see the effectiveness of the context model (discussed in chapter 4) which has been used to correct the prediction, the following approach has been applied: two levels are individually taken to measure the prediction correction effect on some kwon image set. The third level (pyramids) and the last level (rectangles) residuals, before and after the correction, have been measured by the first order entropy. Figure 29 illustrates the third level correction efficiency.

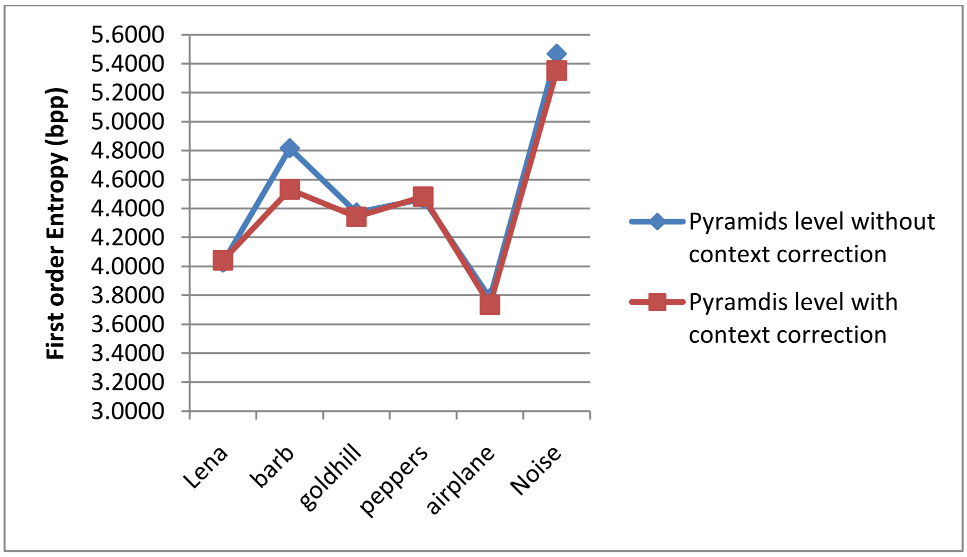


Figure 29 the third level (pyramids) context based correction effect

The algorithm that has been used in this experiment is the KLOCMIC. Figure 30 illustrates the fourth level (rectangles) correction effect based on the first order entropy in (bpp).

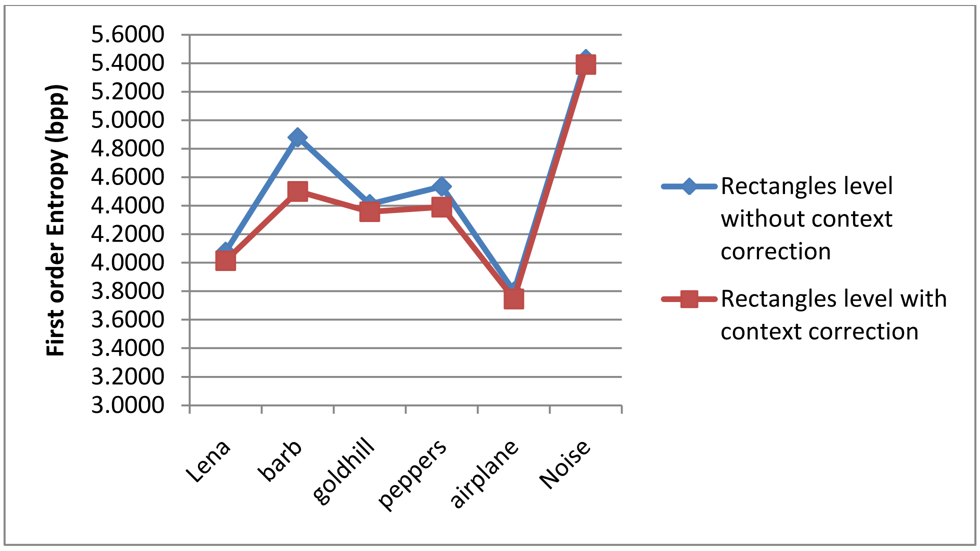


Figure 30 . The fourth level (rectangles) context based correction effect

The enhancement achieved after the prediction correction is about ***1.66%*** in terms of the first order entropy (bpp) at the third level (pyramids) and ***2.73%*** at the fourth level (rectangles). The overall efficiency is about ***2.2%.***

7.4 The coding schemes evaluation

As discussed in Chapter 6, three different coding schemes have been presented in this work. They are: 1) JPEG-LS based method (scheme A). 2) Multi-dimensional scheme (scheme B) using previous prediction error values. 3) Hybrid scheme between the previous two approaches. Table 7 shows a comparison between these three different schemes on the same image set and based on the bit rate (bpp)

Table 7 the coding schemes comparison (bpp)

Image	**KLOCMIC Scheme A**	**KLOCMIC Scheme B**	**KLOCMIC Hybrid**
Lena	4.27	4.29	4.26
Goldhill	4.60	4.66	4.60
Balloon	3.00	3.06	3.01
Airplane	3.98	4.04	3.99
Peppers	4.56	4.64	4.54
Mean	**4.082**	**4.138**	**4.08**

As shown in Table 7, the compression performance of both Scheme A and the hybrid scheme is similar. However, the hybrid scheme suffers from higher execution time among the three coding schemes presented in this work.

7.5 The perceptual LOCMIC evaluation

The simple and the widely used quality metrics are the Mean Squared Error (MSE) and the Peak Signal to Noise Ratio (PSNR) because they are appealing and easy to compute. But they are not very well matched to assess the perceived visual quality. Thus, an interesting alternative has been proposed in [27] which is the Structural SIMilarity (SSIM). The SSIM can be viewed as a quality measure of one of the images being reconstructed, provided the other image is regarded as the original. The SSIM consists of three functions, each of which approximates a certain aspect of the pixels. More details about the implementation point of view can be found in [28]. Table 8 shows the compression performance of the perceptual LOCMIC with the SSIM metric to evaluate the reconstructed image quality.

Table 8. The compression, quality performance of the perceptual LOCMIC

Image	Perceptual LOCMIC	SSIM index
Lena	2.04	0.993
Airplane	1.91	0.996
Goldhill	2.19	0.992
couple	1.73	0.973
Camera	2.30	0.981
Barb	2.41	0.993
Peppers	2.17	0.992

The perfect value of the SSIM is 1 and achieved when the reconstructed image does not differ from the original image.

To evaluate the compression performance with the near lossless JPEG-LS, the NEAR value is set to 2. Table 9 shows the comparison based on bit rates (bpp)

Table 9. A compression, near lossless JPEG-LS and the perceptual LOCMIC

Image	JPEG-LS , NEAR=2	Perceptual LOCMIC
Lena	2.38	2.04
Airplane	1.84	1.91
Goldhill	2.33	2.19
couple	1.83	1.73
Camera	2.28	2.30
Barb	2.53	2.41
Peppers	2.29	2.17
Mean	**2.21**	**2.11**

As shown in Table 9. The perceptual LOCMIC has gained about ***4.7%*** in compression performance in comparison with the near-lossless JPEG-LS at NEAR value of 2. However, to have a fair comparison in terms of reconstructed image quality based on the SSIM metric, Figure 31 shows that comparison between the perceptual LOCMIC and JPEG-LS at nearly the same bit rate.

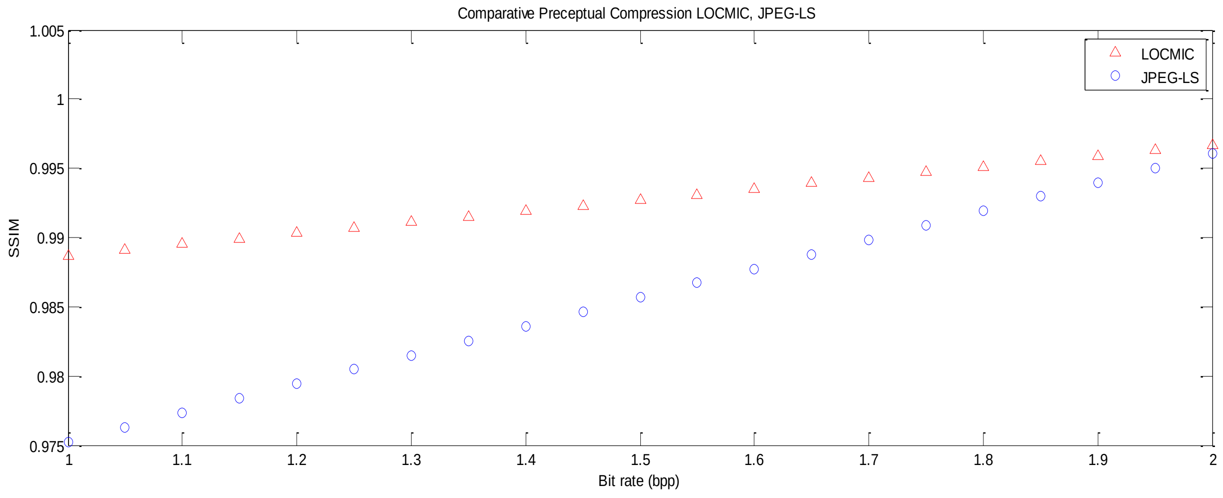


Figure 31 Comparative (quality-compression) performance between near lossless JPEG-LS and perceptual LOCMIC

To have a look on the reconstructed picures, Figure 32 and Figure 33 show the orignal and the reconstructed mandrill and peppers respecicvely after applying the perceputal LOCMIC. Apperantly, the reconstricted pictures have the same percepual quality as the orignal, especially at this resolution shown in Figure 32. 33.

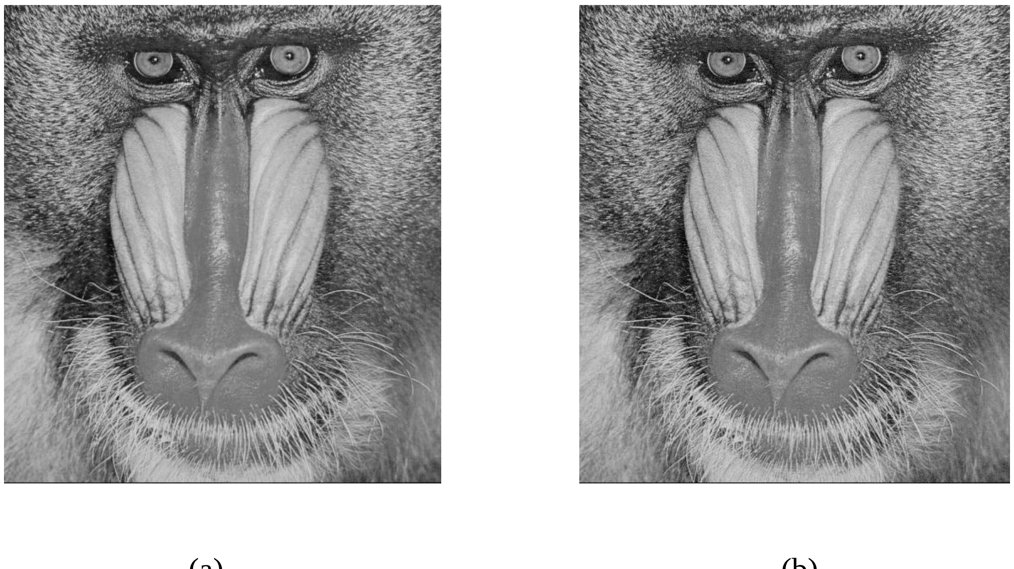


Figure 32 a) Original mandrill. b) Reconstructed mandrill with SSIM=0.9955 and bit rate of 3.46 bpp

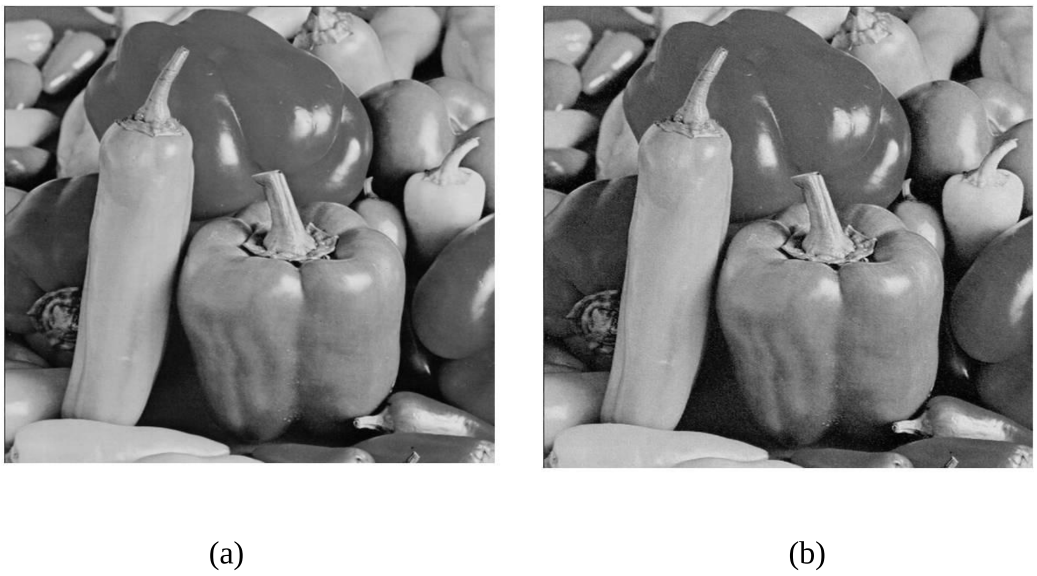
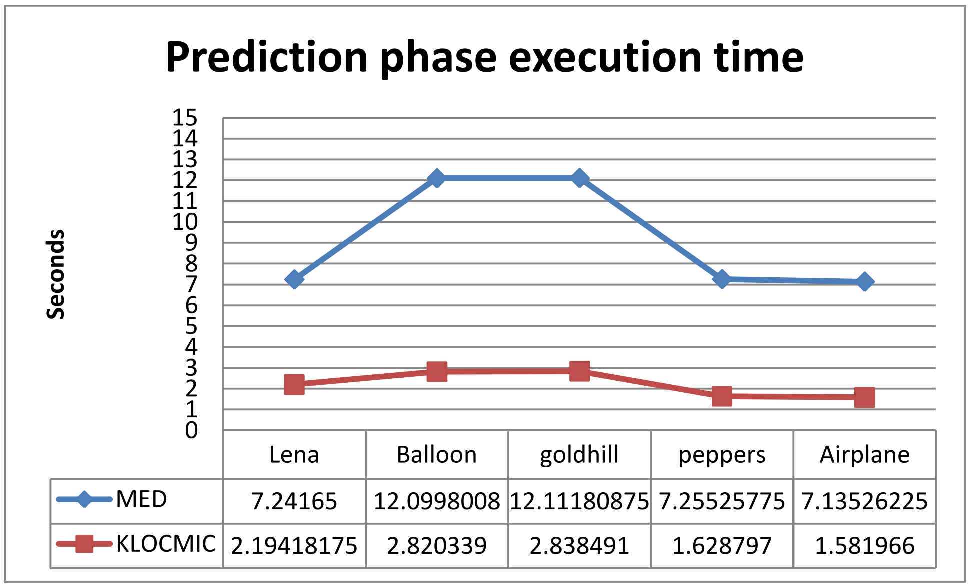


Figure 33 a) Original peppers. b) Reconstructed peppers with SSIM=0.9923 and bit rate of 2.18 bpp

7.6 Prediction time analysis

In recent years, the algorithm complexity has been a main concern in many embedded systems (e.g. digital cameras). With the rapid growth in photography and imaging techniques, the need for storing higher resolution images with huge file sizes has immerged. Hence, developing high performance image compression schemes with low complexity, e.g., short execution time, has become a challenge. To measure the execution time, the MATLAB offers special command for that which is “tic, toc”.

The experiments have been conducted on a core 2 duo, 2.00 GHz DELL personal computer, with 2 GB installed RAM and 32 bit window vista. The experiments followed some rules: 1) the prediction phase is the target of the evaluation. 2) The algorithm is executed six successive times on the same picture. 3) The average of the six-time executions on one image is calculated. Figure 32 shows an execution time comparison between the MED predictor and the KLOCMIC predictor in seconds.



	Lena	Balloon	goldhill	peppers	Airplane
◆ MED	7.24165	12.0998008	12.11180875	7.25525775	7.13526225
■ KLOCMIC	2.19418175	2.820339	2.838491	1.628797	1.581966

Figure 34 Prediction execution time comparative evaluation (seconds)

The Figure above shows that KLOCMIC predictor has achieved an excellent gain over MED predictor in about ***75%.***

The KLOCMIC predictor is based on very simple computations (i.e., multiplication, addition, shifting) and does not have a switching formula to detect edges in different directions which in turn may add to the computation time.

8 Conclusions and future work

8.1 General remarks and conclusions

In this work, the multi-resolution image compression scheme has been found to be an interesting alternative of the conventional image compression schemes in some applications where the low complexity and the high compression performance are needed.

8.1.1 Remarks on the LOCMIC prediction model

In this work, different prediction schemes have been presented. The KLOCMIC, MLOCMIC and CLOCMIC have shown the best compression performance among other implemented methods. Moreover, the execution time of the proposed schemes has been minimized thanks to the simple formulas used in predicting the pixel values. KLOCMIC and MLOCMIC have not used any edge detection formula like MED. However, the powerful technique of clamping the average by the median value has shown a good prediction accuracy over known prediction algorithms, e.g., MED, MLP. Furthermore, The HINT framework has been found to be a flexible framework with which one can use non-causal pixel values in the prediction to achieve better inspection on the predicted value. The CLOCMIC has encompassed the advantage of detecting the edge direction like MED and the elegant formulas used in KLOCMIC and MLOCMIC.

8.1.2 Remarks on the perceptual LOCMIC

In chapter 5, a superior integration of the HINT framework with the JND scheme to achieve a perceptual, i.e., visual coding has been proposed. Empirically, the use of a 3×3 window size instead of a 5×5 window size as implemented in the original JND and applied in [15, 20] to calculate the gradient operators, has approximately given the same picture quality using SSIM metric, but with better compression performance in about 2.5%. The interpretation is that on the HINT framework, the area of active pixels has been empirically analyzed to be within the 3×3 window centered at the middle pixel, in almost of the nature images.

8.1.3 Remarks on the coding schemes

In this work, three different approaches to the Golomb parameter K adaption have been presented. Experimentally, using (scheme A) which is the JPEG-LS based method has been found to give the best compression performance among the three approaches (about 1.2% better than scheme B). On the other hand, (scheme B) which is based on multi-dimensional adaption rules, performed well in terms of execution time (about 2% faster than (scheme A). Finally, the hybrid scheme has achieved similar compression performance as (scheme A), but suffered from slower execution time among the three approaches. The reason is that the hybrid scheme uses vast of computation operations (e.g. Expectation value calculation, more than ten averaging operations).

8.2 Future work

The future work will be investigating the potential of a parallel LOCMIC version. The idea is as shown in Figure 35 and Figure 36. At the second level (stars) the interpolation could be done in parallel with the third level (pyramids) using two different threads. This may lead to potentially minimized throughput. However, the main drawback of this approach is that fewer pixels can be exploited in the interpolation at one of the threads. In Figure 35, particularly in Thread 1, the second level will be forced to use only two nearby pixels, namely the directly above and below pixels. Hence, the prediction accuracy might become worse in some images. In Thread 2, the normal interpolation as discussed in chapter 3 takes place, i.e., the stars pixels are interpolated using the four corner (circle) pixels. The total number of levels will become three. Specifically: 1) The circles 2) The stars and pyramids as one level. 3) The rectangles level. Figure 35 illustrates the parallelization model at the second level and the interpolation order in each thread.

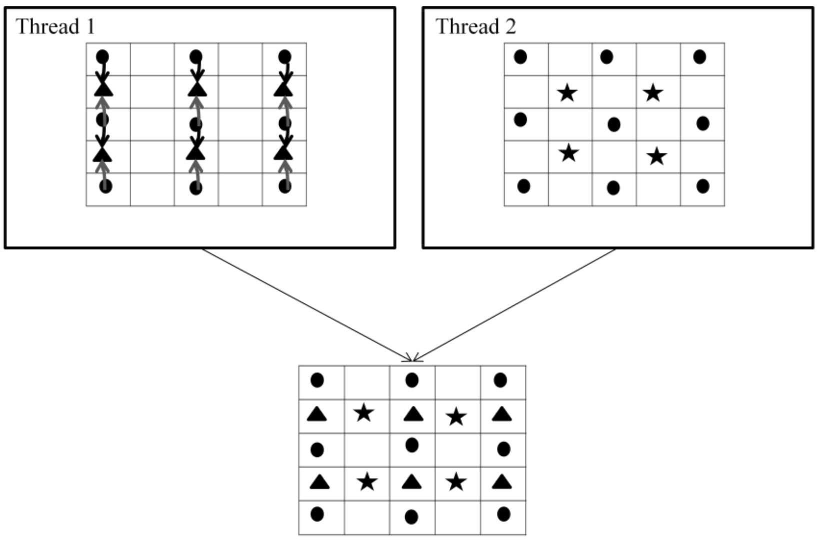


Figure 35 Parallelization model at the second level

The other possibility is to apply the parallelization at the third level where pyramid pixels and rectangle pixels are interpolated simultaneously in Thread 1, 2 respectively. The drawback of less nearby pixels availability is not presented well at this level because there would be four nearby pixels in each thread, as shown in Figure 36, to be exploited in interpolating the remaining pixels.

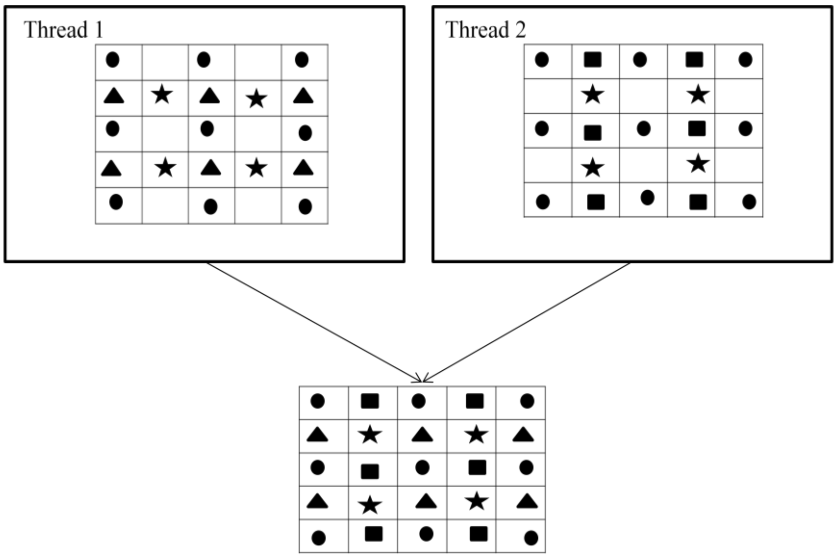


Figure 36 Parallelization model at the third level

The increasing availability of parallel computing architectures and the massive volume of scientific data being produced make the parallel data compression a good subject of research. Thus, the next research will be directed to a parallel version of LOCMIC.

REFRENCES

1. Weinberger, Marcelo J., Gadiel Seroussi, and Guillermo Sapiro. "The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS." *Image Processing, IEEE Transactions on* 9.8 (2000): 1309-1324.

2. Mrak, Marta, Detlev Marpe, and Thomas Wiegand. "A context modeling algorithm and its application in video compression." Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on. Vol. 3. IEEE, 2003.

3. Memon, Nasir, and Xiaolin Wu. "Recent developments in context-based predictive techniques for lossless image compression." *The Computer Journal*40.2 and 3 (1997): 127-136.

4. Wu, Xiaolin, and Nasir Memon. "CALIC-a context based adaptive lossless image codec." *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*. Vol. 4. IEEE, 1996.

5. Tao, Tao. *Compressed pattern matching for text and images*. Diss. University of Central Florida Orlando, Florida, 2005.

6. Silvia Ahmed, “Parallel hardware architecture for JPEG-LS based on domain decomposition using context sets”, Master thesis, Institut für Parallele und Verteilte Systeme, Universität Stuttgart, 2011.

7. Hontsch, Ingo, and Lina J. Karam. "Adaptive image coding with perceptual distortion control." *Image Processing, IEEE Transactions on* 11.3 (2002): 213-222.

8. Ghouti, Lahouari, and Ahmed Bouridane. "A just-noticeable distortion (jnd) profile for balanced multiwavelets." *European Signal Processing Conference,(EUSIPCO 2005), Antalya, Turkey*. 2005.

9. Yap, Vooi Voon. *Wavelet-based image compression for mobile applications*. Diss. Middlesex University, 2005.

10. Rabiee, Hamid R., R. L. Kashyap, and H. Radha. "Multiresolution image compression with BSP trees and multilevel BTC." *Image Processing, 1995. Proceedings., International Conference on*. Vol. 3. IEEE, 1995.

11. Kattan, Ahmed. *Evolutionary Synthesis of Lossless Compression Algorithms: The GP-zip Family*. Diss. University of Essex, 2010.

12. Salomon, David. *Data compression: the complete reference*. Springer-Verlag New York Incorporated, 2004.

13. Howard, Paul G., and Jeffrey Scott Vitter. "Fast and efficient lossless image compression." *Data Compression Conference, 1993. DCC'93.*. IEEE, 1993.

14. Howard, Paul G., and Jeffrey S. Vitter. "Fast progressive lossless image compression." *IS&T/SPIE 1994 International Symposium on Electronic Imaging: Science and Technology*. International Society for Optics and Photonics, 1994.

15. Gera, Yurij, et al. "Fast and Context-Free Lossless Image Compression Algorithm Based on JPEG-LS." *Data Compression Conference (DCC), 2012*. IEEE, 2012.

16. Howard, Paul G., and Jeffrey Scott Vitter. "New methods for lossless image compression using arithmetic coding." *Information processing & management*28.6 (1992): 765-779.

17. Burt, Peter, and Edward Adelson. "The Laplacian pyramid as a compact image code." *Communications, IEEE Transactions on* 31.4 (1983): 532-540.

18. Viergever, Max A., and Paul Roos. "Hierarchical interpolation." *Engineering in Medicine and Biology Magazine, IEEE* 12.1 (1993): 48-55.

19. Removing Grain filters, URL: http://www.removegrain.de.tf

20. Bandekar, Namrata, Anant Malewar, and Vikram M. Gadre. "A Perceptually Tuned Model for Just Noticeable Distortion Estimation for Videos." *IETE Journal of Research* 57.2 (2011): 125.

21. Kai Liu, “A Just-Noticeable-Distortion Based Perceptually Lossless Image Compression Codec”, Master thesis, Institut für Parallele und Verteilte Systeme, Universität Stuttgart, 2012.

22. Golomb, Solomon W., Robert E. Peile, and Robert A. Scholtz. *Basic concepts in information theory and coding: the adventures of secret agent 00111*. Springer, 1994.

23. Schalkoff, Robert J. *Digital image processing and computer vision*. New York: Wiley, 1989.

24. Taubman, David S., Michael W. Marcellin, and Majid Rabbani. "JPEG2000: Image compression fundamentals, standards and practice." *Journal of Electronic Imaging* 11.2 (2002): 286-287.

25. Bekkouche, Hocine, and Michel Barret. "Adaptive multiresolution decomposition: application to lossless image compression." *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*. Vol. 4. IEEE, 2002.

26. Hsieh, F., and K. Fan. "A High Performance Lossless Image Coder." *IPPR Conf. on Computer Vision & Graphic Image Processing*. 2005.

27. Wang, Zhou, et al. "Image quality assessment: From error visibility to structural similarity." *Image Processing, IEEE Transactions on* 13.4 (2004): 600-612.

28. The SSIM Index for Image Quality Assessment, URL: http://www.cns.nyu.edu/~lcv/ssim

29. SHVETS, IGOR. "Non-contact high resolution microwave scanning measurement technology." (2003).

30. Ludwig, Sönke. Implementation of a spatio-temporal Laplacian image pyramid on the GPU. Diss. Ph. D. dissertation, Universität zu Lübeck, Feburary, 2008.

31. Wu, Xiaolin. "Lossless compression of continuous-tone images via context selection, quantization, and modeling." Image Processing, IEEE Transactions on 6.5 (1997): 656-664.

32. LIM LIP YEOW, "A THEORETICAL LOOK AT PIXEL ORDERING." Thesis, NATIONAL UNIVERSITY OF SINGAPORE, 1999.

Declaration

All the work contained within this thesis, except where otherwise acknowledged, was solely the effort of the author. At no stage was any collaboration entered into with any other party.

Stuttgart, 15. November 2012 ______________________