Hindawi Publishing Corporation
EURASIP Journal on Advances in Signal Processing
Volume 2009, Article ID 372180, 11 pages
doi:10.1155/2009/372180

Research Article

Two-Stage Interpolation Algorithm Based on Fuzzy Logics and Edges Features for Image Zooming

Hsiang-Chieh Chen[1] **and Wen-June Wang**[1, 2]

[1] *Department of Electrical Engineering, National Central University, Taoyuan 32001, Taiwan*
[2] *Department of Electrical Engineering, National Taipei University of Technology, Taipei 10608, Taiwan*

Correspondence should be addressed to Wen-June Wang, wjwang@ee.ncu.edu.tw

Received 18 February 2009; Revised 6 June 2009; Accepted 26 August 2009

Recommended by William Sandham

This work presents an innovative two-stage interpolation algorithm for image resolution enhancement and zooming applications. The desired high-resolution images are obtained via two interpolative stages. In the first stage, aligned pixels are first estimated using a fuzzy inference system, whose critical parameters are optimized by particle swarm intelligence. In the second stage, interior pixels are then restored by utilizing the edge properties of nearby pixels. From experimental results, numerical comparison confirms the superiority of the proposed interpolation algorithm over other existing methods. Furthermore, visual illustrations including zoomed parts and error maps demonstrate the significant improvement of the proposed method, particularly in the regions that contain many local edges and sharp details.

1. Introduction

In fundamental image processing techniques, image interpolation methods are extensively researched because of their wide use of digital imaging applications, such as consumer electronics, multimedia transmission, remote sensing, and medical imaging. The image interpolation adopted in image enlargement and reconstruction commonly estimates the pixel value of a specified position. As presented in many studies [1–7], image interpolation was used to produce a high-resolution (HR) image from its associated low-resolution (LR) version. Numerous image interpolation methods that calculate the interpolated value as a weighted sum of neighboring pixels have been proposed in early years [8, 9]. The nearest neighbor method is simple to implement, but it suffers much from blocking effect. Another well-known method is the bilinear interpolation; however, it conducts blurry edges or zigzagging structures in HR images while there are sharp details or discontinuities in LR images. The visual quality of enlarged images can be improved by cubic splines interpolation [9] and cubic convolution [10–12]. All these popular methods only consider spatial distances between image pixels and thus often lead unsatisfactory performance in interpolated results.

To solve the above problem, various interpolation approaches have been introduced to obtain qualified HR images via considering local edge properties [1–4, 13–16]. Moreover, some nonlinear methods have attempted to enlarge images using intelligent schemes, such as optimal recovery [7], neural networks [17, 18], fuzzy logics [19], support vector machines [20], and vector quantization [21, 22]. Another subfield of image interpolation is performed in wavelet domain, but the noninteger magnification factors cannot be used unfortunately [23, 24]. Several state-of-the-art methods have recently been presented and achieved acceptable performance in HR images. An adaptive interpolation approach that integrates piecewise autoregressive models into image blocks is presented in [6]. In [25], the nonlinear filters obtained from an adaptive procedure are applied in sharp regions to avoid blurring effect, while the linear filters are employed in smooth regions. Yoo derives a closed-form formulation using least-square techniques to adapt linear interpolation [26]. Based on interpolation error theorem, the error-amended sharp edge is introduced in [27]. The image enlargement approach presented in [28] adopts a codebook consisting of low- and high-frequency components of an original image.

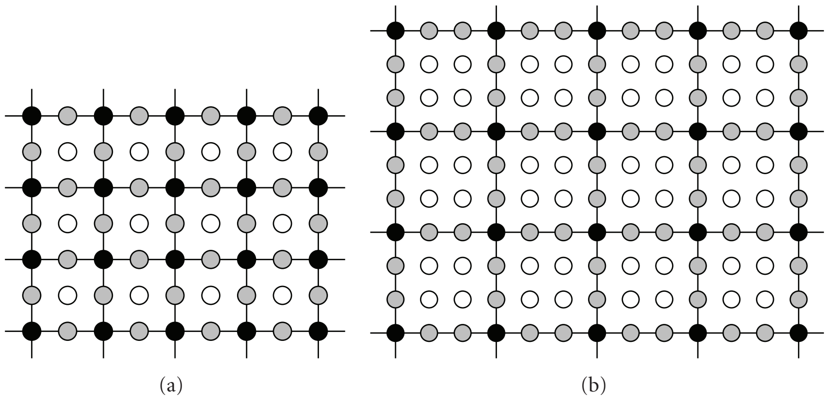


FIGURE 1: Production of LR images using subsampling: (a) $\kappa = 2$, and (b) $\kappa = 3$.

Since human visual system is strongly sensitive to the variations in intensity, an effective scheme that deals with spatial sharpness and edges is very useful to improve the quality of enlarged HR images. This study proposes an efficient interpolation algorithm comprising two main stages. In the first stage, an intelligent framework that combines fuzzy inference system and particle swarm optimization is adopted to restore the aligned pixels. In the second stage, the interior pixels are obtained via extending the edge properties of their neighbors. Experimental results demonstrate that the proposed method certainly achieves good capacity for enhancing spatial resolution of images.

This paper is organized as follows. Section 2 introduces the proposed two-stage interpolation algorithm in detail. Based on particle swarm optimization technique, Section 3 presents a learning procedure to determine the critical parameters of the presented fuzzy inference system. Section 4 demonstrates the experimental results presented in numerical comparisons and visual illustrations. Final, a short conclusion is made in Section 5.

2. Two-Stage Interpolation Algorithm

2.1. Basic Concepts. Assume that a given LR image I_{LR} of size $W \times H$ is to be enlarged into an HR image I_{HR} of size $\kappa W \times \kappa H$, where κ is a predefined magnification factor (MF). The LR image is generally treated as a subsampled version from its associated HR image. Figure 1 illustrates two standard examples of the production of LR images by subsampling for MFs $\kappa = 2$ and $\kappa = 3$, in which the black dots represent the preserved pixels in the LR image and the other dots represent lost pixels. The HR image is reconstructed in two interpolative stages: (1) the *aligned pixels* (gray dots in Figure 1) along each dimension are estimated, and (2) the *interior pixels* (white dots in Figure 1) are estimated. The following two subsections present the proposed two-stage interpolation algorithm.

2.2. The First Stage: Estimating Aligned Pixels. Figures 2(a) and 2(b) depict the aligned pixels located at $(x + u, y)$ and $(x, y + v)$ in unit division $[x, x + 1] \times [y, y + 1]$, respectively. Take Figure 2(a) as an example to describe the process of estimating aligned pixels. Let $f_{x,y} \in I_{\mathrm{LR}}$ denote the pixel value at position (x, y), where $x = 1, 2, \ldots, W$, and $y = 1, 2, \ldots, H$. The linear interpolation method yields the interpolated value of a specific point $(x + u, y)$ as

$$\tilde{f}_{x+u,y} = (1 - u) f_{x,y} + u\ f_{x+1,y}, \tag{1}$$

where $u \in (0, 1)$ is the normalized Euclidean distance. For an arbitrary MF κ, $u = \varepsilon/\kappa$ for $\varepsilon = 1, 2, \ldots, \kappa - 1$. However, the sharp details in reconstructed HR images are usually destroyed because the linear interpolation greatly suffers from blurring effects. Figure 3 demonstrates a one-dimensional example of a blurred edge when linear interpolation is adopted, in which the original HR data in (a) are first subsampled into the decimated LR data in (b) and are then reconstructed to meet the original resolution in (c).

To introduce the novel contribution of this work, Figure 4 plots the pixels of interest and presents the important notation. As indicated in Figure 4, the interpolated result $\tilde{f}_{x+u}$ does not accurately represent the original value $f_{x+\mathrm{u}}$, as an interpolation error of $e_{x+u} = |f_{x+u} - \tilde{f}_{x+u}|$ exists. Intuitively, reducing the weight of sample f_x (located on the sharp side in Figure 4) and increasing the weight of sample f_{x+1} (located on the smooth side in Figure 4) can effectively reduce this undesired interpolation error. Restated, the gradient measurement of each sample of interest helps in accurately estimating the interpolated result. Consequently, this subsection introduces a novel scheme that utilizes the gradients of sampling nodes x and $(x + 1)$ to obtain new weights of samples f_x and f_{x+1}. However, the method for calculating (or modifying) weights is just a linguistic manner since neither a deterministic approach nor a mathematical derivation has been given. Fuzzy techniques have been extensively adopted in identifying unknown models or systems. Therefore, this study is the first to develop a fuzzy inference system (FIS) for deriving the weight of each neighboring sample.

Let $\aleph_4$ denote the four corner pixels of the unit division $[x, x+1] \times [y, y+1]$, such that $\aleph_4 = \{(i, j) \mid i \in \{x, x+1\};\ j \in \{y, y+1\}\}$. Let $g_d[\mathbf{n}]$, which is called the gradient component along the dth dimension of a specified pixel $\mathbf{n}$, be the premise

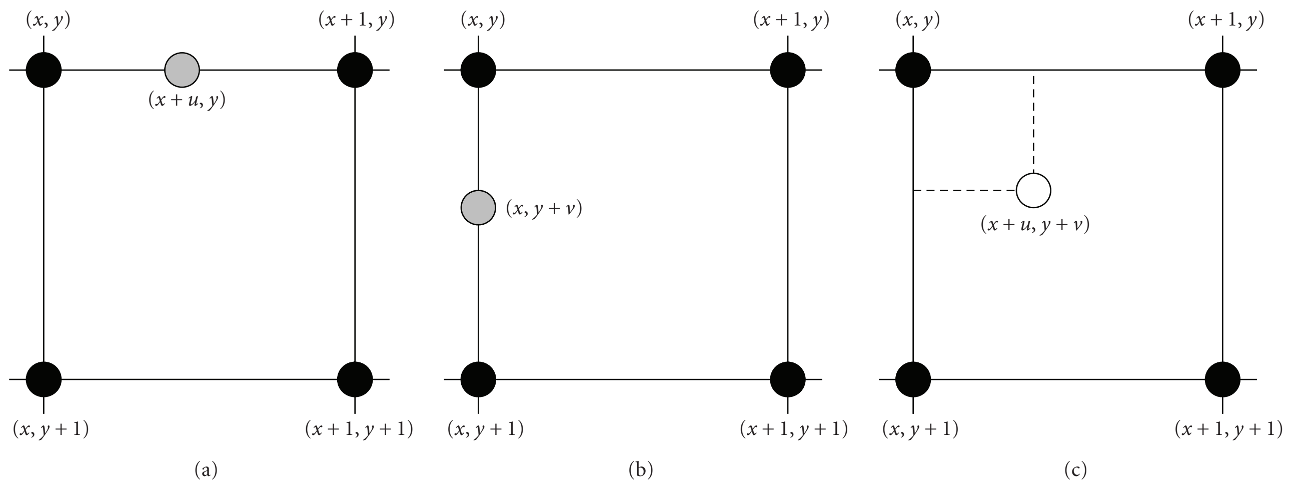


Figure 2: Three cases of interpolation: (a) aligned pixel on x-axis, (b) aligned pixel on y-axis, and (c) interior pixel.

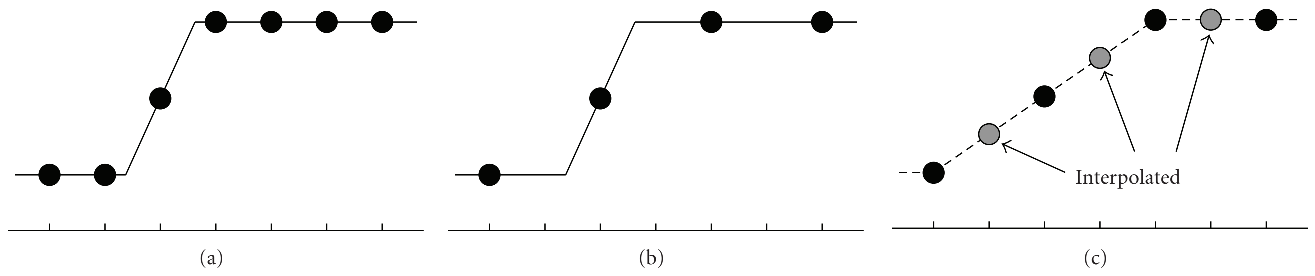


Figure 3: Blurring effect of linear interpolation in one-dimensional case: (a) original HR data points, (b) LR data by subsampling, and (c) recovered data by linear interpolation.

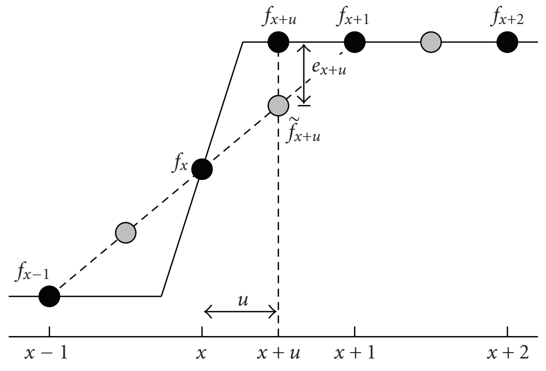


Figure 4: One-dimensional example of the conventional linear interpolation.

variable of the proposed FIS:

$$g_d[\mathbf{n}] = \begin{cases} \left| f_{i+1,j} - f_{i-1,j} \right|, & \text{if } d = 1, \\ \left| f_{i,j+1} - f_{i,j-1} \right|, & \text{if } d = 2, \end{cases} \tag{2}$$

where $\mathbf{n} = (i, j)$ is selected from the corner pixels $\aleph_4$. Define a weighting factor $\tau_d[\mathbf{n}]$ as the consequent variable; this factor will be used to determine the weights of the considered pixels. Now, the modification rules are first constructed in linguistic terms. For example, **IF** *gradient is large* (*or small*), **THEN** *weighting factor is small* (*or large*).

Hence, the rth fuzzy rule is written in the following form:

$$\textbf{Rule } r: \text{ IF } g_d[\mathbf{n}] \text{ is } A_r, \text{THEN } \tau_d[\mathbf{n}] \text{ is } B_{\Gamma-r+1}, \tag{3}$$

where $r = 1, 2, \ldots, \Gamma$, and Γ is the total number of fuzzy rules, and A_r and $B_{\Gamma-r+1}$ are selected from $\{A_1, A_2, \ldots, A_\Gamma\}$ and $\{B_1, B_2, \ldots, B_\Gamma\}$, respectively (Figure 5). When $\tilde{g}_d[\mathbf{n}]$ is inputted to the FIS, the derived output $\tilde{\tau}_d[\mathbf{n}]$ is calculated via fuzzy inference and defuzzification as follows:

$$\tilde{\tau}_d[\mathbf{n}] = \frac{\sum \tau_d[\mathbf{n}] \cdot B'(\tau_d[\mathbf{n}])}{\sum B'(\tau_d[\mathbf{n}])},$$
$$B'(\tau_d[\mathbf{n}]) = \max_{\mu=1,2,\ldots,\Gamma} \left\{ \min\left[A_\mu(\tilde{\tau}_d[\mathbf{n}]), B_{\Gamma-\mu+1}(\tau_d[\mathbf{n}]) \right] \right\}. \tag{4}$$

As illustrated in Figure 2(a), the interpolated value of the aligned pixel $(x+u, y)$ is calculated by

$$\tilde{f}_{x+u,y} = \frac{\hat{\tau}_1[\mathbf{n}_1]}{\omega_1}(1-u) f_{x,y} + \frac{\hat{\tau}_1[\mathbf{n}_2]}{\omega_1} u f_{x+1,y}, \tag{5}$$

where $\omega_1 = \hat{\tau}_1[\mathbf{n}_1](1-u) + \hat{\tau}_1[\mathbf{n}_2]u$, and $\mathbf{n}_1 = (x, y)$ and $\mathbf{n}_2 = (x+1, y)$ are the neighboring pixels. Similarly, as shown

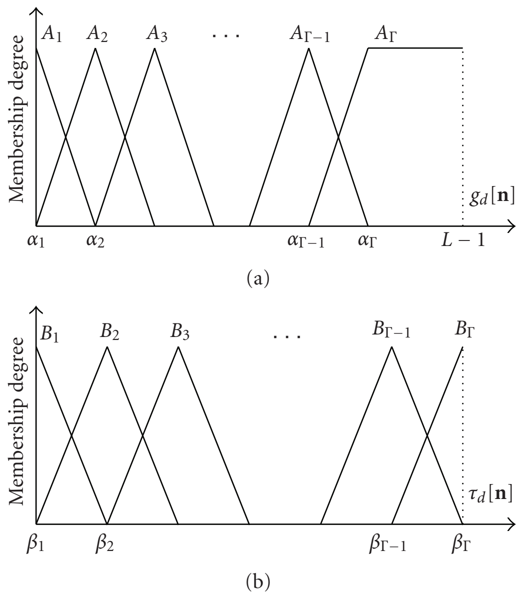


FIGURE 5: Partitioned fuzzy sets of: (a) premise variable, and (b) consequent variable.

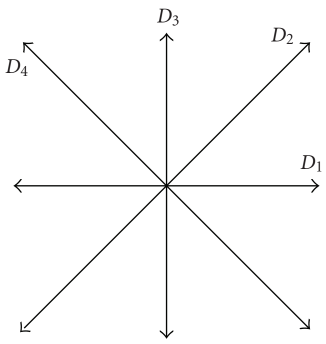


FIGURE 6: Four directions of an edge.

in Figure 2(b), the interpolated value of the pixel $(x, y+v)$ is calculated by

$$\tilde{f}_{x,y+v} = \frac{\hat{\tau}_2[\mathbf{n}_1]}{\omega_2}(1-v)f_{x,y} + \frac{\hat{\tau}_2[\mathbf{n}_3]}{\omega_2}vf_{x,y+1}, \tag{6}$$

where $\omega_2 = \hat{\tau}_2[\mathbf{n}_1](1-v) + \hat{\tau}_2[\mathbf{n}_3]v$, and $\mathbf{n}_3 = (x, y+1)$. Therefore, the aligned pixels that were lost in LR images are restored using (5) and (6).

In brief, the first interpolative stage is proposed for estimating the aligned pixels from their neighbors that were preserved in LR images. By combining gradient components and fuzzy theory, the accuracy of interpolated values is certainly improved.

2.3. The Second Stage: Estimating Interior Pixels. The second stage presented in this subsection is employed to estimate the values of interior pixels that are marked as white dots in Figure 1. Since edge features are critical to representing the local characteristics of an image, the interior pixels are interpolated from their neighboring pixels by considering the edge properties. Here, the neighboring pixels may be either the preserved pixels in the given LR image (black dots in Figure 1) or the aligned pixels that were estimated in the first stage (gray dots in Figure 1). Figure 2(c) presents an interior pixel located in the unit division $[x, x+1] \times [y, y+1]$, in which $u = \varepsilon_1/\kappa$ and $v = \varepsilon_2/\kappa$ for $\varepsilon_1, \varepsilon_2 = 1, 2, \ldots, \kappa - 1$. The edge strengths and orientations of all pixels in $\aleph_4$ must be obtained first. Thus, the edge components of a specified pixel $\mathbf{n} \in \aleph_4$ are calculated using

$$\begin{aligned} E_x(\mathbf{n}) &= -f_{i-1,j-1} - 2f_{i-1,j} - f_{i-1,j+1} \\ &\quad + f_{i+1,j-1} + 2f_{i+1,j} + f_{i+1,j+1}, \\ E_y(\mathbf{n}) &= -f_{i-1,j-1} - 2f_{i,j-1} - f_{i+1,j-1} \\ &\quad + f_{i-1,j+1} + 2f_{i,j+1} + f_{i+1,j+1}. \end{aligned} \tag{7}$$

Accordingly, the edge strength and orientation of $\mathbf{n}$ are obtained as follows:

$$\begin{aligned} E(\mathbf{n}) &= |E_x(\mathbf{n})| + \left|E_y(\mathbf{n})\right|, \\ \theta(\mathbf{n}) &= \tan^{-1}\left[\frac{E_x(\mathbf{n})}{E_y(\mathbf{n})}\right]. \end{aligned} \tag{8}$$

For an interior pixel $(x+u, y+v)$, the edge strength caused by a specific pixel $\mathbf{n}$ is defined as

$$\Xi_{x+u,y+v}(\mathbf{n}) = (1 - |x+u-i|)(1 - |y+v-j|)E(\mathbf{n}). \tag{9}$$

The pixel in $\aleph_4$ that has maximal edge strength impacting on interior pixel $(x+u, y+v)$ is then selected as the dominant pixel $\hat{\mathbf{n}}$:

$$\hat{\mathbf{n}} = \arg_{\mathbf{n}\in\aleph_4}\left\{\max\left[\Xi_{x+u,y+v}(\mathbf{n})\right]\right\}. \tag{10}$$

Base on the assumption that the edge orientation of an interior pixel $(x+u, y+v)$ is directly designated as that of the dominant pixel, the orientation of $(x+u, y+v)$ is defined as $\theta(\hat{\mathbf{n}})$. Then, $\theta(\hat{\mathbf{n}})$ is quantized in four ordinary directions $D_\lambda (\lambda = 1, 2, 3, 4)$, as shown in Figure 6. Hence, the value of an interior pixel is interpolated based on different orientations (along the direction with maximal edge strength) and is given in the following cases, where each case corresponds a specific direction.

Case 1. When the direction of the interior pixel is D_1 as shown in Figure 7(a), the interpolated pixel value is obtained from

$$\tilde{f}_{x+u,y+v} = (1-u)f_{x,y+v} + uf_{x+1,y+v}. \tag{11}$$

Case 2. When the direction of the interior pixel is D_3 as shown in Figure 7(b), the interpolated pixel value is obtained from

$$\tilde{f}_{x+u,y+v} = (1-v)f_{x+u,y} + vf_{x+u,y+1}. \tag{12}$$

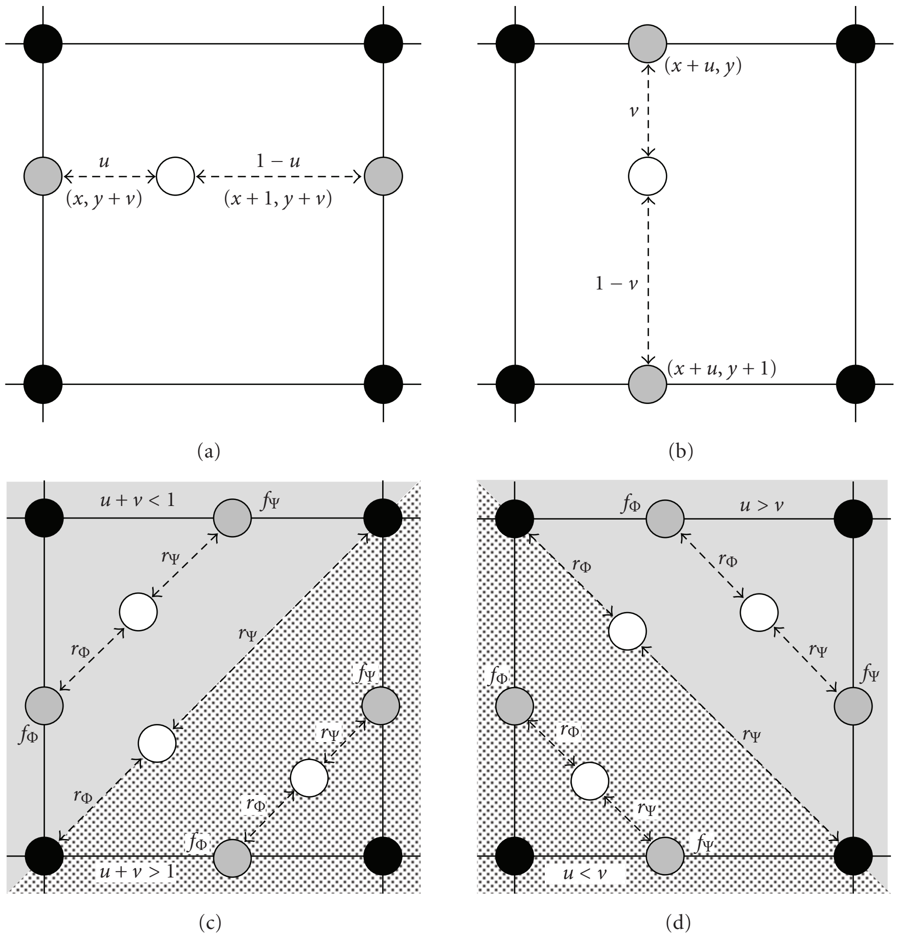


Figure 7: Four classical cases for interpolating interior pixels.

Case 3. When the direction of the interior pixel is D_2 as shown in Figure 7(c), the pixel value is linearly interpolated from those of its neighbors:

$$\tilde{f}_{x+u,y+v} = \frac{r_\Psi}{r_\Phi + r_\Psi} f_\Phi + \frac{r_\Phi}{r_\Phi + r_\Psi} f_\Psi, \tag{13}$$

where f_Φ, f_Ψ, r_Φ, and r_Ψ are obtained by considering the following three subcases.

Subcase 3.1. $f_\Phi = f_{x,y+1}$, $f_\Psi = f_{x+1,y}$, $r_\Phi = \sqrt{u^2 + (1-v)^2}$, $r_\Psi = \sqrt{(1-u)^2 + v^2}$, as $u + v = 1$.

Subcase 3.2. $f_\Phi = f_{x,y+u+v}$, $f_\Psi = f_{x+u+v,y}$, $r_\Phi = u$, $r_\Psi = v$, as $u + v < 1$.

Subcase 3.3. $f_\Phi = f_{x+u+v-1,y+1}$, $f_\Psi = f_{x+1,y+u+v-1}$, $r_\Phi = 1 - v$, $r_\Psi = 1 - u$, as $u + v > 1$.

Case 4. When the direction of the interior pixel is D_4 as shown in Figure 7(d), the pixel value is interpolated using (13), and subcases are described below.

Subcase 4.1. $f_\Phi = f_{x,y}$, $f_\Psi = f_{x+1,y+1}$, $r_\Phi = \sqrt{u^2 + v^2}$, $r_\Psi = \sqrt{(1-u)^2 + (1-v)^2}$, as $u = v$.

Subcase 4.2. $f_\Phi = f_{x+u-v,y}$, $f_\Psi = f_{x+1,y-u+v+1}$, $r_\Phi = v$, $r_\Psi = 1 - u$, as $u > v$.

Subcase 4.3. $f_\Phi = f_{x,y-u+v}$, $f_\Psi = f_{x+u-v+1,y+1}$, $r_\Phi = u$, $r_\Psi = 1 - v$, as $u < v$.

The two main stages of the estimation of the aligned pixels based on local gradients and of the estimation of interior pixels using edge properties of neighbors have now been introduced. Consequently, the proposed two-stage interpolation algorithm is carried out by implementing these two stages in correct order.

3. Parameters Determination

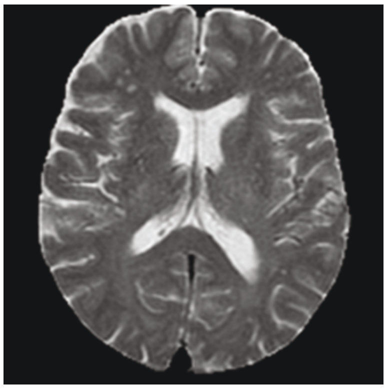
This section discusses the critical parameters of the proposed FIS that is used in the first stage of the estimation. The fuzzy sets of the FIS significantly influence the effectiveness of the proposed interpolation algorithm. Triangular and trapezoidal functions are the first recommended fuzzy membership functions because of their simplicity of implementation.

(a)

(b)

(c)

(d)


(e)

(f)

Figure 8: A set of test images: (a) Airplane, (b) Boats, (c) Cameraman, (d) Lena, (e) MRI, and (f) Peppers.

Moreover, some critical parameters must still be determined. They include $\{\alpha_1, \alpha_2, \ldots, \alpha_\Gamma\}$, $\{\beta_1, \beta_2, \ldots, \beta_\Gamma\}$ and Γ. Suppose that $\{\alpha_1, \alpha_2, \ldots, \alpha_\Gamma\}$ and $\{\beta_1, \beta_2, \ldots, \beta_\Gamma\}$ are equally spaced in the intervals $[0, \alpha_\Gamma]$ and $[0, 1]$, respectively, such that $\alpha_1 = 0$, $\beta_1 = 0$ and $\beta_\Gamma = 1$. The parameters are now reformulated as

$$\alpha_m = \frac{m-1}{\Gamma-1}\alpha_\Gamma, \quad \beta_m = \frac{m-1}{\Gamma-1}, \quad \text{for } m = 1, 2, \ldots, \Gamma. \tag{14}$$

Therefore, the total number of parameters is significantly reduced, and only two parameters, α_Γ and Γ, need now be considered. Next, the particle swarm optimization procedure is adopted to determine parameters.

Particle swarm optimization (PSO) is a population-based evolutionary algorithm that was inspired by the social behavior of biological organisms [29, 30], which is associated with an optimization search for solutions. As an optimization approach, the PSO algorithm provides an iteratively searching capability in which each particle moves through the multidimensional search space. The ℓth particle in the n-dimensional search space is specified by three components, which are its position $\mathbf{p}_\ell = [p_{\ell 1}, p_{\ell 2}, \ldots, p_{\ell n}]$, its velocity $\mathbf{v}_\ell = [v_{\ell 1}, v_{\ell 2}, \ldots, v_{\ell n}]$, and the best position that it has achieved so far $\mathbf{b}_\ell = [b_{\ell 1}, b_{\ell 2}, \ldots, b_{\ell n}]$ (for individual best). Particles are initialized randomly and the particles then move through the search space. In time step t, the position of the ℓth particle $\mathbf{p}_\ell$ ($\ell = 1, 2, \ldots, M$) is updated by adding a velocity vector $\mathbf{v}_i$, where M is the total number of particles in the swarm. In standard PSO, the velocity and position are updated using (15):

$$\begin{aligned} \mathbf{v}_\ell(t+1) &= w(t)\mathbf{v}_\ell(t) + \rho_1[\mathbf{b}_\ell(t) - \mathbf{p}_\ell(t)] + \rho_2\left[\hat{\mathbf{b}}(t) - \mathbf{p}_\ell(t)\right], \\ \mathbf{p}_\ell(t+1) &= \mathbf{p}_\ell(t) + \mathbf{v}_\ell(t+1), \end{aligned} \tag{15}$$

where $\hat{\mathbf{b}}$ is the best position found so far among all particles in the swarm (for global best); ρ_1 and ρ_2 are random numbers distributed in the interval $[0, 1]$ that are generated in each time step.

Let $f_{x,y} \in I_{\mathrm{HR}}$ and $\tilde{f}_{x,y} \in \tilde{I}_{\mathrm{HR}}$ represent the pixel values in the original image and in the interpolated image, respectively. The peak signal-to-noise ratio (PSNR) between the two images, given by (16), is adopted as the fitness function J_ℓ to evaluate the solution for the ℓth particle:

$$\mathrm{PSNR} = 10 \cdot \log_{10} \frac{(L-1)^2}{\mathrm{MSE}}, \tag{16}$$

$$\mathrm{MSE} = \frac{1}{\kappa W \cdot \kappa H} \sum_{x=1}^{\kappa W} \sum_{y=1}^{\kappa H} \left(\tilde{f}_{x,y} - f_{x,y}\right)^2, \tag{17}$$

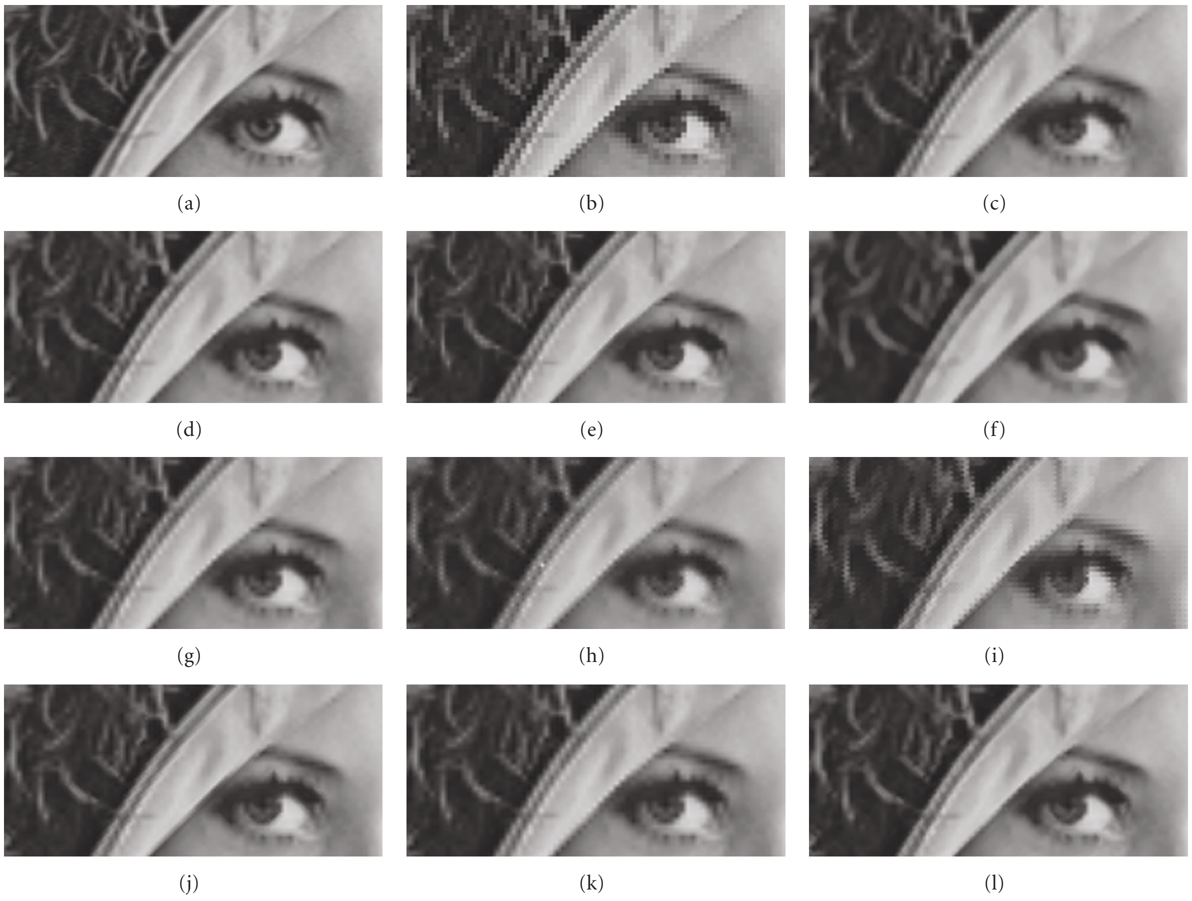


Figure 9: Close-up part of (a) original *Lena* image and its reproduced version interpolated by (b) Nearest Neighbor, (c) Bilinear, (d) WaDi-Bil, (e) Adaptive-Bil, (f) CFLS, (g) ESIF, (h) EDI, (i) EOA, (j) LAZA, and (k) SAI and (l) the proposed method.

```
for each time step $t$ do
  for $\ell$th particle in swarm do
      Update the position $\mathbf{p}_\ell$ using (14).
      Calculate fitness function $\mathbf{J}_\ell$.
      Update the best positions $\mathbf{b}_\ell$ and $\hat{\mathbf{b}}_\ell$.
  end for
end for
```

Algorithm 1: Main procedure in particle swarm optimization.

where $L = 2^8 = 256$ for 8-bit images; κW and κH denote the width and the height of the interpolated images. The main procedure in particle swarm optimization is summarized as in Algorithm 1.

Assume that an integer-valued particle $\mathbf{p}_\ell = [p_{\ell 1}, p_{\ell 2}]$ moves through a two-dimensional search space that is bounded in $[0, L-1] \times [1, \Gamma_{\max}]$. The terms $p_{\ell 1}$ and $p_{\ell 2}$ stand for parameters α_Γ and Γ, respectively. To reduce the computational burden, the maximal number of fuzzy rules is set to $\Gamma_{\max} = 15$. The values optimized by PSO are $\alpha_\Gamma = 136$ and $\Gamma = 9$. Consequently, the critical parameters that characterize the proposed FIS are completely determined using (14).

4. Experimental Results

The performance of each tested interpolation method was evaluated by reproducing HR images from their associated LR images. The numerical tool MATLAB that is commonly used in engineering applications was adopted to simulate the following experiments. Figure 8 shows a set of test images of size 512×512 pixels. The results of the proposed interpolation algorithm were compared with those obtained using several existing methods. These methods are the nearest neighbors method, bilinear, warped distance for bilinear (WaDi-Bil) [14], adaptive bilinear (Adaptive-Bil) [13], closed-form least-squares technique for bilinear (CFLS) [26], edge-sensitive interpolation filter (ESIF) [1], new edge-directed interpolation (EDI) [2], edge-oriented algorithm (EOA) [3], locally adaptive zooming algorithm (LAZA) [5], and soft-decision estimation technique for adaptive image interpolation (SAI) [6]. The original image was first subsampled into an LR version with a reduced size of 256×256 pixels. This associated LR image was reproduced to meet its original size using each interpolation method. The effectiveness of each method was quantitatively evaluated in terms of PSNR between the original image and the reproduced one. An MF $\kappa = 2$ was chosen in

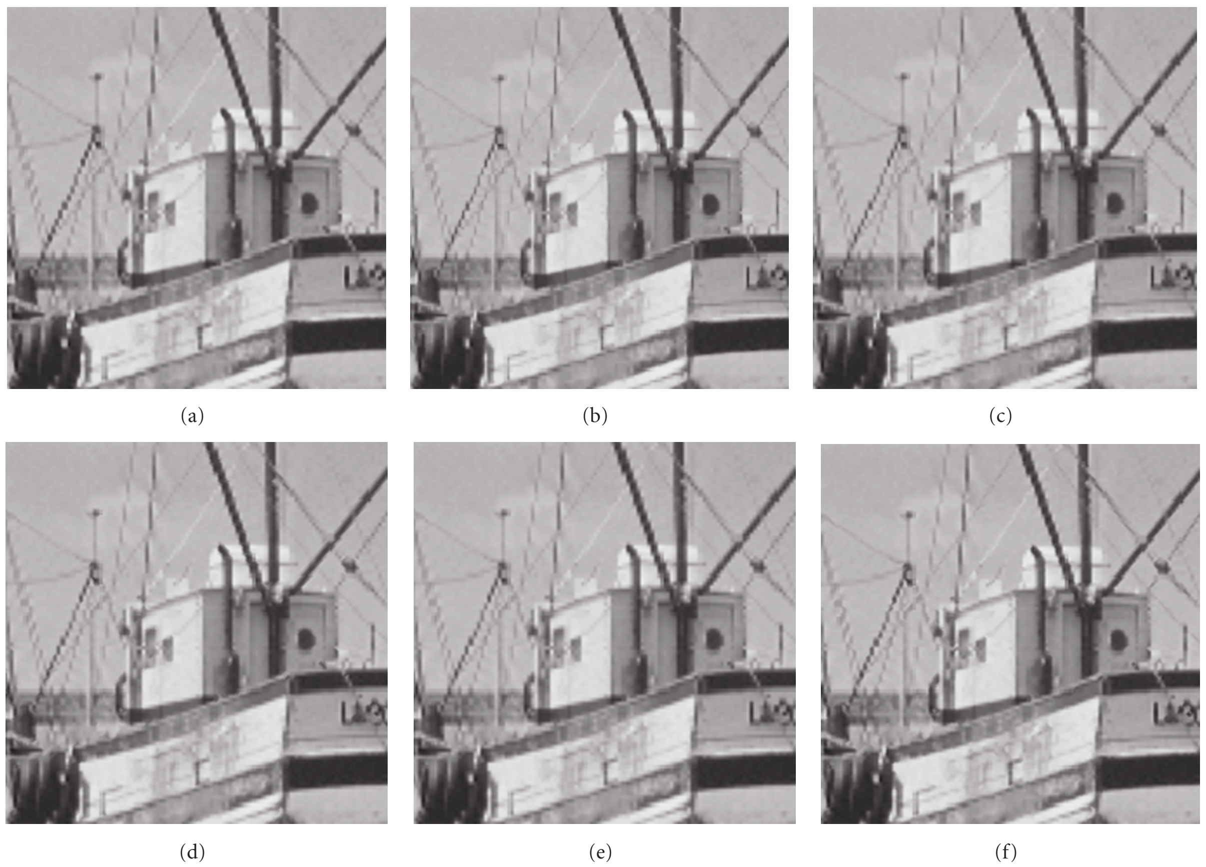


Figure 10: Close-up part of reproduced *Boats* image processed by six well-performing methods: (a) WaDi-Bil, (b) Adaptive-Bil, (c) ESIF, (d) LAZA, (e) SAI, and (f) the proposed method.

Table 1: Numerical results presented in PSNRs between different interpolation methods.

Methods	Peak signal-to-noise ratios (in dBs) for different test images					
	Airplane	*Boats*	*Cameraman*	*Lena*	*MRI*	*Peppers*
Nearest neighbor	27.19	26.11	27.48	29.10	31.32	29.52
Bilinear	32.75	30.36	34.14	35.65	41.10	35.27
WaDi-Bil [14]	33.22	30.50	35.42	36.19	42.63	35.73
Adaptive-Bil [13]	33.15	30.41	34.35	35.85	41.12	36.22
CFLS [26]	31.74	29.84	33.14	34.09	36.52	33.91
ESIF [1]	32.89	30.43	34.68	35.89	42.64	35.67
EDI [2]	32.88	29.96	33.10	35.60	41.54	35.75
EOA [3]	31.35	29.37	34.05	34.24	38.79	33.95
LAZA [5]	32.75	30.42	34.24	35.91	41.13	35.47
SAI [6]	32.86	30.38	35.06	35.84	42.65	36.40
Proposed method	33.67	30.86	36.27	36.85	43.44	36.42

this experiment because some of the compared methods have been proposed for use in 2× image enlargement applications [2, 3, 5, 6]. Table 1 numerically compares the PSNR values obtained using different methods. Notably, the proposed two-stage interpolation algorithm outperforms other methods by an average of 0.7–5 dB. The SAI approach resulted in acceptable performance but cost much execution time since it estimates four pixel values using 21 neighboring pixels (complete derivation could be attached in [6]). Table 2 lists the structural similarity (SSIM) comparison among different methods. The SSIM index that is normalized in $[0, 1]$ quantifies the visibility of differences between the original HR image and the interpolated image [31].

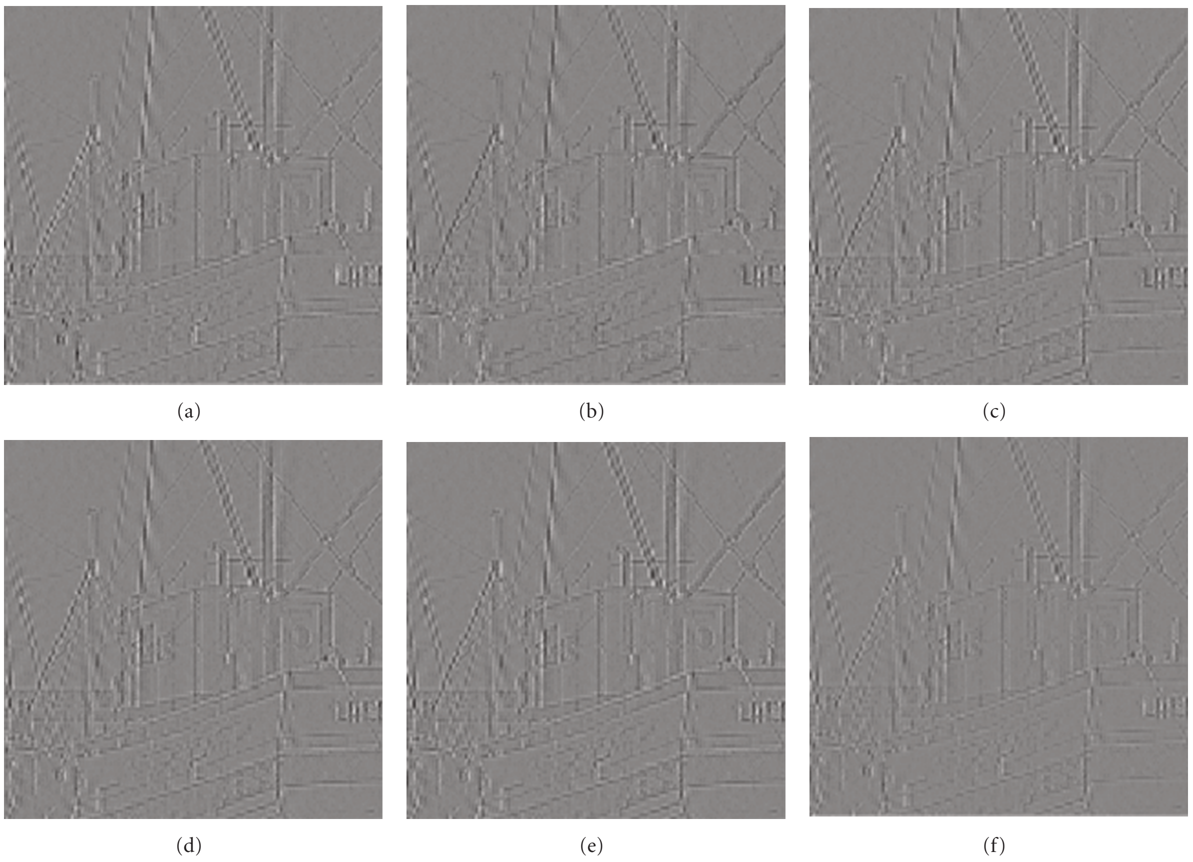


FIGURE 11: Error image with respect to original image for Figure 10.

TABLE 2: Numerical results presented in SSIMs between different interpolation methods.

Methods	Structural similarity for different test images					
	Airplane	*Boats*	*Cameraman*	*Lena*	*MRI*	*Peppers*
Nearest Neighbor	0.9508	0.9157	0.9640	0.9456	0.981	0.9536
Bilinear	0.9816	0.9594	0.9902	0.9830	0.9973	0.9819
WaDi-Bil [14]	0.9825	0.9602	0.9917	0.9838	0.9977	0.9825
Adaptive-Bil [13]	0.9820	0.9558	0.9895	0.9827	0.9977	0.9827
CFLS [26]	0.9824	0.9687	0.9831	0.9820	0.9934	0.9844
ESIF [1]	0.9818	0.9595	0.9907	0.9831	0.9977	0.9823
EDI [2]	0.9816	0.9593	0.9836	0.9829	0.9973	0.9825
EOA [3]	0.9810	0.9584	0.9881	0.9821	0.9857	0.9816
LAZA [5]	0.9816	0.9555	0.9908	0.9830	0.9978	0.9826
SAI [6]	0.9830	0.9603	0.9926	0.9832	0.9981	0.9835
Proposed Method	0.9836	0.9614	0.9929	0.9851	0.9985	0.9834

Figure 9 presents a close-up part of the *Lena* image, interpolated using different image interpolation approaches. As demonstrated, the nearest neighbor method yields blocking artifacts; the bilinear and CFLS methods blur the details and edges, but the proposed algorithm preserves them. Figure 10 displays another simulation of the *Boats* image, which contains many edges. In this figure, six well-performing interpolation methods were demonstrated. To compare clearly the performance of different methods, Figure 11 illustrates an error map of the interpolated image with respect to the original image. The halftone grayscale pixels (with a gray value of 128) are error-free in the interpolated image, while the dark (or light) pixels exhibit negative (or positive) interpolation errors. All the simulation results including Figures 9–11 are available online. Please refer to http://www.ee.ncu.edu.tw/~fuzzylab/ExpResults/ for downloading the experimental results. The proposed interpolation algorithm certainly reduces the interpolation errors, particularly in sharp regions.

5. Conclusions

This study has presented an efficient interpolation method for image processing applications. The HR images are reproduced by estimating the values of aligned pixels and then estimating the values of interior pixels. The first stage deals with the lost pixels using an intelligent scheme that combines fuzzy logic and particle swarm optimization. The grayscale values of aligned pixels are interpolated by using the optimized fuzzy inference system whose input is the local gradient information. In the second interpolative stage, the lost pixels in an interior region are restored from the edge features of their neighbors. Numerical comparisons verify the effectiveness of the proposed interpolation algorithm applied to different images. Close-up observations and error maps demonstrate the superiority of the proposed algorithm over other methods in restoring local edges. The proposed two-stage interpolation algorithm yields substantially improved performance of image zooming and enlargement.

Acknowledgments

The authors would like to thank the National Science Council of the Republic of China, Taiwan, for financially supporting this research under the Grant NSC-96-2221-E-027-136. The authors also thank the National Central University, Taiwan, for providing website services. Please refer to http://www.ee.ncu.edu.tw/~fuzzylab/ExpResults/ (National Central University) for downloading the results and related materials.

References

[1] S. Carrato, G. Ramponi, and S. Marsi, "Simple edge-sensitive image interpolation filter," in *Proceedings of the IEEE International Conference on Image Processing*, vol. 3, pp. 711–714, 1996.

[2] X. Li and M. T. Orchard, "New edge-directed interpolation," *IEEE Transactions on Image Processing*, vol. 10, no. 10, pp. 1521–1527, 2001.

[3] M.-J. Chen, C.-H. Huang, and W.-L. Lee, "A fast edge-oriented algorithm for image interpolation," *Image and Vision Computing*, vol. 23, no. 9, pp. 791–798, 2005.

[4] L. Zhang and X. Wu, "An edge-guided image interpolation algorithm via directional filtering and data fusion," *IEEE Transactions on Image Processing*, vol. 15, no. 8, pp. 2226–2238, 2006.

[5] S. Battiato, G. Gallo, and F. Stanco, "A locally adaptive zooming algorithm for digital images," *Image and Vision Computing*, vol. 20, no. 11, pp. 805–812, 2002.

[6] 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, 2008.

[7] D. D. Muresan and T. W. Parks, "Adaptively quadratic (AQua) image interpolation," *IEEE Transactions on Image Processing*, vol. 13, no. 5, pp. 690–698, 2004.

[8] T. M. Lehmann, C. Gönner, and K. Spitzer, "Surveys: interpolation methods in medical image processing," *IEEE Transactions on Medical Imaging*, vol. 18, no. 11, pp. 1049–1075, 1999.

[9] P. Thévenaz, T. Blu, and M. Unser, "Interpolation revisited," *IEEE Transactions on Medical Imaging*, vol. 19, no. 7, pp. 739–758, 2000.

[10] R. G. Keys, "Cubic convolution interpolation for digital image processing," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 29, no. 6, pp. 1153–1160, 1981.

[11] S. E. Reichenbach and F. Geng, "Two-cimensional cubic convolution," *IEEE Transactions on Image Processing*, vol. 12, no. 8, pp. 857–865, 2003.

[12] J. Z. Shi and S. E. Reichenbach, "Image interpolation by two-dimensional parametric cubic convolution," *IEEE Transactions on Image Processing*, vol. 15, no. 7, pp. 1857–1870, 2006.

[13] J. W. Hwang and H. S. Lee, "Adaptive image interpolation based on local gradient features," *IEEE Signal Processing Letters*, vol. 11, no. 3, pp. 359–362, 2004.

[14] G. Ramponi, "Warped distance for space-variant linear image interpolation," *IEEE Transactions on Image Processing*, vol. 8, no. 5, pp. 629–639, 1999.

[15] Q. Wang and R. K. Ward, "A new orientation-adaptive interpolation method," *IEEE Transactions on Image Processing*, vol. 16, no. 4, pp. 889–900, 2007.

[16] T. Hermosilla, E. Bermejo, A. Balaguer, and L. A. Ruiz, "Non-linear fourth-order image interpolation for subpixel edge detection and localization," *Image and Vision Computing*, vol. 26, no. 9, pp. 1240–1248, 2008.

[17] T. Sigitani, Y. Iiguni, and H. Maeda, "Image interpolation for progressive transmission by using radial basis function networks," *IEEE Transactions on Neural Networks*, vol. 10, no. 2, pp. 381–390, 1999.

[18] N. Plaziac, "Image interpolation using neural networks," *IEEE Transactions on Image Processing*, vol. 8, no. 11, pp. 1647–1651, 1999.

[19] A. Amanatiadis, I. Andreadis, and K. Konstantinidis, "Design and implementation of a fuzzy area-based image-scaling technique," *IEEE Transactions on Instrumentation and Measurement*, vol. 57, no. 8, pp. 1504–1513, 2008.

[20] L. Ma, Y. Shen, and J. Ma, "Local spatial properties based image interpolation scheme using SVMs," *Journal of Systems Engineering and Electronics*, vol. 19, no. 3, pp. 618–623, 2008.

[21] D. G. Sheppard, K. Panchapakesan, A. Bilgin, B. R. Hunt, and M. W. Marcellin, "Lapped nonlinear interpolative vector quantization and image super-resolution," *IEEE Transactions on Image Processing*, vol. 9, no. 2, pp. 295–298, 2000.

[22] S.-H. Hong, R.-H. Park, S. Yang, and J.-Y. Kim, "Image interpolation using interpolative classified vector quantization," *Image and Vision Computing*, vol. 26, no. 2, pp. 228–239, 2008.

[23] A. Temizel and T. Vlachos, "Wavelet domain image resolution enhancement," *IEE Proceedings: Vision, Image and Signal Processing*, vol. 153, no. 1, pp. 25–30, 2006.

[24] S. G. Chang, Z. Cvetković, and M. Vetterli, "Locally adaptive wavelet-based image interpolation," *IEEE Transactions on Image Processing*, vol. 15, no. 6, pp. 1471–1485, 2006.

[25] F. Aràndiga, R. Donat, and P. Mulet, "Adaptive interpolation of images," *Signal Processing*, vol. 83, no. 2, pp. 459–464, 2003.

[26] H. Yoo, "Closed-form least-squares technique for adaptive linear image interpolation," *Electronics Letters*, vol. 43, no. 4, pp. 210–212, 2007.

[27] Y. Cha and S. Kim, "The error-amended sharp edge (EASE) scheme for image zooming," *IEEE Transactions on Image Processing*, vol. 16, no. 6, pp. 1496–1505, 2007.

[28] N. Suetake, M. Sakano, and E. Uchino, "Image enlargement based on self-produced codebook," *Electronics Letters*, vol. 43, no. 3, pp. 152–153, 2007.

[29] J. Kennedy, “Particle swarm optimization,” in *Proceedings of the IEEE International Conference on Neural Networks (ICNN ’95)*, vol. 4, pp. 1942–1948, Australia, November 1995.

[30] R. Eberhart and J. Kennedy, “A new optimizer using particle swarm theory,” in *Proceedings of the 6th International Symposium on Micro Machine and Human Science (MHS ’95)*, pp. 39–43, Nagoya, Japan, October 1995.

[31] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.