Edge Adaptive Hierarchical Interpolation for Lossless and Progressive Image Transmission

Yenewondim Biadgie[1], Young Chul Wee[1] and Jung-Ju Choi[2]
[1] Department of Computer Science and Engineering, Ajou University
Suwon, 443-749 – Rep. of Korea
[e-mail: wondim2005@yahoo.com, ycwee@ajou.ac.kr]
[2] Department of Digital Media, Ajou University
Suwon, 443-749 – Rep. of Korea
[e-mail: jungju@ajou.ac.kr]
*Corresponding author: Jung-Ju Choi

Received May 25, 2011; revised October 2, 2011; accepted October 26, 2011; published November 29, 2011

Abstract

Based on the quincunx sub-sampling grid, the New Interleaved Hierarchical INTerpolation (NIHINT) method is recognized as a superior pyramid data structure for the lossless and progressive coding of natural images. In this paper, we propose a new image interpolation algorithm, Edge Adaptive Hierarchical INTerpolation (EAHINT), for a further reduction in the entropy of interpolation errors. We compute the local variance of the causal context to model the strength of a local edge around a target pixel and then apply three statistical decision rules to classify the local edge into a strong edge, a weak edge, or a medium edge. According to these local edge types, we apply an interpolation method to the target pixel using a one-directional interpolator for a strong edge, a multi-directional adaptive weighting interpolator for a medium edge, or a non-directional static weighting linear interpolator for a weak edge. Experimental results show that the proposed algorithm achieves a better compression bit rate than the NIHINT method for lossless image coding. It is shown that the compression bit rate is much better for images that are rich in directional edges and textures. Our algorithm also shows better rate-distortion performance and visual quality for progressive image transmission.

Keywords: Pyramid data structures, image interpolation, lossless image compression, progressive image transmission

This work was partially supported by Defense Acquisition Program Administration and Agency for Defense Development under the contract. (UD060048AD)

DOI: 10.3837/tiis.2011.11.011

1. Introduction

When a large digital image is transmitted across a low-bandwidth connection as a sequence of rows of the image from top to bottom, users can recognize the image after a significant portion of the image is transmitted. There is a long time delay before users can visually inspect the image. Image compression is a promising technique for reducing the transmission time. However, for lossless image applications such as medical imaging, the compression ratio is usually very low; thus, the transmission time remains long for lossless image applications [1]. Progressive image transmission can also alleviate a long transmission time in some applications [2][3][4]. In progressive image transmission schemes, an approximate version of the original image is transmitted, after which the quality of the image is progressively improved over a number of successive transmissions. The progressive image transmission technique reduce the transmission time by terminating the successive transmissions at any time when the intermediate quality is satisfactory but not a higher quality is required for applications such as interactive image searches, World Wide Web browsing, image broadcasting, or remote surveillance.

Based on the way that the image is progressively updated, there are two types of progressive image transmission: progressive fidelity transmission and progressive resolution transmission. Progressive fidelity transmission is achieved by refining the precision of an image at each transmission stage. Initially, the low-precision form of an image is sent. In subsequent stages, the precision is gradually increased until the image reaches the full precision level. For example, the bit-plane coding of an image transmits each binary digit of each pixel from the most significant bit to the least significant bit. This implies that each pixel value is refined progressively but that the reconstructed image is inherently the same size as the original image. Hence, an image can be represented by a single-resolution coding technique in progressive fidelity transmission. In progressive resolution transmission, an image with reduced resolution is transmitted first and displayed at a small size. The information to obtain a high-resolution image from a low-resolution image is transmitted progressively. Therefore, an image being processed by means of progressive resolution transmission is usually represented by a multi-resolution coding technique. Typically, this technique is based on a pyramid data structure.

A number of pyramid data structures have been evaluated for multi-resolution lossless image coding and progressive image transmission. The Hierarchical INTerpolation (HINT) technique [5] showed superior performance compared to other basic pyramid data structures such as a mean pyramid or a difference pyramid. The Interleaved Hierarchical INTerpolation (IHINT) [6] and the New Interleaved Hierarchical INTerpolation (NIHINT) [7] methods were also developed as improvements of HINT to reduce the entropy of interpolation errors further. However, HINT, IHINT, and NIHINT as interpolative pyramid algorithms do not use techniques to identify the directions of strong intensity correlations near the target pixel so as to exploit the directional correlations among neighboring pixels. Moreover, the finite impulse response (FIR) interpolation filters of these pyramid algorithms are not adaptive to local structures around edges. As a result, interpolation errors along sharp edges are large and cannot be encoded efficiently by entropy coders. These interpolative algorithms also tend to generate interpolated images with blurred edges and annoying artifacts near the edges.

In this paper, we propose a new image interpolation algorithm, Edge Adaptive Hierarchical INTerpolation (EAHINT), to reduce the entropy in interpolation errors by exploiting the

directional correlations around a target pixel. To model the strength of the local edge, we use the local variance of the causal context of the target pixel as a measure of the local activity level. Based on the local variance, three statistical decision rules are designed to classify the local edge into a strong, a weak, or a medium edge. If a local area with a strong edge is detected, the minimum gradient direction is selected from the horizontal, vertical, and diagonal directions and the target pixel is then interpolated using the neighboring pixels along the selected direction. If a local area with a medium edge is detected, multi-directional adaptive weighted interpolation is performed. Otherwise, for a local area with a weak edge, we apply non-directional interpolation with static weights. Hence, according to the type of a local edge, the proposed algorithm switches among its three interpolators. We also present experimental results to compare the compression bit rate, rate-distortion performance, and visual quality with respect to NIHINT, which is known to be a superior interpolative pyramid algorithm for lossless and progressive image coding.

The rest of the paper is organized as follows. Section 2 reviews existing image interpolation algorithms as related work. Section 3 gives a general overview of the proposed algorithm and a detailed description of three interpolation methods. The experimental results are then presented in Section 4, which is followed by the conclusion in Section 5.

2. Related Work

Numerous studies related to image interpolation algorithms have been proposed over the last two decades. Image interpolation algorithms are typically used for image up-scaling, where the missing pixels of an up-scaled image are traditionally computed from simple interpolating polynomials such as pixel duplication, nearest neighbor, bilinear or cubic interpolation, or kernel-based methods. Since the coefficients of the interpolating polynomial are fixed once and remain the same throughout the image, they cannot capture the rapidly evolving statistics around the edges. As a result, they often produce an up-scaled image with blurred edges and jagged contours around the edges. Recently, several spatially adaptive interpolation techniques have been proposed to find the optimal coefficients by means of least-squares optimization [8][9][10][11][12][13]. They show the superior quality compared to other types of image interpolation techniques. Although they do not assume the strength or direction of a local edge, they are effective for an arbitrary edge direction. However, finding the optimal coefficients is computationally very expensive because the least-squares covariance matrix for each pixel is computed from a large window. Hence, their use is limited in progressive transmission applications that require fast encoding. There are also several fast image up-scaling techniques that utilize smoothing filters, where the low-resolution image is a down-sampled image of the smoothed high-resolution image. These techniques utilize a bilateral filter [14], a Gaussian point spread function kernel [15], a 2x2 mean filter [16], and the Lagrange filter [17]. Because the filtered image is not identical to the original image, they are not suitable for lossless image compression.

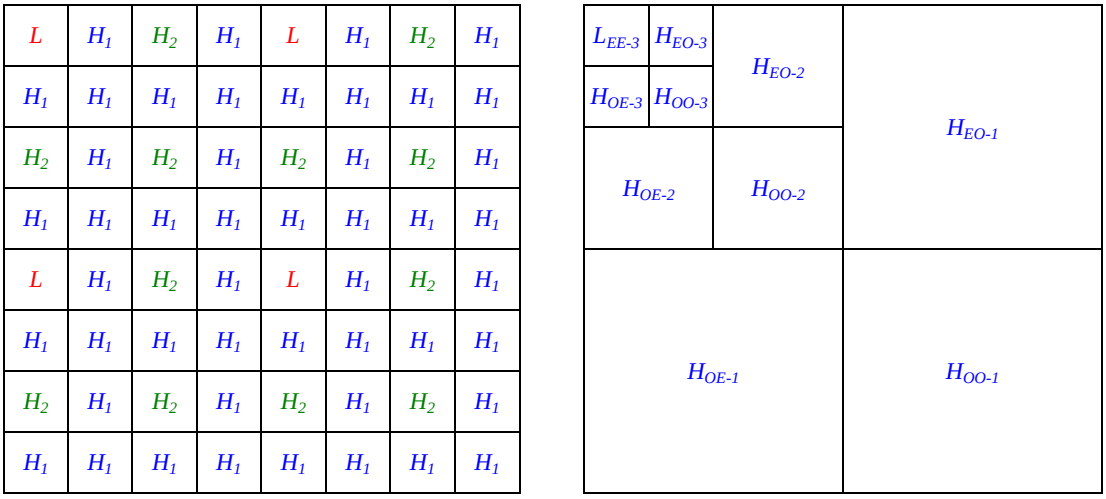
In this section, we focus on image interpolation algorithms for lossless and progressive transmission applications, specifically HINT [5], IHINT [6] and NIHINT [7]. We also address the weaknesses of these algorithms. These interpolative algorithms as well as our algorithm use the same down-sampling technique.

Let $I_o(r, c)$, $r = 0, 1, 2, 3, \ldots, R\text{-}1$, $c = 0, 1, 2, 3, \ldots, C\text{-}1$, $R = p \times 2^K$, and $C = q \times 2^K$ be the original input image, where p, q, and K are positive integers. The set of reduced-resolution images can be obtained by sub-sampling the original image such that

$$I_l(i, j) = I_{l-1}(2i, 2j) = I_o(i \times 2^l, j \times 2^l), \quad (1)$$

for l =1, 2, 3, ..., K, i = 0, 1, 2, 3, ..., (R/2^l)-1, and j = 0, 1, 2, 3, ..., (C/2^l)-1. The set of down-sampled images I_l at level l is a multi-resolution pyramid image representation, and K represents the top-level (base-band level) image I_K with a size of $p \times q$. In this multi-resolution image representation, a higher-resolution image I_{l-1} of $2R \times 2C$ pixels is reduced to a lower-resolution image I_l of $R \times C$ pixels recursively such that $I_{l-1}(2i, 2j) = I_l(i, j)$. Hence, a coarse representation of an image is formed by simply removing pixels from the image and using the remaining pixels to form a lower resolution image. The removed pixels can be predicted by an interpolation method from the remaining pixels, after which the interpolation errors are entropy-coded. The pixels remaining after each level of down-sampling are represented as low-band (L-band) nodes while the prediction errors that replace the removed pixels are represented as high-band (H-band) nodes [1]. An example of this multi-resolution image representation scheme when K=2 is shown in **Fig. 1 (a)**.

Prediction errors of L-band nodes and triplets of the interpolation errors of H-band nodes can be arranged recursively into sub-band-like regions, as shown in **Fig. 1(b)** when K=3. The subscript EE (even-even) represents the L-band nodes at the $I_{l-1}(2i, 2j)$ index. The subscripts OO (odd-odd), EO (even-odd) and OE (odd-even) represent the H-band nodes with $I_{l-1}(2i+1,2j+1)$, $I_{l-1}(2i+1, 2j)$ and $I_{l-1}(2i, 2j+1)$ indices, respectively. The numerical subscripts attached to EE, OO, EO and OE determine the level of image decomposition.



(a) **(b)**

Fig. 1. **(a)** An example of multi-resolution image representation. Pixels labeled with H_i are removed at the multi-resolution pyramid level i; hence, L and H_2 pixels remain at I_1 and only L pixels remain at I_2. **(b)** The base band and triplets of interpolation errors are recursively arranged into sub-band-like regions when K=3.

For progressive resolution transmission, the lower-resolution image L_{EE-3} formed by the prediction errors of L-band nodes is first transmitted, and triplets of the interpolation errors of the H-band nodes (H_{OO-3}, H_{EO-3}, H_{OE-3}) are then transmitted to refine the lower resolution image L_{EE-3}. On the receiver side, the second-level image can be obtained by interpolating triplets of H-band nodes and adding the interpolation errors of H_{OO-3}, H_{EO-3} and H_{OE-3} to L_{EE-3}. The resulting image is labeled L_{EE-2}. This process is repeated for the remaining levels of the pyramid structure so that the set of displayed pyramid images is the ordered set {L_{EE-3}, L_{EE-2},

L_{EE-1}, I_o**}**, where I_o is the original image.

For progressive fidelity transmission, after the lower-resolution image L_{EE-3} is transmitted, the receiver expands the reduced resolution to the same resolution as the original image using an interpolative pyramid algorithm. The triplets of the interpolation errors of the H-band nodes (H_{OO-3}, H_{EO-3}, H_{OE-3}) are then transmitted. The receiver adds the interpolated values of the H-band nodes and their interpolation errors to replace the original values with the interpolated values. Hence, the visual quality of the image improves progressively. Repeating similar operations on the remaining levels, the original image can be reconstructed perfectly.

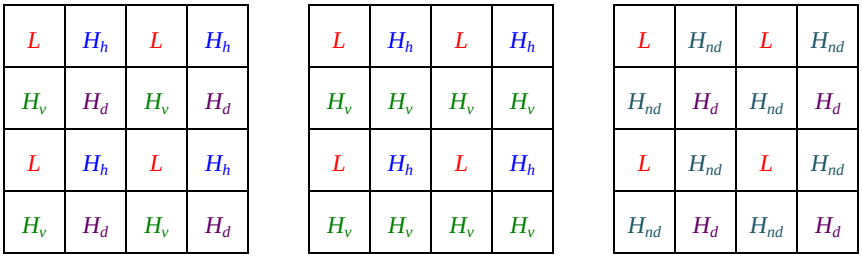


Fig. 2. Image decomposition schemes in **(a)** HINT, **(b)** IHINT, and **(c)** NIHINT.

2.1 The HINT Algorithm

The HINT algorithm proceeds in a sequence of rows of an input image from top to bottom and columns from left to right to interpolate missing pixels. As shown in **Fig. 2(a)**, horizontal nodes (H_h) at level l are interpolated using the average of two horizontally aligned L-band nodes from level l+1; similarly, vertical nodes (H_v) at level l are interpolated using the average of two vertically aligned L-band nodes from level l+1. Diagonal nodes (H_d) at level l are interpolated using the average of four L-band nodes from level l+1. Because previously decoded nodes lying on the same pyramid level of the current node are not exploited to interpolate the current node, the four better-fitting neighbors that form a quincunx shape around the current node are not available. Therefore, the errors of the diagonal sub-band nodes are much more energetic than those of the horizontal and vertical sub-band nodes.

2.2 The IHINT Algorithm

The IHINT algorithm splits the interpolation process into two cascaded directional stages interleaved with the encoding of interpolation errors. Interpolation is initially done along either the horizontal or the vertical direction, after which the other direction is done. As shown in **Fig. 2(b)**, horizontal nodes (H_h) are interpolated using the average of two L-band nodes, and vertical nodes (H_v) are then interpolated using the average of two L-band nodes or two horizontal nodes (H_h) at the same pyramid level. This shows that the pixels encoded in the first stage (horizontal interpolation in this example) can be used for the second stage of interpolation because the decoder reconstructs the original intensity values by adding the transmitted triplets of interpolation errors to the interpolated values. However, IHINT will be inadequate when the interpolation kernel is non-separable. Moreover, two out of the four best fitting neighbors are not available during the second stage of interpolation.

2.3 The NIHINT Algorithm

This algorithm decomposes the interpolation process into diagonal and non-diagonal stages, as shown in **Fig. 2(c)**. First, all diagonal nodes (H_d) are interpolated, followed by non-diagonal

nodes (H_{nd}). This order of interpolation is appropriate because once the original intensity values of all diagonal nodes (H_d) are reconstructed, the four best-fitting neighbors (two L-band nodes and two H_d-band nodes) for non-diagonal nodes are available.

In the first stage of interpolation for H_d nodes, a five-point FIR-median hybrid (FMH) filter was employed such that

$$Y = \text{Median of } \{P_1, P_2, P_3, P_4, P_{FIR}\}, \tag{2}$$

where P_1, P_2, P_3 and P_4 are the four nearest pixel values at the corner positions and P_{FIR} is the value obtained using a 7×7 FIR filter. In the second stage of interpolation for H_{nd} nodes, a seven-point FMH filter was employed such that

$$Z = \text{Median of } \{P_1, P_2, P_3, P_4, P_H, P_V, P_{FIR}\}, \tag{3}$$

where P_1, P_2, P_3 and P_4 are the four best-fitting neighbor pixels; P_H and P_V are two-point horizontal and vertical averages; and P_{FIR} is the value obtained using a 5×5 FIR filter.

2.4 Weakness of the HINT, IHINT and NIHINT Interpolating Algorithms

As shown clearly in **Fig. 2(b)** and **Fig. 2(c)**, for all H_h target pixels, the HINT and IHINT algorithms inherently assume that the left and right neighboring L-band pixels are closely related to the target pixel. However, this assumption is not effective when a vertical or diagonal edge exists around the target pixel. Similarly, for all H_v target pixels, these algorithms inherently assume that the upper and lower neighbor pixels are closely related to the target pixel. This assumption is not also effective when a horizontal or diagonal edge exists around the target pixel. This observation shows that the interpolation process remains mechanically fixed in horizontal or vertical directions without adapting to the direction of strong pixel correlation. Moreover, for all H_d target pixels, the HINT algorithm simply uses four-point average interpolation without considering the presence of any diagonal edges. The NIHINT algorithm simply uses the median operator without using an explicit technique to identify the direction of the strong correlation near the target pixel to exploit directional correlations among neighboring pixels. As a result, the interpolation errors by the NIHINT algorithm along sharp edges become large and thus cannot be coded efficiently by entropy coders. These algorithms also generate interpolated images with blurred edges and annoying artifacts near areas that contains sharp edges.

In this paper, we propose a new interpolation algorithm, termed Edge Adaptive Hierarchical INTerpolation (EAHINT). It adaptively exploits the pixel correlation along the edges of a target pixel. The proposed algorithm reduces the entropy of interpolation errors more than the existing algorithm NIHINT and does not generate either an over-blurred image or annoying artifacts around sharp edges.

3. Edge Adaptive Hierarchical Interpolation

Because a strong edge typically appears along the boundary between two regions of an image, the intensity value changes abruptly from one region to the other across the edge. The variance (σ^2) of the causal neighbors of a target pixel is large if the local window consists of pixels from two different regions. In contrast, the variance of the causal neighbors of a target pixel is small within slowly varying local areas. Hence, using the local variance as a measure of the local activity level, the local area can be classified as an edge area or a smooth area. Hence, the proposed algorithm can switch between a non-directional static weighting interpolator for a

smooth area and a one-directional interpolator for an edge area. However, the proposed algorithm uses the sub-sampled lower-resolution image as input to generate the higher-resolution image. If sub-sampling is done within an edge area, it may break down the local geometry [18]. In other words, if the decomposition removes the directional correlation, it is very difficult to infer the location and the direction of the edge within the higher-resolution image based on the edge information obtained from the lower-resolution image [19][20]. Thus, one-directional interpolation may not be effective in subsequent stages of image decomposition. Therefore, we adopt a multi-directional interpolator as another component in our algorithm.

Given two threshold values T_1 and T_2 ($T_1 < T_2$), three statistical decision rules are designed to detect the strength of a local edge. We apply one of them for the target pixel.

- If $\sigma^2 < T_1$, a weak edge is inferred within a local window and the local window is considered as a smooth area. We apply a non-directional static weighting linear interpolator to the target pixel.
- If $\sigma^2 > T_2$, we first partition the pixels within a local window into two groups according to the mean value μ of the pixels in a causal context; one group consists of the pixels whose intensities are greater than the mean and the other consists of the pixels whose intensities are less than or equal to the mean. The variance of each group is then computed as σ_1^2 and σ_2^2. If $\sigma^2 > \sigma_1^2 + \sigma_2^2$, a strong edge with a strong directional correlation is inferred within a local window and the local window is considered as a strong edge area. Note that the variance σ^2 should be compared with $\sigma_1^2 + \sigma_2^2$ because a region with uniformly distributed intensity values may result in a considerable amount of variance [21]. We apply a one-directional interpolator to the target pixel along the edge.
- Otherwise, that is, if $T_1 < \sigma^2 \leq T_2$ or ($\sigma^2 > T_2$ and $\sigma^2 \leq \sigma_1^2 + \sigma_2^2$), a medium edge is inferred within a local window and the local window is considered as an area with medium edge strength, which is considered as a textured area. We apply a multi-directional adaptive weighting interpolator to the target pixel as a compromise between a strong edge area and a smooth area. It can handle textured and noisy areas.

The two threshold values T_1 and T_2 are obtained empirically from statistical observations using the property of the variance. Assuming D_M be the maximum of the deviations D_i of the causal neighbors P_i such that $D_i = |P_i - \mu|$, we can obtain $\sigma^2 \leq D_M^2$. Initially, we set $D_M = 0$, indicating that the causal neighbors are completely smooth. We increase the value of D_M in steps of 1 while observing the change in the entropy and the visual quality due to the interpolation error. According to our statistical analysis, the reasonable ranges of T_1 and T_2 are [25, 36] and [225, 256], respectively, for most test images. Through extensive experiments, we found that $T_1 = 30$ and $T_2 = 250$ work very well for all test images. Thus, we used these threshold values in our experimental results. The pseudo code of the proposed EAHINT algorithm is represented below.

Pseudo Code 1. Edge Adaptive Hierarchical Interpolation

Let $X = (P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8)$ // See **Fig. 3**

$\mu = \frac{\sum_{i=1}^{8} P_i}{8}$ // Mean of the local window

$\sigma^2 = \frac{\sum_{i=1}^{8}(P_i - \mu)^2}{8}$ // Variance of the local window

Let G_1 and G_2 be two empty sets

if $\sigma^2 > T_2$ **then**

  foreach $P_i \in X$ **do**

    if $P_i > \mu$ **then**

      $G_1 = G_1 \cup P_i$

    else

      $G_2 = G_2 \cup P_i$

  Compute σ_1^2 and σ_2^2 for each group G_1 and G_2

  if $\sigma^2 > \sigma_1^2 + \sigma_2^2$ **then**

    One-Directional Interpolator to X // Pseudo Code 2

  else

    Multi-Directional Adaptive Wighting Interpolator to X // Pseudo Code 4

else if $T_1 < \sigma^2 \leq T_2$ **then**

  Multi-Directional Adaptive Weighting Interpolator to X // Pseudo Code 4

else

  Non-Directional Static Weighting Linear Interpolator to X // Pseudo Code 3

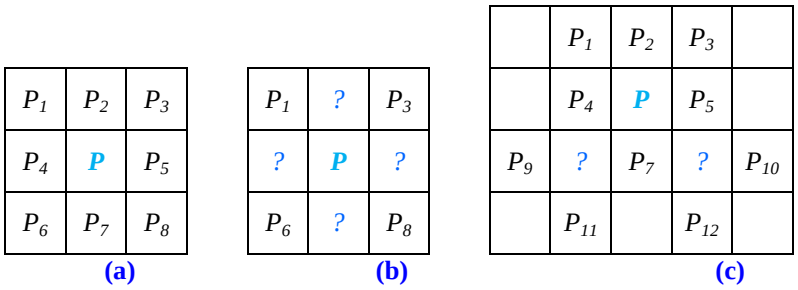


Fig. 3. **(a)** 3x3 local window to compute the variance, gradient, and edge orientation; **(b)** the pixels directly available in the first stage of interpolation; and **(c)** the pixels directly available in the second stage of interpolation.

In a regular raster scanning order, all of the pixels around the target pixel cannot be used for prediction because only the pixels on the top and to the left of the target pixel are known both to the encoder and the decoder. This context is known as the 180°-type context, as the causal context forms a semi-circle around the target pixel [22]. On the other hand, the NIHINT algorithm provides a causal context which encloses the current pixel completely. This type of context is known as the 360°-type context. With the 360°-type context, the local image features, such as the intensity gradient, the edge orientation, and the texture, are better modeled by a predictor. We use the 360°-type context (See **Fig. 3(a)**) to compute the local variance as a measure of the local activity level. We also use the 360°-type context to estimate local intensity gradients in different directions to detect the orientation of a strong edge if it exists.

However, as shown in **Fig. 3(b)**, only the four pixels (P_1, P_3, P_6, P_8) are available in the first stage of the interpolation directly from the upper level, where these pixels are labeled as L (See also **Fig. 2(c)**). The four best-fitting neighbors (P_2, P_4, P_5, P_7), which are labeled with question marks, are not available directly, where these pixels are labeled as H_{nd}. Hence, to estimate the variance, gradient, and edge orientation from the local geometry using the 360°-type context, the missing pixel intensity values have to be interpolated from pixel level

geometry using the IHINT algorithm as a sub-interpolation technique. Similarly, as shown in **Fig. 3(c)**, pixels P_2, P_4, P_5, P_7, P_9, and P_{11} are available directly in the second stage of the interpolation because they are labeled as either L or H_d (See also **Fig. 2(c)**). Pixels P_1 and P_3 are also available from the upper and left neighbors even if they are labeled as H_{nd}. We adaptively interpolate P_6 (and P_8) along their minimum intensity difference direction either horizontally or vertically if one intensity difference is less than a threshold value and the other difference is greater than the same threshold value. Otherwise, we simply take the average of their best-fitting neighbors P_4, P_9, P_7, and P_{11} (and P_5, P_7, P_{10}, and P_{12}, respectively). Note that the NIHINT algorithm did not exploit pixels P_1, P_3, P_6, or P_9.

3.1 One-Directional Interpolator

In principle, the direction of an edge is a continuous variable that ranges from 0^o to 180^o. Because it is very difficult to estimate the orientation of an edge in a continuous domain within a digitized local window, we quantize the local directional correlations into four directional angles (0^o, 45^o, 90^o and 135^o). The direction of a strong correlation to the target pixel is identified using well-known gradient estimation techniques [18][23][24][25].

If a one-directional interpolator is activated in the first stage of interpolation, the direction with the smallest local gradient is selected from the two diagonal directions. The target pixel is then interpolated using the two-point average along that direction. However, if this interpolator is activated during the second stage of interpolation, one direction is selected from four directions (horizontal, vertical, and the two diagonal directions). In this stage, the two diagonal gradients are computed from the sum of three absolute values of differences, whereas the horizontal and vertical gradients are computed from the sum of five absolute values of differences. As a result, the two diagonal directions are more likely to have minimum local variation and to be selected for one-directional interpolation compared to the horizontal and vertical directions. For a fair comparison, each diagonal gradient was multiplied by the constant integer m, where $m \geq 2$. Experimentally, we found that $m = 4$ works very well. The pseudo code of the one-directional interpolator is presented as follows, where $[\bullet]$ denotes the rounding operation to the nearest integer.

Pseudo Code 2. One-Directional Interpolator

Let $X = (P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8)$ // See **Fig. 3**
//In the first stage of interpolation
$\Delta d = |P_4 - P_2| + |P_6 - P_3| + |P_7 - P_5|$
$\Delta a = |P_4 - P_7| + |P_1 - P_8| + |P_2 - P_5|$
if $\Delta d < \Delta a$ **then**
    $P = [(P_3 + P_6)/2]$
else
    $P = [(P_1 + P_8)/2]$
// In the second stage of interpolation
$\Delta d = m\,(|P_4 - P_2| + |P_6 - P_3| + |P_7 - P_5|)$
$\Delta a = m\,(|P_4 - P_7| + |P_1 - P_8| + |P_2 - P_5|)$
$\Delta h = |P_1 - P_2| + |P_2 - P_3| + |P_4 - P_5| + |P_6 - P_7| + |P_7 - P_8|$
$\Delta v = |P_1 - P_4| + |P_4 - P_6| + |P_2 - P_7| + |P_3 - P_5| + |P_5 - P_8|$
if $\Delta h < \Delta v, \Delta d, \Delta a$ **then**
  $P = [(P_4 + P_5)/2]$
else if $\Delta v < \Delta h, \Delta d, \Delta a$ **then**
  $P = [(P_2 + P_7)/2]$
else if $\Delta a < \Delta h, \Delta v, \Delta d$ **then**

$P = [(P_3+P_6)/2]$
else
$P = [(P_1+P_8)/2]$

3.2 Non-Directional Static Weighting Linear Interpolator

If a non-directional static weighting linear interpolator is activated during the first stage of interpolation, the target pixel is interpolated only from the average of the four neighboring pixels located at the corner positions. If this interpolator is activated in the second stage, the average of the four best-fitting neighbor pixels (non-diagonal neighbors) is calculated first. Then, the average of the four neighboring pixels located at the corner positions (diagonal neighbors) is calculated. Finally, the two averages are combined through the multiplication of 0.95 and 0.05 weights to the non-diagonal and diagonal averages, respectively. Finding the two weights is empirical. However, these weights can be assigned by considering the inherent characteristics of the quincunx and rectangular sub-sampling shapes around the target pixel. The quincunx sub-sampling shape, formed around the target pixel by horizontal and vertical neighbors, does not reduce the resolution on "pure" horizontal and vertical frequencies. It limits only the high diagonal frequencies. In contrast, the rectangular sub-sampling shape formed by the corner neighbors will limit horizontal or vertical high frequencies. It will also limit diagonal high frequencies [6]. As a result, the horizontal and vertical neighbor pixels have higher weights than the diagonal neighbor pixels. The two static weights were found experimentally. The pseudo code of this interpolator is presented as follows.

Pseudo Code 3. Non-Directional Static Weighting Linear Interpolator

Let $X = (P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8)$ // See **Fig. 3**
// In the first stage of interpolation
$P = V_{da} = [(P_1+P_3 +P_6+P_8)/4]$
// In the second stage of interpolation
$V_{da} = [(P_1+P_3 +P_6+P_8)/4]$
$V_{hv} = [(P_2+P_4 +P_5+P_7)/4]$
$P = [0.95V_{hv} + 0.05V_{da}]$

3.3 Multi-Directional Adaptive Weighting Interpolator

If a multi-directional adaptive weighting interpolator is activated during the first stage of interpolation, two-point average interpolations will be performed first along the two diagonal directions. The diagonal gradients are then inverted and normalized to compute their corresponding directional weights [7][20]. Finally, the two directional interpolation results are combined using their corresponding directional weights to compute the final estimation of the target pixel. Hence, the higher the gradient in a particular direction, the lower the weight is assigned and vice versa. To make the weighting coefficients more different, each gradient is multiplied by itself k times before inversion and normalization, where k is an integer constant with $k \geq 2$. Experimentally, we found that $k = 3$ works very well.

However, if this interpolator is activated in the second stage, two-point average interpolation will be computed first along four directions. The diagonal gradients are then inverted and normalized to compute their corresponding directional weights. The horizontal and vertical gradients are also inverted and normalized to compute their corresponding directional weights. The two-point average interpolation results along the diagonal directions are combined using directional weights to estimate the target pixel from the diagonal neighbor pixels. Similarly, the two-point average interpolation results along the horizontal and vertical

directions are combined using directional weights to estimate the target pixel from the non-diagonal neighbor pixels. Finally, the two interpolation results obtained from the diagonal and non-diagonal neighbor pixels are combined by assigning 0.95 and 0.05 weights for the diagonal and non-diagonal estimations, respectively. The pseudo code representation of this interpolator is presented below.

Pseudo Code 4. Multi-Directional Adaptive Weighting Interpolator

Let $X = (P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8)$ // See **Fig. 3**
// In the first stage of interpolation
$I_d = [(P_3+P_6)/2]$
$I_a = [(P_1+P_8)/2]$
$\Delta d = |P_4 - P_2|+|P_6 - P_3|+|P_7 - P_5|$
$\Delta a = |P_4 - P_7|+|P_1 - P_8|+|P_2 - P_5|$
// compute the unnormalized coefficients u_d and u_a for I_d, and I_a, respectively.
// add 1 to the denominator in order to avoid division by zero
$u_d = 1/(\Delta d^k+1)$
$u_a = 1/(\Delta a^k+1)$
// compute the normalized coefficients n_d and n_a for I_d and I_a, respectively.
$n_d = u_d/(u_d+u_a)$
$n_a = u_d/(u_d+u_a)$
$P = V_{da} = [n_d I_d+n_a I_a]$
// In the second stage of interpolation
$I_d = [(P_3+P_6)/2]$, $I_a = [(P_1+P_8)/2]$
$I_h = [(P_4+P_5)/2]$, $I_v = [(P_2+P_7)/2]$
$\Delta d = |P_4 - P_2|+|P_6 - P_3|+|P_7 - P_5|$
$\Delta a = |P_4 - P_7|+|P_1 - P_8|+|P_2 - P_5|$
$\Delta h = |P_1 - P_2|+|P_2 - P_3|+|P_4 - P_5|+|P_6 - P_7|+|P_7 - P_8|$
$\Delta v = |P_1 - P_4|+|P_4 - P_6|+|P_2 - P_7|+|P_3 - P_5|+|P_5 - P_8|$
// compute the unnormalized coefficients for I_d, I_a, I_h and I_v
// add 1 to the denominator in order to avoid division by zero
$u_d = 1/(\Delta d^k+1)$
$u_a = 1/(\Delta a^k+1)$
$u_h = 1/(\Delta h^k+1)$
$u_v = 1/(\Delta v^k+1)$
// compute the normalized coefficients for I_d, I_a, I_h and I_v
$n_d = u_d/(u_d+u_a)$
$n_a = u_d/(u_d+u_a)$
$n_h = u_v/(u_h+u_v)$
$n_v = u_h/(u_h+u_v)$
$V_{da} = [n_d I_d+n_a I_a]$
$V_{hv} = [n_h I_h+n_v I_v]$
$P = [0.95V_{hv} + 0.05V_{da}]$

4. Experimental Results and Discussion

In this section, we show our experimental results to compare the performance of the proposed algorithm EAHINT to that of NIHINT. Five common natural images 512x512 in size with 8 bits per pixel, specifically Baboon, Boat, Lena, Barbara, and Pepper, are used in our experiments. Because the input image is encoded only once to a single bit stream, the total lossless bit rate is a common criteria to measure the performance of a multi-resolution lossless image compression scheme. When multi-resolution lossless image compression is used for

progressive transmission, the decoder performs the up-scaling of each layer to obtain the next layer progressively. Therefore, the rate distortion and the visual quality at the intermediate level of transmission are used to evaluate the performance of the interpolation algorithms.

4.1 Total Lossless Compression Bit Rate

To compute the total lossless compression bit rate, a different codebook was assumed to encode each pyramid level. The progressive bit rate up to level l, $R(l)$, is equal to the total accumulated bit rate up to level l such that

$$R(l) = R(l+1) + r(l), \tag{4}$$

where $R(l+1)$ is the progressive bit rate up to its upper level and $r(l)$ is the bit rate used for transmitting the data at level l [2]. However, the bit rate of each pyramid level $r(l)$ is given in terms of bits per node (bpn) and therefore must be normalized to the actual number of pixels in the original image. Hence, we use the equivalent pyramid entropy H_0^{eq} measured at the pixel-by-pixel level for comparison purposes such that

$$H_0^{eq} = (N_n/N_p)\, H_0(l), \tag{5}$$

where $H_0(l)$ is the zeroth-order entropy of the l-th level of the differential pyramid measured on a node-by-node basis, and N_n and N_p are the numbers of nodes and pixels, respectively. For the NIHINT and EAHINT pyramid algorithms, note that $N_n/N_p = 1/4^K$ for the base-level K and $N_n/N_p = 1/4^{l+1}$ when $0 \le l \le K$-1. As these interpolative pyramid algorithms guarantee lossless reproduction after transmitting the bottom level of the pyramid, we simply use the total accumulated bit rate up to the bottom-level $l = 0$ as the total lossless pyramid bit rate R_L such that

$$R_L = \frac{H_0(G_K)}{4^K} + \sum_{l=0}^{K-1} \frac{3H_0(D_l)}{4^{l+1}}, \tag{6}$$

where $H_0(G_K)$ and $H_0(G_K)/4^K = H_0^{eq}(G_K)$ denote the zeroth-order entropy in bpn and the zeroth-order equivalent entropy in bpp of the differential pyramid at the base-band level K, respectively. $H_0(D_l)$ and $3H_0(D_l)/4^{l+1} = H_0^{eq}(D_l)$ denote the zeroth-order entropy in bpn and the zeroth-order equivalent entropy in bpp of the differential pyramid at level l [5].

The entropy values of the Lena image computed by NIHINT and by the proposed EAHINT algorithms are shown in **Table 1** for different base-band level K values. When the base-band level K value increases, the percentage of entropy reduction by the EAHINT algorithm is marginal. Due to the absence of low-pass filtering, sub-sampling breaks down the local geometry as the base-band level K value increases, making it difficult to process the next level of image decomposition. Moreover, the intermediate images will have poor approximation to the original image due to accumulated aliasing artifacts [5][18][19][20]. Based on this reasoning, the base-band level in interpolating pyramid algorithms is usually set to K=2 or K=3. The total lossless compression bit rate R_L for the other test images is also presented in **Table 2** when K=3. The EAHINT algorithm reduces the total lossless bit rate R_L significantly compared to the NIHINT algorithm for the Barbara, Boat and Lena images, whereas the reduction is marginal for the Baboon and Pepper images. This result was expected somewhat because the Boat image contains many sharp edges, the Barbara image contains strong directional textures, and the Lena image has fairly sharp edges with a small amount of texturing. However, the Baboon image has numerous fine details that are scarcely predictable, and the Pepper image has many smooth regions that are separated by sharp edges, where the total number of strong edge pixels is low compared to the total number of pixels.

Table 1. $H_0(D_l)$ and $H_0^{eq}(D_l)$ of EAHINT and NIHINT methods for the test image Lena while varying the base-band level K from 1 to 3. $H_0(G_K)$ and $H_0^{eq}(G_K)$ are generated using the Median Edge Detector (MED) of the JPEG-LS standard image compression. For R_L, the gain represents the percentage of entropy reduction by EAHINT compared to NIHINT.

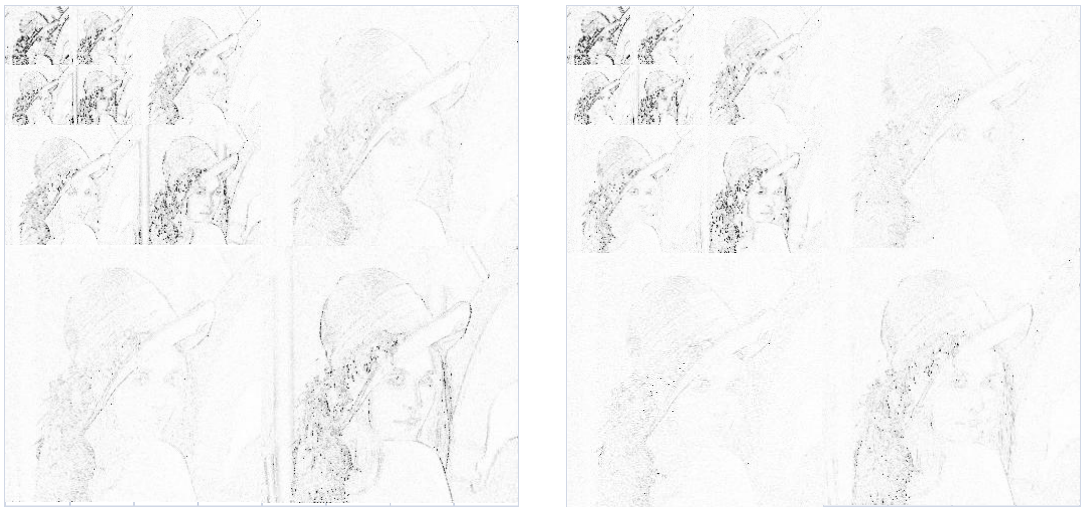
l	Entropy	K=1		K=2		K=3	
		NIHINT	EAHINT	NIHINT	EAHINT	NIHINT	EAHINT
3	H_0	-	-	-	-	6.1108	6.1108
	H_0^{eq}	-	-	-	-	0.0955	0.0955
2	H_0	-	-	5.6412	5.6412	5.5442	5.3070
	H_0^{eq}	-	-	0.3526	0.3526	0.2599	0.2488
1	H_0	5.1587	5.1587	4.9734	4.7303	4.9744	4.7303
	H_0^{eq}	1.2897	1.2897	0.9325	0.8869	0.9327	0.8869
0	H_0	4.3601	4.0479	4.3601	4.0479	4.3601	4.0479
	H_0^{eq}	3.2701	3.0359	3.2701	3.0359	3.2701	3.0359
R_L		4.5598	4.3256	4.5552	4.2754	4.5582	4.2671
Gain		-	5.14%	-	6.14%	-	6.39%

Table 2. R_L of EAHINT and NIHINT for the base-band level K=3. The gain represents the percentage of entropy reduction by EAHINT compared to NIHINT.

R_L	Baboon	Barbara	Boat	Pepper	Lena
NIHINT	6.3342	5.5360	5.1946	4.7419	4.5582
EAHINT	6.1350	5.2048	4.9118	4.5211	4.2671
Gain	3.14%	5.98%	5.44%	4.66%	6.39%

Fig. 4 shows two differential pyramids of the Lena image generated by the NIHINT and EAHINT algorithms after three levels of image decomposition. It is clear that the H-band sub-images generated by the EAHINT algorithm show less signal energy. Therefore, they are much less noticeable. The reduction in energy in this case shows that better compression results can be obtained using the proposed algorithm compared to NIHINT.

Generally, prediction errors fall in the closed interval [-255,255] for 8-bit gray-scale images. These prediction errors are mapped into the interval [0,255] using a simple mapping technique [23] to display differential images, as shown in **Fig. 4**. To show the superior performance of the EAHINT algorithm pictorially in comparison to NIHINT along sharp edges and textured areas, we generated the black-and-white image shown in **Fig. 5**, as follows. For any pixel p in the image, let e_E and e_N be prediction errors in the interval [0,255] which are generated by the EAHINT and NIHINT algorithms, respectively. If the value of e_E is less than the value of e_N, black color is assigned for the corresponding pixel p. Otherwise, white color is assigned. **Fig. 5** shows that our interpolation scheme is superior to NIHINT along sharp edges and around textured areas. Because e_E and e_N are equal in the base-band image (see **Table 1**), the top left image is completely white.



(a) **(b)**

Fig. 4. Differential pyramids of the Lena image generated by (a) NIHINT and (b) EAHINT after three levels of image decomposition. The darker the pixel, the larger the error. As shown in **Table 1**, the top left image is identical in both algorithms because this image is the base-band level image.

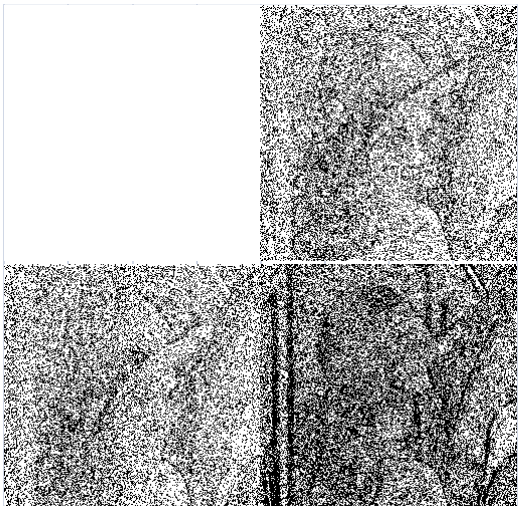


Fig. 5. Comparison of EAHINT and NIHINT algorithm results along sharp edges or around textured areas after one level of image decomposition for the Lena image. The black pixels demonstrate that the EAHINT is superior to NIHINT along sharp edges or around textured areas.

4.2 Rate Distortion and Visual Quality at Intermediate Levels

Given a progressive bit rate up to level l, to estimate the average distortion of interpolated images at intermediate levels l of the pyramid, the receiver expands the reduced resolution to the same resolution as the original image using an interpolative pyramid algorithm. The average distortion of the interpolated image compared to the original image is measured using an objective image quality metric known as the Peak Signal-to-Noise Ratio (PSNR). The progressive entropy in bpp and the PSNR values of all test images are summarized in **Table 3** for the base-band level K=3. Our EAHINT algorithm shows a marginal improvement in the PSNR for the Baboon image, where numerous and scarcely predictable fine details exist.

Meanwhile, it shows significant improvement for images that have strong directional textures (Barbara), small amounts of texture with fairly sharp edges (Lena), or many sharp edges (Boat).

Fig. 6 and **Fig. 7** show zoomed-in portions of the Lena and Pepper images to compare the visual quality of the interpolation algorithms. The proposed algorithm preserves sharp edges better and generates images of higher visual quality compared to the NIHINT algorithm.

Table 3. The entropy and PSNR of the images interpolated by NIHINT and EAHINT for the base-band level K=3. The gain represents the percentage of PSNR enhancement by EAHINT compared to NIHINT.

image	l	NIHINT		EAHINT		
		Entropy	PSNR	Entropy	PSNR	Gain
Baboon	3	0.11	18.4348	0.11	18.5414	0.59%
	2	0.44	19.6941	0.43	19.7640	0.35%
	1	1.69	22.2330	1.68	23.0407	3.63%
Barbara	3	0.11	19.9701	0.11	20.6506	3.41%
	2	0.40	21.7546	0.39	22.3056	2.53%
	1	1.51	24.1009	1.47	25.3870	5.34%
Boat	3	0.10	20.5483	0.10	21.7256	5.73%
	2	0.38	23.3334	0.37	24.7363	6.01%
	1	1.42	27.5309	1.38	30.0667	9.21%
Pepper	3	0.10	20.9992	0.10	23.2694	10.81%
	2	0.36	24.9267	0.35	27.8741	11.82%
	1	1.28	29.4197	1.24	33.0776	12.43%
Lena	3	0.10	22.3863	0.10	24.4215	9.09%
	2	0.36	25.8468	0.34	28.3859	9.82%
	1	1.29	30.8009	1.23	34.1301	10.8%

Fig. 6. **(a)** Zoomed-in portion of the original Lena image, **(b)** the image interpolated by EAHINT, and **(c)** the image interpolated by NIHINT from top to bottom of pyramid levels l=3 (from 64x64 to 512x512), l=2 (from 128x128 to 512x512), and l=1 (from 256x256 to 512x512).

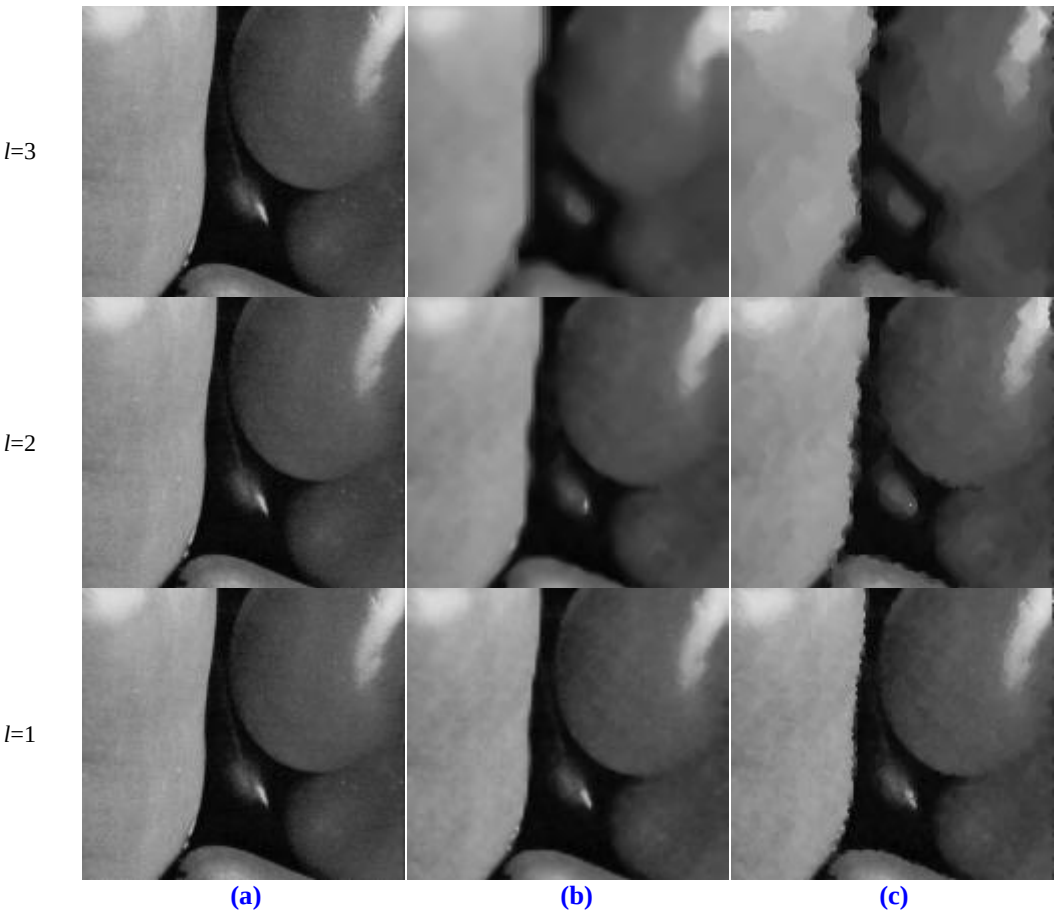


Fig. 7. **(a)** Zoomed-in portion of the original Pepper image, **(b)** an interpolated image by EAHINT, and **(c)** an interpolated image by NIHINT from top to bottom of pyramid levels l=3 (from 64x64 to 512x512), l=2 (from 128x128 to 512x512), and l=1 (from 256x256 to 512x512).

5. Conclusion

In this paper, a new image interpolation algorithm, Edge Adaptive Hierarchical INTerpolation (EAHINT) is presented. To exploit directional correlations from the 360°-type context fully,

the missing neighbor pixels are interpolated using the IHINT algorithm as a sub-interpolation technique in the first stage of interpolation, after which an adaptive IHINT algorithm is used as a sub-interpolation technique during the second stage of interpolation. The proposed algorithm also uses the upper and left neighboring pixels from the second stage of interpolation, whereas the NIHINT algorithm does not use these neighboring pixels. The proposed algorithm also adopts the two stages of interpolation used in the NIHINT algorithm. All of these conditions allow the proposed algorithm to model local image features such as the variance, gradient and edge orientation information in a better way.

According to the strength of the local edge, the interpolator switches adaptively between a non-directional static weighting linear interpolator for smooth areas, a one-directional interpolator for sharp edge areas, and multi-directional adaptive weighting interpolator for areas with medium edge strength. Although the actual interpolators are still linear functions, the switching mechanism can attempt to deal with non-linear local structures such as edges and textures. The one-directional interpolator and the multi-directional adaptive weighting interpolator are activated only when an edge with strong or medium strength is detected. As a result, this hybrid approach involving static and adaptive interpolators reduces the overall computational complexity.

We presented experimental results to evaluate the performance of the proposed algorithm with respect to NIHINT. The experimental results showed that the proposed algorithm achieved better total lossless bit rates for multi-resolution lossless image compression while achieving better rate distortion and visual quality for progressive image transmission.

The limitation of EAHINT comes mainly from its use of the selected threshold values T_1 and T_2 in **Pseudo Code 1**. Statistically selected T_1 and T_2 values may not achieve the best performance for images which are not evaluated in this paper. This limitation remains as future work.

References

[1] A.J. Penrose, "Extending Lossless Image Compression," *Technical Report UCAM-CL-TR-526*, University of Cambridge, Dec. 2001.

[2] M. Goldberg, L. Wang, "Comparative Performance of Pyramid Data Structures for Progressive Image Transmission," *IEEE Transactions on Communications*, vol. 39, no. 4, pp. 540-548, Apr. 1991. Article (CrossRef Link)

[3] Y.-K. Chee, "Survey of Progressive Image Transmission Methods," *International Journal of Imaging Systems and Technology*, vol. 10, no. 1, pp. 3-19, 1999. Article (CrossRef Link)

[4] A. Said, W.A. Pearlman, "An Image Multiresolution Representation for Lossless and Lossy Compression," *IEEE Transactions on Image Processing*, vol. 5, no. 9, pp. 1303-1310, Sep. 1996. Article (CrossRef Link)

[5] P. Roos, M. Viergever, M. van Dijke, J. Peters, "Reversible Intraframe Compression of Medical Images," *IEEE Transactions on Medical Imaging*, vol. 7, no. 4, pp. 328-336, Dec. 1988. Article (CrossRef Link)

[6] A. Abrardo, L. Alparone, F. Bartolini, "Encoding-Interleaved Hierarchical Interpolation for Lossless Image Compression," *Signal Processing*, vol. 56, no. 3, pp. 321-328, Feb. 1997. Article (CrossRef Link)

[7] B. Zeng, M.S. Fu, C.C. Chuang, "New Interleaved Hierarchical Interpolation with Median-based Interpolators for Progressive Image Transmission," *Signal Processing*, vol. 81, no. 2, pp. 431-438, Feb. 2001. Article (CrossRef Link)

[8] X. Li, M. Orchard, "New Edge-Directed Interpolation," *IEEE Transactions on Image Processing*, vol. 10, no. 10, pp. 1521-1527, Oct. 2001. Article (CrossRef Link)

[9] N. Asuni, A. Giachetti, "Accuracy Improvements and Artifacts Removal in Edge based Image Interpolation," in*Proc. of 3rd International Conference on Computer Vision Theory and Applications*, vol. 1, pp. 58-65, 2008.
[10] X. Zhang and X. Wu, "Image interpolation by adaptive 2-D autoregressive modeling and soft-decision estimation," *IEEE Transactions on Image Processing*, vol. 17, no. 6, pp. 887-896, June 2008. Article (CrossRef Link)
[11] W. Tam, C. Kok, W. Siu, "A Modified Edge Directed Interpolation for Images," in *Proc. of 17th European Signal Processing Conference*, pp. 283-287, Aug. 2009.
[12] K. Hung, W. Siu, "Improved Image Interpolation using Bilateral Filter for Weighted Least Square Estimation," in *Proc. of 17th IEEE Internal Conference on Image Processing*, pp. 3297-3300, Sep. 2010. Article (CrossRef Link)
[13] D. Garcia, R. de Queiroz, "Least-Squares Directional Intra Prediction in H.264/AVC," *IEEE Signal Processing Letters*, vol. 17, no. 10, pp. 831-834, Oct. 2010. Article (CrossRef Link)
[14] J. Han, J. Kim, S. Cheon, J. Kim, S. Ko, "A Novel Image Interpolation Method Using the Bilateral Filter," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 1, pp. 175-181, Feb. 2010. Article (CrossRef Link)
[15] W. Dong, L. Zhang, G. Shi, X. Wu, "Nonlocal Back-Projection for Adaptive Image Enlargement," in *Proc. of 16th IEEE International Conference on Image Processing*, pp. 349-352, Nov. 2009. Article (CrossRef Link)
[16] Y. Wee, H. Shin, "A Novel Fast Fractal Super Resolution Technique," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 3, pp. 1537-1541, Aug. 2010. Article (CrossRef Link)
[17] Y. Lee, J. Yoon, "Nonlinear Image Upsampling Method based on Radial Basis Function Interpolation," *IEEE Transactions on Image Processing*, vol. 19, no. 10, pp. 2682-2692, Oct. 2010. Article (CrossRef Link)
[18] D. Su, P. Willis, "Image Interpolation by Pixel-Level Data-Dependent Triangulation", *Computer Graphics Forum*, vol. 23, no. 2, pp. 189–201, June 2004. Article (CrossRef Link)
[19] W. Ding, F. Wu, X. Wu, S. Li, H. Li, "Adaptive Directional Lifting-based Wavelet Transform for Image Coding," *IEEE Transactions on Image Processing*, vol. 16, no. 2, pp. 416-427, Feb. 2007. Article (CrossRef Link)
[20] S. Bayrakeri, R. Mersereau, "A New Method for Directional Image Interpolation," in *Proc. of International Conference on Acoustics, Speech, and Signal Processing*, vol. 4, pp. 2383-2386, May 1995. Article (CrossRef Link)
[21] L.-J. Kau, Y.-P. Lin, "Least-Squares-based Switching Structure for Lossless Image Coding," *IEEE Transactions on Circuits and Systems I*, vol. 54, no. 7, pp. 1529-1541, July 2007. Article (CrossRef Link)
[22] X. Wu, "Lossless Compression of Continuous-Tone Images via Context Selection, Quantization, and Modeling," *IEEE Transactions on Image Processing*, vol. 6, no. 5, pp. 656-664, May 1997. Article (CrossRef Link)
[23] X. Wu, N. Memon, "Context-based, Adaptive, Lossless Image Coding," *IEEE Transactions on Communications*, vol. 45, no. 4, pp. 437-444, Apr. 1997. Article (CrossRef Link)
[24] O.N. Gerek, A.E. Cetin, "Lossless Image Compression using an Edge Adapted Lifting Predictor," in *Proc. of IEEE International Conference on Image Processing*, vol. 2, pp. II-730-3, Sep. 11-14, 2005. Article (CrossRef Link)
[25] M. Weinberger, G. Seroussi, G. Sapiro, "The LOCO-I Lossless Image Compression Algorithm: Principles and Standardization into JPEG-LS," *IEEE Transactions on Image Processing*, vol. 9, no. 8, pp. 1309-1324, Aug. 2000. Article (CrossRef Link)

Yenewondim Biadgie received his B.S. degree in Mathematics in 2000 from Bahirdar University in Ethiopia and his M.S. degree in Information Science in 2006 from Addis Ababa University in Ethiopia. He worked for Arbaminch University and Addis Ababa University for 2002-2003 and 2007-2008, respectively. He is currently a Ph.D. student in Computer Engineering at Ajou University. His Research interests are pattern recognition, digital image processing, digital image compression, and digital video compression.

Young Chul Wee received his B.S. degree in 1982, his M.S. degree in 1984, and his Ph.D. degree in 1989, all in Computer Science from University of NY at Albany. He worked for Samsung Electronics and Hyundai Electronics for 1990-1995 and 1995-1998, respectively. He is currently an Associate Professor in the Department of Information and Computer Engineering at Ajou University. His research interests are image processing, information compression, and computer graphics.

Jung-Ju Choi is an Associate Professor of Digital Media at Ajou University, Korea. He received his B.S. in Computer Science from KAIST in 1990 and his M.S. and Ph.D. in Computer Science and Engineering from POSTECH in 1992 and 1997, respectively. His research interests include computer graphics, image and video processing, and mobile multimedia systems.