Improved image magnification algorithm based on Otsu thresholding

Suheir M. ElBayoumi Harb a, Nor Ashidi Mat Isa a,*, Samy A. Salamah b

a Imaging and Intelligent Systems Research Team (ISRT), School of Electrical and Electronic Engineering, Universiti Sains Malaysia, 14300 Nibong Tebal, Penang, Malaysia
b Computer and Engineering Department, Palestine Technical College, DeirAlbalah, Gaza, Palestine

Article info
Article history:
Received 3 April 2014
Received in revised form 23 March 2015
Accepted 24 March 2015
Available online 20 April 2015

Keywords:
Cubic convolution interpolation
Image magnification
Gradient
Otsu thresholding

Abstract
An improved image magnification algorithm for gray and color images is presented in this paper to meet the challenge of preserving high-frequency components of an image, including both image edges and texture structures. In the proposed algorithm, a new edge detection method that uses the well-known Otsu automatic optimum thresholding is proposed to distinguish strong edge pixels. The parameters of the original directional cubic convolution interpolation algorithm, which were selected based on training, were eliminated. As a result, our algorithm achieves more accurate edge detection, better interpolation results, and less computational complexity. Simulation results demonstrate that the improved algorithm can reconstruct the magnified image, preserve edges and textures simultaneously, and reduce common interpolation artifacts. Furthermore, it generates higher visual quality of the magnified images and achieves higher peak signal-to-noise ratio, structural similarity, and feature similarity compared with other state-of-the-art methods.

1. Introduction

Image magnification, which is known as producing high-resolution (HR) images from their low-resolution (LR) counterparts, is highly in demand nowadays and has attracted increasing attention in the research community because of its wide applications in consumer electronics, remote sensing, medical imaging, advertising and printing, and many others. Interpolation is commonly employed in image magnification to estimate unknown pixel values from known pixel values [1]. Many interpolation methods, including adaptive and non-adaptive methods, have been proposed in the last several decades. Classical non-adaptive methods, such as linear and cubic convolution methods [2] are preferable because of their computational simplicity, however, these methods are unable to adapt with varying local structures of a LR image, which causes undesirable artifacts such as blurring, blocking, and ringing around edges [3]. Several adaptive interpolation methods [2,4–15], including edge-directed methods, have been proposed to address the problems of the aforementioned algorithms and to improve the perceptual quality of the interpolated images. Edge-directed methods, which aim for interpolation along edges and not across them, can be classified into two main types: explicit [2,7,10,13,15,16] and implicit [4,6,9,12,14]. Explicit methods perform interpolation along explicitly estimated edge orientations, whereas implicit methods utilize edge characteristics such as orientation and energy via parametric functions [12].

* Reviews processed and recommended for publication to the Editor-in-Chief by Associate Editor Dr. Ferat Sahin.
* Corresponding author. Tel.: +60 45996051; fax: +60 45941023.
E-mail addresses: suheirma@gmail.com (S.M. ElBayoumi Harb), ashidi@eng.usm.my (N.A.M. Isa), Samy_ptc@yahoo.com (S.A. Salamah).

http://dx.doi.org/10.1016/j.compeleceng.2015.03.025
0045-7906/© 2015 Elsevier Ltd. All rights reserved.
Li and Orchad [4] proposed a technique to estimate the orientation of edges implicitly by exploiting their geometric duality to estimate the covariance of the HR image from the local covariance of the LR image. Their method guarantees appealing visual quality and smoothed long-edge structures. However, it fails to preserve textures as well as produces some artifacts along edges. Zhang and Wu [6] presented a directional filtering and data fusion technique that interpolates the missing pixel in two orthogonal directions and fuses the results by using linear minimum mean square error estimation, while Giachetti and Asuni [10] proposed an iterative curvature-based interpolation method (ICBI) that interpolates the pixels locally along the direction with a lower second-order derivative value and exploits iterative refinement to remove artifacts and preserve edges and details. However, although ICBI can produce an image with a natural appearance, the final image exhibits ringing artifacts around edges and shows discontinuities in ridges. In addition, ICBI requires high computational cost.

In [8], Getreuer proposed an edge-adaptive image interpolation method. In the method, the local orientation of the image contours is detected by measuring the total variation over a contour stencil. Small total variation along a contour stencil candidate curve indicates approximate smoothness along that curve and thus the stencil is a good approximation to the contour. The method is computationally efficient and shows ability to interpolate edge structures but shows less ability to interpolate texture structures.

Jurio et al. [11] proposed a simple image magnification algorithm based on intervals. In this algorithm, a block expanding approach in which a new block is constructed for every LR image pixel is employed. Each block in the HR image is obtained by a weighted aggregation of the intensities of the pixels in the neighborhood of each pixel in the LR image by making use of the interval information and a linear operator. The interval is constructed from a LR pixel and its neighborhood where the length of the interval is used as a measure of the variation of intensities in the neighborhood. The algorithm is computationally efficient. However, blocking artifacts in edge areas and some pixilation in texture areas are problems of this algorithm.

Recently, Jing and Wu [14] proposed a simple and fast directional inverse distance weighting interpolation method (DIDWI). This method is designed to estimate the intensity distance in the HR image by using the intensity distance in the LR image based on geometric duality between distances in the LR and the HR images along the same direction. However, while DIDWI preserves texture areas, it still suffers from blocking artifacts along strong edges.

Zhou et al. [2] proposed a new directional cubic convolution interpolation (DCCI) algorithm for image zooming. This algorithm explicitly decides the local edge direction based on the ratio of the two orthogonal directional gradients and a preset threshold before interpolating the missing pixel by using cubic convolution interpolation along the detected edge direction. For texture areas, the missing pixel is interpolated by combining the interpolated cubic convolution values and the two orthogonal directional gradients adjusted by a parameter that was determined experimentally. The results, which are heavily dependent on the selection of the two parameters mentioned earlier, show that the DCCI algorithm preserves image edge structures but suffers from directional artifacts in certain texture areas.

In this paper, an improved version of the DCCI algorithm (DCCI_Otsu) is proposed in which a new strong edge detection method is presented. The proposed detection method employs the well-known automatic optimum Otsu thresholding. By eliminating the use of the DCCI parameters, the proposed DCCI_Otsu achieves much more stable results and better magnified image quality in terms of edge and texture preservation with lower computational time. Furthermore, the improved algorithm can magnify color images.

The remainder of this paper is organized as follows: Section 2 presents the related work including the description of the DCCI algorithm and its limitation and Otsu thresholding method. The proposed DCCI_Otsu is presented in Section 3. Section 4 provides the experimental results, including the qualitative and quantitative evaluations. Finally, Section 5 concludes the paper.

2. Related work

In this section, the directional cubic convolution interpolation algorithm and its limitation is first described. Then, a quick review on thresholding is introduced followed by a brief description of Otsu optimum thresholding that will be employed in the proposed DCCI_Otsu.

2.1. DCCI and its limitation

As adopted in various methods [4–9], the DCCI [2] algorithm initially expands the source LR image \( X_{(H \times W)} \) with dimension \((H \times W)\) into an HR image \( Y_{(2H-1 \times 2W-1)} \) with size \((2H-1 \times 2W-1)\) for a scaling factor of 2. The missing pixels are interpolated in a two-stage filling process. The original pixels from the LR image are depicted in Fig. 1 as solid black circles and the missing pixels are depicted as squares and white circles. The interior white circles are interpolated in the first stage, whereas the square aligned pixels are interpolated in the second stage. Pixels interpolated in the second stage have two orientations marked by gray squares and black squares; interpolation of gray square missing pixels is shown in Fig. 1(f). In the first stage, in a \(7 \times 7\) window around every missing interior pixel \(p\), the two orthogonal directional gradients \(G_1\) and \(G_2\) are computed in the diagonal directions 45 and 135, respectively, as in (1) and (2). Gradients used to compute both \(G_1\) and \(G_2\) are depicted by Figs. 1(a) and 1(b), respectively. The edge direction at the missing pixel is detected by comparing the ratio of \(G_1\) and \(G_2\) by using a threshold \(th\) that has been set experimentally to 1.15 based on training. The missing pixel is computed by using the dot product of the intensity values for the four known pixels along the detected edge and the cubic convolution kernel...
For illustration, the four known pixels along diagonal directions 45 and 135 are shown in Fig. 1(c). The missing pixel \( p \) along diagonal direction 45 is computed as \( p_1 \), while the missing pixel along the diagonal direction 135 is computed as \( p_2 \). For non-edge pixels, the missing pixel \( p \) is interpolated by using \( p_1 \) and \( p_2 \) as in (3), where the weights \( w_1 \) and \( w_2 \) that combine \( p_1 \) and \( p_2 \) are computed as in (4) and (5), respectively.

\[
G_1 = \sum_{p=3,i=3}^{1} \sum_{q=3,j=3}^{1} |y_{2i+p,2j+q} - y_{2i+p-2,2j+q+2}|, \\
G_2 = \sum_{p=3,i=3}^{1} \sum_{q=3,j=3}^{1} |y_{2i+p,2j+q} - y_{2i+p-2,2j+q-2}|,
\]

The parameter \( k \) in (4) and (5) is also determined experimentally and was set to 5. It has been used to adjust the weights of the interpolation coefficients.

\[
p = \frac{w_1 p_1 + w_2 p_2}{w_1 + w_2},
\]

\[
w_1 = \frac{1}{1 + G_1}
\]

\[
w_2 = \frac{1}{1 + G_2}
\]

The interpolated pixels in the first stage are considered as the already known pixels in the second stage and are depicted as gray circles in Fig. 1(d) to 1(f). As mentioned earlier, the missing aligned pixels have two orientations. In this stage, edge detection and pixel interpolation are identical for both orientations and similar to the previous stage. However, edge detection is performed in a 5 \times 5 window. Considering the interpolation of the missing gray square aligned pixels, the horizontal gradient \( G_3 \) and the vertical gradient \( G_4 \) are computed as in (6) and (7), respectively. The gradients used to compute \( G_3 \) are depicted in Fig. 1(d), while the gradients used to compute \( G_4 \) are depicted in Fig. 1(e). The values \( p_1 \) and \( p_2 \) have to be

---

**Fig. 1.** (a)-(c) A 7 \times 7 window used to compute gradients and pixel interpolation in the first stage. (a) Gradients in the diagonal 45 direction; (b) Gradients in the diagonal 135 direction; (c) Solid black circles along the lines represent the pixels that participate in the interpolation along the diagonal 45 direction and along the diagonal 135 direction; (d)-(f) The dotted square embraces the 5 \times 5 window used to compute the vertical and horizontal gradients in the second stage. Gray circles are the pixels that were filled in the first stage. (d) Gradients in the vertical direction; (e) Gradients in the horizontal direction; (f) Solid black circles along the vertical line represent the pixels that participate in the interpolation along the vertical direction, whereas gray circles along the horizontal line represent the pixels that participate in the interpolation along the horizontal direction.
calculated in vertical and horizontal directions, respectively. Fig. 1(f) shows the pixels that are involved in computing \( p1 \) along the vertical direction and \( p2 \) along the horizontal direction. The missing pixel \( p \) can be produced in a similar manner as in the first stage. (For more details, refer to [2].)

\[
G_3 = \sum_{p=0, q=0}^{2} |y_{2i-1, p2j-1} - y_{2i+1, p2j-1}| + \sum_{p=0, q=1}^{2} |y_{2i-1, p2j-1} - y_{2i+1, p2j+1}|, 
\]

\[
G_4 = \sum_{p=0, q=0}^{2} |y_{2i-1, 2j-1} - y_{2i+1, 2j-1}| + \sum_{p=0, q=1}^{2} |y_{2i-1, 2j-1} - y_{2i+1, 2j+1}|, 
\]

The performance of the DCCI algorithm heavily depends on the selection of two parameters: the parameter \( th \) used in edge detection and the parameter \( k \) used in adjusting the interpolation weights. The \( k \) exponent parameter lies in the interval \([1,5]\) and is incremented by step 1, whereas the \( th \) threshold parameter varies from 1 to 1.5 at a 0.05 step. The selection of values \((k = 5 \text{ and } th = 1.15)\) was justified in [2]. Thus, these values were used in the DCCI algorithm. To investigate the effect of the parameter selection on the performance of the DCCI algorithm, the gray versions of the 24 Kodak color images [16] were directly down-sampled by a factor of 2 and then magnified by the same factor with the use of a combination of different \( th \) and \( k \) values, as conducted in [2]. The average peak signal-to-noise ratio (PSNR) of the DCCI magnified images using every threshold at different \( k \) values was computed. The PSNR curves used to justify the selection of the parameters have been reproduced in this study and are shown in Fig. 2.

Considering the selected threshold value (1.15) reported in [2], Fig. 2 reveals that the PSNR curves exhibit approximately similar PSNR values at different \( k \) values. Thus, we may conclude that varying the \( k \) parameter provides modest to no change on the DCCI performance. On the contrary, the PSNR curves change drastically when varying the parameter \( th \) indicating that this parameter has more influence on the resultant images. For more discussion on the selection of the \( th \) value, refer to [2].

In terms of visual appearance, Fig. 3 reveals that using different parameter values may yield different visual qualities of the magnified images. Five combinations of the \( k \) and \( th \) parameter values used in magnifying the down sampled version of image kodim13 by a factor of 2 are presented in this paper for analytical purposes. Four selected values of both parameters are the combination of the boundary values of the parameters’ ranges in addition to the selected parameter values for \( k \) and \( th \) utilized in the DCCI algorithm. For example, both images produced by using the parameters \( k = 1 \) and \( th = 1 \) and the parameters \( k = 1 \) and \( th = 1.5 \) differed in terms of natural appearance and texture preservation for kodim13. Fig. 3 also shows that the parameter \( th \) has more influence on the performance of the DCCI algorithm; whereas by using a similar \( th \) parameter value regardless of the \( k \) parameter value surprisingly results in similar evaluation measures for PSNR, structural similarity index matrix (SSIM), and feature similarity index (FSIM), and similar visual appearance.

As mentioned earlier, edge detection is performed by using the ratio between the two orthogonal gradients and the parameter \( th \). With the use of the selected values for both \( k \) and \( th \), strong edge pixels were detected by using the DCCI algorithm for the peppers image, as depicted in Fig. 4. Fig. 4(b) shows that many pixels have been mistakenly classified as strong edge pixels in the peppers image and were hence interpolated as edge pixels, thereby leading to degradation of the magnified image especially in texture areas. The DCCI algorithm fails to accurately distinguish strong and non-strong edge pixels. Consequently, it produces directional artifacts in certain texture regions and thus, it is less able to preserve the texture of the magnified image.

### 2.2. Otsu thresholding

Thresholding is a simple and effective tool to separate objects from the background. It has been employed in various computer vision applications including image segmentation and edge detection techniques. Thresholding techniques can roughly be categorized into global and local thresholding methods. In global thresholding, a single threshold is selected for use over

![Fig. 2. PSNR curves for the average PSNR for 24 gray Kodak images produced by the DCCI algorithm by using different \( th \) values that range from 1 to 1.5 at a 0.05 step for every \( k \) in [1,6].](image-url)
Fig. 3. Comparison of the magnified *kodim13* image by using the DCCI algorithm with various values for $k$ and $th$. (a) original image; (b) downscaled image by a factor of 2; (c)–(g) magnified by various $k$ and $th$; (h) magnified by DCCI_Otsu.
the entire image, whereas local thresholds can be computed at every pixel in the image based on local image characteristics in a neighborhood around that pixel [1]. Global thresholding methods are less sensitive to image noise than local thresholding algorithms [17] and are commonly used for their simplicity and low computational complexity. Among the global thresholding techniques, the Otsu thresholding method is one of the most attractive methods for segmenting natural images because it selects a threshold value that maximizes the between-class variance, a well-known measure used in statistical discriminant analysis. The method selects the optimal threshold that gives the best separation between pixel classes and it bases its computation on the image histogram, which can be easily obtained [1]. A brief description of Otsu thresholding as an automatic optimal global thresholding method is presented next.

Supposing that the pixels in a given image represented in $L$ gray levels are on the range $[0, L - 1]$, let the number of pixels with gray level $i$ be $n_i$, and total number of pixels in the gray image be $n$. The probability of occurrence of gray level $i$ is defined as

$$p_i = \frac{n_i}{n} \quad (8)$$

The average gray level of the entire image is computed as

$$\mu_T = \sum_{i=0}^{L-1} ip_i \quad (9)$$

With the use of single threshold $t$, the pixels of a given image can be grouped into two classes $C_1$ and $C_2$ where pixels in $C_1$ are in the range $[0, t]$ and pixels in $C_2$ are in the range $[t + 1, L - 1]$. The probabilities of the two classes are

$$w_1(t) = \sum_{i=0}^{t} p_i \quad (10)$$
$$w_2(t) = \sum_{i=t+1}^{L-1} p_i \quad (11)$$

The mean gray level values of the two classes can be calculated as

$$\mu_1(t) = \sum_{i=0}^{t} ip_i / w_1(t) \quad (12)$$
$$\mu_2(t) = \sum_{i=t+1}^{L-1} ip_i / w_2(t) \quad (13)$$

Otsu showed that the optimal threshold $t^*$ can be determined by using discriminant analysis and by maximizing the between-class variance as in (14).

$$t^* = \text{Arg Max}_{0 \leq t \leq L-1} \{\sigma_B^2(t)\} \quad (14)$$

where the between-class variance $\sigma_B$ is defined as

$$\sigma_B^2(t) = w_1(t)(\mu_1(t) - \mu_T)^2 + w_2(t)(\mu_2(t) - \mu_T)^2 \quad (15)$$
3. Directional cubic convolution interpolation with Otsu thresholding (DCCI_Otsu)

Although Otsu thresholding is a global thresholding method, experimental results indicate that it can efficiently discriminate strong edge pixels in the DCCI algorithm. In addition, its computational simplicity makes it preferable over local thresholding. To improve the DCCI [2] algorithm, Otsu thresholding method is employed to determine an optimal threshold for discriminating strong edge pixels in DCCI_Otsu. Moreover, the algorithm is extended to magnify color images.

To detect strong edge pixels and to decide the edge direction, our proposed method is to use the difference between the two orthogonal gradients in every local window around the missing pixel and compare it to the threshold $t^*$ computed in (14) as illustrated in Fig. 5. Conversely, the ratio of two orthogonal gradients and a preset threshold obtained via training were utilized in [2]. The proposed method does not rely on any parameter setting for edge detection as in [2]; the use of the $k$ parameter in (4) and (5) is eliminated based on the analysis conducted in Section 2.1. Therefore, the exponent operations executed in every local window around non-edge missing pixels in (4) and (5) are canceled which leads to reduce the computational complexity of the proposed algorithm. Fig. 4(c) shows that the proposed detection method can effectively discriminate strong edge pixels when compared to the detection of the original DCCI algorithm. A schematic diagram for edge detection and interpolation in a $7 \times 7$ window around the missing interior pixel in the first stage of the DCCI_Otsu algorithm is shown in Fig. 5; the modification over the original algorithm is highlighted in dashed rectangles. In the second stage, the same flowchart can be used for edge detection in a $5 \times 5$ window. The horizontal gradients $G3$ and the vertical gradients $G4$ replace $G1$ and $G2$, respectively, and the values $p1$ and $p2$ are calculated in vertical and horizontal directions, respectively. Furthermore, the aligned missing pixel can also be computed as in (3).

To magnify color images, the YCbCr color space was adopted to avoid dealing with the high correlation between channels in the RGB color space [15] and to reduce time complexity. The human visual system (HVS) is more sensitive to the luminance channel $Y$ of the image in the YCbCr color space than to the Chrominance channels $Cb$ and $Cr$. Fig. 6 shows that the signal in the $Cb$ and $Cr$ channels are much smoother than the signal in the $Y$ channel. Therefore, a simple interpolation algorithm such as the bilinear or the bicubic can be adopted to interpolate the chrominance channels in order to reduce the computational burden of the proposed algorithm. Meanwhile, the proposed algorithm can solely be applied to the $Y$ channel. The DCCI algorithm is originally designed for grayscale images. To extend the proposed DCCI_Otsu to color images, Fig. 7 shows the steps involved in interpolating color images. The original LR RGB color image is first converted into the YCbCr color space. The proposed algorithm is applied to the luminance channel $Y$ where Otsu threshold is computed using that channel. Chrominance channels $Cb$ and $Cr$ are interpolated by the bicubic interpolation, and the interpolated channels are later converted into the RGB color space. Transformation from the RGB color space to the YCbCr color space and vice versa [18] can be accomplished by using Eqs. (16) and (17).

\[
\begin{align*}
Y &= 0.257R + 0.504G + 0.098B + 16 \\
Cb &= -0.148R - 0.291G + 0.439B + 128 \\
Cr &= 0.439R - 0.368G - 0.071B + 128
\end{align*}
\]

(16)
\[ R = 1.164(Y - 16) + 1.596(Cr - 128) \]
\[ G = 1.164(Y - 16) - 0.813(Cr - 128) - 0.391(Cb - 128) \]
\[ B = 1.164(Y - 16) + 2.018(Cb - 128) \]

(17)

4. Experimental results and discussion

All the experiments were conducted by using MATLAB software under Windows platform with an Intel Core i5 computer with a processing speed of 3.1 GHz and 2 GB of RAM. The subjective and objective results obtained by the DCCI_Otsu algorithm are compared with other methods in the literature, including bilinear interpolation and bicubic interpolation, in addition to six recently developed algorithms, namely, DCCI [2], DFDF [6], ICBI [10], CSEAI [8], IINFO [11], and DIDWI [14]. The DCCI_Otsu algorithm was evaluated quantitatively by using a set of 116 color images and their corresponding gray level images. These images were downloaded from different sources [19–23] and cover a wide range of content including different types of edge and texture structures. Our dataset include natural images [19–21], standard images [23] and some plain texture images [22]. Images in the dataset have been used as test images in the evaluation of many magnification algorithms [2,4–6,12,14] in the literature.

4.1. Qualitative analysis

A group of images were selected as test images, including kodim22, statues, D75, musician, peppers, and cameraman, to evaluate the subjective performance of the DCCI_Otsu algorithm and to compare it with the other magnification methods. Kodim22 is down-sampled with a factor of 2 by using direct down-sampling and then magnified by the proposed
Fig. 8. Comparison of the *kodim22* image at 2× magnification: (a) original image; (b) bilinear; (c) bicubic; (d) DIDWI; (e) DFDF; (f) ICBI; (g) DCCI; (h) IINFO; (i) CSEAI; (j) proposed DCCI_Otsu.
DCCI_Otsu algorithm and the other compared methods. Fig. 8 depicts the magnified images of kodim22. The DCCI algorithm can preserve areas of edges such as the roof of the house in kodim22, but fails to preserve areas of texture in the same image such as the textures in the trees and the grass areas where directional artifacts are noticeable in these areas because texture pixels are interpolated as edge pixels. The CSEAI suffers directional artifacts in texture areas in the kodim22 image however it shows ability in preserving edge structures. The bilinear and bicubic methods efficiently preserve the texture areas but exhibit blurry and server-blocking artifacts around the diagonal edges. The IINFO algorithm shows lower ability to preserve the texture of the trees than the bilinear and the bicubic where pixilation effect can be observed. Meanwhile, server-blocking artifacts around diagonal edges can also be noticed. The DFDF, DIDWI, and ICBI methods are better able to preserve the diagonal edges in comparison with the bilinear and the bicubic methods but still suffer some blocking artifacts when compared with DCCI and DCCI_Otsu. However, the DFDF and DIDWI methods preserve texture better than the ICBI, CSEAI and DCCI algorithms. Moreover, the proposed DCCI_Otsu provides the best image quality among the depicted images in terms of edge smoothness and detail preservation; DCCI_Otsu is reasonably able to preserve the edge and the texture areas simultaneously.

Fig. 9 reveals that edge preservation by the proposed DCCI_Otsu algorithm is comparable to that of the DCCI algorithm; both algorithms exhibit the best ability to preserve edges among all the competing methods followed by the CSEAI and the DFDF algorithms. Moreover, Fig. 10, which depicts portions of statues, D75 and kodim22 images, shows that the proposed DCCI_Otsu algorithm preserves texture structures in kodim22 and statues images better than the DCCI algorithm. Meanwhile, the proposed algorithm shows comparable appearance to the DCCI in preserving texture structures that contain lines and curves as in D75 image. Fig. 11 shows the interpolated portions for a part of the pepper image magnified by a factor of 4 after direct down-sampling. DCCI_Otsu provides better visual appearance than DCCI in the body of the large green pepper, which shows a more natural-looking image compared with DCCI, which exhibits directional artifacts. The DCCI_Otsu image is comparable to the DCCI image in terms of preserving edges; conversely, the bilinear, the bicubic and the IINFO images once again suffer blocking artifacts along the diagonal edge structure. Moreover, pixilation effect is obvious in the body of the green pepper in the IINFO resultant image. Similarly, water-coloring effect is evident in the same area of the CSEAI resultant image. The DFDF, DIDWI, and ICBI methods suffer less blocking artifacts along edges compared with the bilinear and bicubic methods, but the latter two show a more natural appearance in the pepper body compared with the DCCI. Moreover, the ICBI algorithm exhibits some ringing artifacts along the right vertical edge of the pepper.

For a large magnification factor, a part of the cameraman image is magnified by a factor of 8. In Fig. 12, the bilinear and the bicubic interpolation methods repeatedly fail in edge regions, whereas the DIDWI performs better; conversely, the DIDWI is outperformed in texture areas. The DCCI and DCCI_Otsu show comparable appearance in edge regions and the best appearance among the other methods, but DCCI_Otsu outperforms DCCI in the texture area. The DCCI_Otsu also beats the CSEAI in the texture area where directional artifacts can be observed. Besides, some blocking artifacts along the left leg of the camera stand can also be noticed the CSEAI resultant image. The IINFO once again shows some pixilation in the texture area, and shows less ability to preserve edges among the other algorithms as can be noticed in the figure. The ICBI shows ringing artifacts along edges, whereas DFDF presents dashed artifacts; conversely, both techniques show ability to preserve the texture area. To further show the performance of the proposed algorithm, the magnified images of additional ten color test LR images obtained by direct down-sampling are provided as Supplementary material. The results indicate that the proposed algorithm outperforms its peers both qualitatively and quantitatively.

4.2. Quantitative analysis

Various computational models have been used to measure the image quality in accordance with subjective assessments. In this paper, three objective measures—PSNR, SSIM, and FSIM—are used for evaluation. The PSNR is a pixel-based measure that has been commonly used for evaluating the quality of the magnified images by various magnification techniques [2,10,11,14]. The PSNR is computed based on the mean square error (MSE). The MSE is the mean of the squared differences between every pixel in the original image and its corresponding pixel in the magnified image as in (18) where \( f_1(x) \) and \( f_2(x) \) are pixels of the original image \( f_1 \) and the magnified image \( f_2 \) of size \( H \times K \) at the corresponding location \( x \), respectively. The PSNR between the original image and the magnified image is given by (19) and measured in decibels (dB).

\[
\text{MSE} = \frac{1}{HK} \sum_{i=1}^{H} \sum_{j=1}^{K} (f_1(x) - f_2(x))^2,
\]

\[
\text{PSNR} = 20 \times \log_{10} \left( \frac{255}{\sqrt{\text{MSE}}} \right),
\]

Higher PSNR values correspond to lower MSE and thus imply less distortion and indicate better quality of the magnified image.

The SSIM is a structure-based method that measures the quality of images based on structural content between two images, the original \( f_1 \) and the magnified \( f_2 \) [24]. SSIM measure is based on the assumption that the HVS is highly adapted to extract structural information from the viewing scene. It expresses the similarity in terms of luminance, contrast and structure. Luminance is estimated by the mean intensity, contrast is estimated by the standard deviation and structure is estimated by the covariance of the luminance and contrast components.
estimated by the correlation. The similarity in local windows is computed as in (20) by combining local statistics of averages, standard deviations and correlations from local windows in the original and the magnified image in order to compose an SSIM index map of the magnified image. Then, a single quality measure is pooled from the map by computing the mean of the SSIM index map as in (21).

\[
\text{SSIM}(y, z) = \frac{(2\mu_y\mu_z + c_1)(2\sigma_{yz} + c_1)}{\left(\mu_y^2 + \mu_z^2 + c_1\right)\left(\sigma_y^2 + \sigma_z^2 + c_2\right)}
\]  

(20)
In (20), \(y\) and \(z\) are two corresponding windows in the original and the magnified images, \(n\) is the number of pixels in each window, \(\mu_y\) and \(\mu_z\) are the means of pixels in \(y\) and \(z\); \(\sigma_y\) and \(\sigma_z\) are the standard deviations; \(\sigma_{yz}\) is the correlation between pixels in \(y\) and \(z\); \(c_1\) and \(c_2\) are two variables to stabilize the division with weak denominator.

\[
\text{MSSIM}(f_1, f_2) = \frac{1}{m} \sum_{j=1}^{m} \text{SSIM}(y_j, z_j)
\]

In (21), \(m\) is the number of local windows in the image. MSSIM value is in the range \([0, 1]\) where higher values denote better structural similarity and thus better quality.

The FSIM measure \([25]\) combines two features to characterize the image local quality. These features are the phase congruency (PC), which is a primary feature with dimensionless measure of the significance of a local structure and a secondary feature, the image gradient magnitude (GM). The FSIM is based on the fact that the HVS understands an image mainly according to its low-level features. Let \(PC_1\) and \(PC_2\) are the phase congruency maps extracted from \(f_1\) and \(f_2\), respectively while \(G_1\) and \(G_2\) are the gradient maps. The similarity measure between \(PC_1\) and \(PC_2\) can be computed as in (22).

\[
S_{PC}(x) = \frac{2PC_1(x) \cdot PC_2(x) + T_1}{PC_1^2(x) + PC_2^2(x) + T_1}
\]

Similarly, the GM values \(G_1\) and \(G_2\) are compared. The similarity measure is defined as in (23) where \(T_1\) is a positive constant to increase the stability of \(S_{PC}\) while \(T_2\) is a positive constant to increase the stability of \(SG\) and both \(T_1\) and \(T_2\) depends on the dynamic range of PC values and the GM values, respectively.

Fig. 10. Comparison of portions of statues, D75 and kodim22 images magnified by the original DCCI and the proposed DCCI_Otsu by \(2\times\) after direct downsampling: (a) original image; (b) DCCI; (c) proposed DCCI_Otsu.
Fig. 11. Comparison of the portion of peppers image at 4× magnification: (a) original image; (b) downsampled image; (c) bilinear; (d) bicubic; (e) DIDWI; (f) DFDF; (g) ICBI; (h) DCCI; (i) IINFO; (j) CSEAI; (k) proposed DCCI_Otsu.
Fig. 12. Comparison of the portion of *cameraman* image at $8 \times$ magnification: (a) original image; (b) bilinear; (c) bicubic; (d) DIDWI; (e) DFDF; (f) ICBI; (g) DCCI; (h) IIINFO; (i) CSEAI; (j) proposed DCCI_Otsu.
In order to get the similarity $S_{l}(x)$ at each location in the images $f_{1}(x)$ and $f_{2}(x)$, both $S_{PC}(x)$ and $S_{C}(x)$ are combined as in (24) where $\alpha$ and $\beta$ are parameters used to adjust the relative importance of PC and GM features and both have been set to be 1.

$$S_{l}(x) = [S_{PC}(x)]^{\alpha} \cdot [S_{C}(x)]^{\beta}$$

As different locations in the image have different contributions to the human visual system perception and as the human visual cortex is sensitive to phase congruent structures [25], the PC value at a location can reflect the perceptual significance of that location. Therefore, the importance of $S_{l}$ in the overall similarity between $f_{1}$ and $f_{2}$ is weighted using $P_{C_{m}}$ where $P_{C_{m}}$ is computed as in (25) and thus, the FSIM value can be obtained as in (26). The FSIM value is in the range [0,1] where higher values denote better quality and consistency with the subjective evaluations.

$$P_{C_{m}}(x) = \max(P_{C_{1}}(x) \cdot P_{C_{2}}(x))$$

$$FSIM = \frac{\sum_{x \in A} S_{l}(x) \cdot P_{C_{m}}(x)}{\sum_{x \in A} P_{C_{m}}(x)}$$

Apart from evaluating the proposed algorithm in terms of the magnified image quality, the proposed algorithm is examined in terms of computational time. The computational time for all algorithms under testing is recorded and used to compare the complexity of those algorithms. Shorter computational time indicates simpler implementation.

First, we start our quantitative assessment by comparing the use of automatic Otsu threshold and the use of fixed threshold values in the proposed algorithm. Four color images of different types were selected namely, *kodim05* image that mainly contains small and large scale edge structures, *kodim13* that mainly contains texture structures and two images that contain...
both edge and texture structures (i.e. cameraman and kodim01). The corresponding LR images of the selected HR images were produced using direct down-sampling by a factor of 2. Using the proposed algorithm with different threshold values ranging from 30 up to 220 by step 10, the LR images are magnified and then the PSNR values were computed at every threshold value. Fig. 13 shows the PSNR results of each image at every fixed threshold. The PSNR results of each image obtained by using the automatic Otsu thresholding method are marked in the figure by dashed vertical lines. Exact Otsu threshold value for each LR image and the PSNR of each magnified corresponding image using the proposed DCCI_Otsu are also presented in Table 1. It can be observed from Fig. 13 that the PSNR values obtained for each magnified image using the proposed algorithm fall within the best PSNR range that can be obtained using fixed threshold values. For instance, the best PSNR range (i.e. 31.181 to 31.21 with values slightly increase inside the range and then starts to decrease slowly afterwards) can be obtained at the threshold values that range from 60 to 100 for kodim05 image. Meanwhile, using the proposed algorithm with automatic Otsu threshold (i.e. 94) can yield a magnified image with high PSNR value (i.e. 31.219). It can also be observed that for different types of images, a reasonably high PSNR values that are located within the best PSNR range can be obtained by employing Otsu threshold regardless of image type and content.

Second, the quantitative comparisons of the entire dataset described earlier are presented. The HR images of the LR images’ versions of the 116 color test images are also reconstructed by using the competing state-of-art methods mentioned earlier. Experimental results of the quantitative analysis are given in Tables 2 and 3 for the 116 gray images and the corresponding 116 color images, respectively, where the comparison is conducted based on average values of PSNR, SSIM, FSIM, and execution time. The best results are presented in bold text. Tables 2 and 3 reveal that the proposed DCCI_Otsu achieves the highest PSNR, SSIM, and FSIM among the methods, thereby confirming that the proposed algorithm successfully outperforms the competing methods.

In terms of processing time, the proposed DCCI_Otsu consumes lower processing time than the original DCCI due to the elimination of the exponent operations in (4) and (5). The proposed DCCI_Otsu pre-calculates the Otsu threshold from the LR image once and then uses it in the detection scheme in every local window. On the other hand, in the DCCI, the exponent operations in (4) and (5) have to be executed in every local window in non-edge areas in the HR image. As mentioned earlier, Otsu thresholding is known for its computational simplicity. For instance, computing Otsu threshold of the gray cameraman image merely requires 0.016 s on the platform where all experiments were conducted. Meanwhile, the proposed DCCI_Otsu requires 0.714 s to magnify the LR gray cameraman image by $2^2$. The proposed DCCI_Otsu also consumes much lower time

<table>
<thead>
<tr>
<th>Image</th>
<th>Kodim01</th>
<th>Kodim13</th>
<th>Kodim05</th>
<th>Cameraman</th>
</tr>
</thead>
<tbody>
<tr>
<td>Otsu threshold ($t^*$)</td>
<td>96</td>
<td>128</td>
<td>94</td>
<td>88</td>
</tr>
<tr>
<td>PSNR</td>
<td>29.764</td>
<td>27.672</td>
<td>31.219</td>
<td>30.704</td>
</tr>
</tbody>
</table>

Table 2
Comparison of average quantitative analysis for 116 gray scale images at $2^2$ magnification.

<table>
<thead>
<tr>
<th>Method</th>
<th>PSNR</th>
<th>SSIM</th>
<th>FSIM</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bilinear</td>
<td>26.930</td>
<td>0.7907</td>
<td>0.9229</td>
<td>0.011</td>
</tr>
<tr>
<td>Bicubic</td>
<td>26.757</td>
<td>0.7976</td>
<td>0.9247</td>
<td>0.014</td>
</tr>
<tr>
<td>DFDF [6]</td>
<td>28.979</td>
<td>0.8477</td>
<td>0.9512</td>
<td>13.183</td>
</tr>
<tr>
<td>ICBI [10]</td>
<td>28.708</td>
<td>0.8439</td>
<td>0.9495</td>
<td>97.945</td>
</tr>
<tr>
<td>DIDWI [14]</td>
<td>29.048</td>
<td>0.8480</td>
<td>0.9496</td>
<td>0.021</td>
</tr>
<tr>
<td>DCCI [2]</td>
<td>29.075</td>
<td>0.8491</td>
<td>0.9520</td>
<td>3.704</td>
</tr>
<tr>
<td>IINFO [11]</td>
<td>27.052</td>
<td>0.7961</td>
<td>0.9343</td>
<td>1.999</td>
</tr>
<tr>
<td>CSEAI [8]</td>
<td>28.611</td>
<td>0.8394</td>
<td>0.9499</td>
<td>0.073</td>
</tr>
<tr>
<td>DCCI_Otsu</td>
<td><strong>29.250</strong></td>
<td><strong>0.8546</strong></td>
<td><strong>0.9529</strong></td>
<td><strong>2.790</strong></td>
</tr>
</tbody>
</table>

Table 3
Comparison of average quantitative analysis for 116 color images at $2^2$ magnification.

<table>
<thead>
<tr>
<th>Method</th>
<th>PSNR</th>
<th>SSIM</th>
<th>FSIM</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bilinear</td>
<td>31.512</td>
<td>0.7770</td>
<td>0.9218</td>
<td>0.014</td>
</tr>
<tr>
<td>Bicubic</td>
<td>31.316</td>
<td>0.7834</td>
<td>0.9229</td>
<td>0.015</td>
</tr>
<tr>
<td>DFDF [6]</td>
<td>33.098</td>
<td>0.8227</td>
<td>0.9449</td>
<td>13.589</td>
</tr>
<tr>
<td>ICBI [7]</td>
<td>32.870</td>
<td>0.8210</td>
<td>0.9441</td>
<td>100.796</td>
</tr>
<tr>
<td>DIDWI [8]</td>
<td>33.160</td>
<td>0.8223</td>
<td>0.9435</td>
<td>0.061</td>
</tr>
<tr>
<td>DCCI [2]</td>
<td>33.176</td>
<td>0.8246</td>
<td>0.9459</td>
<td>4.266</td>
</tr>
<tr>
<td>IINFO [11]</td>
<td>31.394</td>
<td>0.7750</td>
<td>0.9302</td>
<td>2.059</td>
</tr>
<tr>
<td>CSEAI [8]</td>
<td>32.779</td>
<td>0.8141</td>
<td>0.9437</td>
<td>0.118</td>
</tr>
<tr>
<td>DCCI_Otsu</td>
<td><strong>33.340</strong></td>
<td><strong>0.8294</strong></td>
<td><strong>0.9467</strong></td>
<td><strong>2.888</strong></td>
</tr>
</tbody>
</table>
than DFDF and ICBI. The bilinear, bicubic, DIDWI, and CSEAI methods exhibit lower execution time among other methods but at the expense of poor visual quality in terms of edge preservation for the first three methods and in terms of texture preservation for the last method. The INFO also exhibits lower execution time than the proposed algorithm but at the cost of both edge and texture preservation.

5. Conclusion

In this study, an improved image magnification algorithm is proposed to alleviate the drawbacks of the DCCI algorithm. The drawbacks can be summarized as: (a) directional artifacts in certain smooth and texture areas which are caused by the incorrect detection and interpolation of pixels in those areas as edge pixels, (b) relying on training to attain the threshold \( t \) in the detection scheme, and (c) using the \( k \) parameter that depends on training to adjust the interpolation coefficients in texture areas. To overcome these drawbacks, this paper proposed a new edge detection method based on the difference between the two orthogonal gradients in local windows around missing pixels. Instead of using a fixed threshold parameter obtained by training, Otsu automatic optimum thresholding is used in the proposed detection method. In texture areas, missing pixels are estimated by combining the two interpolated cubic convolution values and the two orthogonal directional gradients where the exponent parameters of these directional gradients employed in the DCCI are eliminated based on empirical analysis. Compared with several stat-of-the-art magnification methods, extensive simulation results reveal that the proposed DCCI_Otsu algorithm performs excellently in preserving texture regions of both gray and color magnified image due to the correct discrimination of pixels in those regions. Moreover, the proposed algorithm shows good capability of preserving edge structures in terms of reducing blocking and ringing artifacts along edges. The resultant magnified images of the proposed algorithm appear to be more natural and are more visually appealing. Results also show that the proposed algorithm has achieved the highest PSNR, SSIM, and FSIM measures among all compared methods including the DCCI. Furthermore, it consumes less computational time in comparison with the original DCCI algorithm.

Acknowledgement

The authors express their sincere thanks to the anonymous reviewers for their significant contribution to the improvement of the final paper. This study was partially supported by the Research University Grant for Individual (RUI) of Universiti Sains Malaysia, Malaysia entitled “Development of an Intelligent Auto-Immune Diseases Diagnostic System by Classification of Hep_2 Immunofluorescence Patterns.”

Appendix A. Supplementary material

Supplementary data associated with this article can be found, in the online version, at http://dx.doi.org/10.1016/j.compeleceng.2015.03.025.

References

Suheir M. ElBayoumi Harb received her B.Sc. degree in Computer Science from Al-Quds University, Jerusalem, Palestine in 1996 and her M.Sc degree in Information Systems with First Class Honors from CDM, DePual University, Chicago, USA in 2004. She is currently a Ph.D. candidate at the School of Electrical and Electronic Engineering, USM in Malaysia. Her research interests include image processing.

Nor Ashidi Mat Isa received his B. Eng. degree in Electrical and Electronic Engineering with First Class Honors and his Ph.D. degree in Electronic Engineering from USM in 1999 and 2003 respectively. Currently, he serves as an Associate Professor at the School of Electrical and Electronic Engineering, USM. His research interests include image processing, neural network and intelligent systems.

Samy A. Salamah received his B. Eng. and M. Eng. degrees in computer engineering (majoring in computer security and data cryptography using VHDL utilizing FPGA) from Arab Academy for Science and Technology, Alexandria, Egypt in 1996 and 2003, respectively. Currently, he serves as Assistant Professor and lectures at Palestine Technical College, Gaza, Palestine. His current research is in image processing.