Fuzzy Sets and Systems 114 (2000) 225–238

FUZZY sets and systems

www.elsevier.com/locate/fss

2-D discrete signal interpolation and its image resampling application using fuzzy rule-based inference

Jia-Lin Chen, Jyh-Yeong Chang*, Kun-Li Shieh

Department of Electrical and Control Engineering, National Chiao Tung University, Taiwan, ROC

Received August 1997; received in revised form March 1998

Abstract

This paper describes an interpolation algorithm for two-dimensional (2-D) discrete signals using fuzzy rule-based inference. The original signal is estimated by the *main-surface* function in the interpolation region, and four *sub-plane* functions surrounding the interpolation region. The main-surface is a bilinearly interpolated function passing through four signal samples in the interpolation region and the four sub-planes reflect the tendencies of pixels from the left, right, up, and down of the interpolation region. Drawing fuzzy inferences about signals from these five functions, we can estimate original signals very well even when the signals are buried in noise. We verified the method by computer simulations of some assumed 2-D signals and by resampling of the actual image data. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: 2-D discrete signal interpolation; Fuzzy inference; Membership function; Image resampling

1. Introduction

Interpolation algorithms for 2-D signals have been proven useful in a wide range of applications, especially in image processing systems [11]. All interpolation algorithms generally start by determining a continuous interpolation function from a set of discrete signal samples, and then resample this interpolated function according to the number of points specified. The simplest algorithms are the nearest neighbor and bilinear interpolations [16]. The more refined cubic *B*-spline interpolation algorithm, proposed by Hou and Andrews [7], is, however, computationally complex. Several alternative cubic convolution techniques [3,15,20] have been proposed to reduce its computational complexity. It must be noted that all these algorithms were proposed to estimate original signals in noise-free environments. However, during the courses of transmission, storage, and retrieval, noise is generally embedded in signals from various sources, i.e., during coding and decoding, errors are induced in signals and transmitted signals are usually corrupted by noise. Because the interpolation curves obtained by using these algorithms are so calculated that they pass exactly through the given noisy

* Corresponding author.

PII: S0165-0114(98)00090-6

points, these algorithms are obviously not good at estimating original signals from noisy signals. To improve this situation, we propose a new 2-D noisy signal interpolation algorithm, based mainly on the fuzzy inference rule.

Fuzzy logic has been successfully applied to various kinds of engineering problems [1,2,9,10,18]. Fuzzy theory allows the inclusion of fuzzy if–then rules concerning fuzzy concepts. These rules may come from human experts or be generated automatically by matching input–output pairs through training routines [4,6,12]. In the field of image processing, successful image enhancement results have been reported after the fuzzy indices of input images were reduced [13,14], and [17] used fuzzy inference rules derived from pixels of interest and its neighboring pixels. In particular, a fuzzy rule-based interpolation algorithm for one-dimensional discrete signals has recently been proposed [19]. This algorithm interpolates between two noise-free points in Euclidean space by aggregating the fuzzy sets defined on the surrounding points. An advanced algorithm for the noisy points is also proposed; this interpolation is done by classifying the neighboring noisy points into groups. In this direction, this paper investigates a fuzzy rule-based interpolation algorithm for 2-D signals.

This paper is organized into four sections. The architectural framework of the entire system is discussed in detail in Section 2. Derivation of the main surface and the four sub-planes are then presented. This is followed by construction of the membership functions and fuzzy rules for inferring 2-D signals. Computer simulations of assumed 2-D signals and image resampling examples are presented in Section 3. Comparison results are also given. Section 4 contains some concluding remarks.

2. 2-D signal interpolation by fuzzy rule-based inference

During the courses of transmission, storage, and retrieval, 2-D signals are usually corrupted by noise. Since the signals are inevitably buried in noise, schemes that can remove the noise during the course of interpolation are required. In the spatial domain, a successful and frequently used method for removing noise from images is neighborhood averaging [5, Section 4.3], a method mainly concerned with masks. If the center of a mask is located at pixel (x, y) in an image, the gray level of (x, y) is replaced by the average gray level of all pixels in the area of the mask. In a similar manner, our proposed 2-D signal interpolation scheme uses fuzzy inference, a more sophisticated neighborhood averaging technique involving application of fuzzy logic to sampled noisy data. That is, we propose a 2-D fuzzy rule-based interpolation scheme in which fuzzy inference is used to derive results similar to those obtained by using neighborhood averaging for interpolation tasks. First, we find the main-surface function of the interpolation region and then find four surrounding plane functions, called sub-planes, of the main-surface. The main-surface is a bilinearly interpolated function passing through four signal samples, and the sub-planes reflect the tendencies of pixel fluctuations from four directions, left, right, up, and down, of the interpolation region. Next, some fuzzy if–then rules for these five functions are presented to carry out the interpolation task. Because of the inference rules employed for interpolation, original signals can be correctly interpolated and resampled despite the original signals having been corrupted by noise.

Let $\hat{f}(x, y)$ be the estimated function and $I_s(i,j)$, $i = 0, 1, \ldots, M-1$, $j = 0, 1, \ldots, N-1$, be the $M \times N$ sampled noisy data as shown in Fig. 1(a). In Fig. 1(b), we define an interpolation region $R_{i,j}$ as follows:

$$R_{i,j} = \{(x, y) \mid x_i \leqslant x \leqslant x_{i+1},\ y_j \leqslant y \leqslant y_{j+1}\}, \quad i = 0, 1, \ldots, M-2,\ j = 0, 1, \ldots, N-2. \tag{1}$$

From (1) we can similarly represent the eight regions, $R_{i-2,j}$, $R_{i-1,j}$, $R_{i+1,j}$, $R_{i+2,j}$, $R_{i,j-2}$, $R_{i,j-1}$, $R_{i,j+1}$, and $R_{i,j+2}$, around the region of interest. In our interpolation algorithm, we use five functions, one surface function generated from $R_{i,j}$ and the other four plane functions generated from $R_{i-1,j}$, $R_{i+1,j}$, $R_{i,j-1}$, and $R_{i,j+1}$, respectively, to determine the estimation function $\hat{f}(x, y)$. Function $F_{\mathrm{c}}(x, y)$, i.e., the *main-surface*, passes through the four signal samples, $I_s(i,j)$, $I_s(i+1,j)$, $I_s(i,j+1)$, and $I_s(i+1,j+1)$, of the interpolation

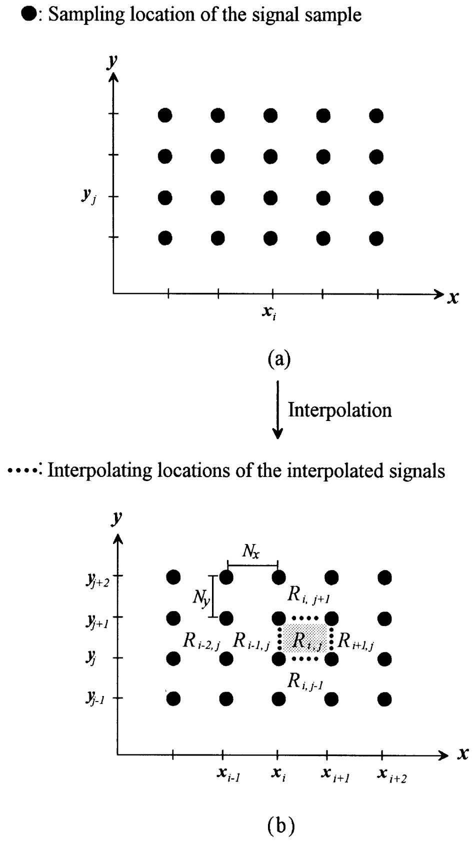

Fig. 1. The interpolation region and its contiguous regions.

region $R_{i,j}$. Let $f_{\rm l}(x,y)$, $f_{\rm r}(x,y)$, $f_{\rm u}(x,y)$, and $f_{\rm d}(x,y)$ be the four *sub-planes* that pass through the local signal averages of the four contiguous regions, $R_{i-1,j}$, $R_{i+1,j}$, $R_{i,j-1}$, and $R_{i,j+1}$, and reflect pixel fluctuation tendencies. Subscripts l, r, u and d denote left, right, up, and down, respectively. To interpolate a point (x,y) in the region $R_{i,j}$, $\hat{f}(x,y)$ is obtained by taking into account the fuzzy aggregation of the function $F_{\rm c}(x,y)$ in the interpolation region and the four planes, $f_{\rm l}(x,y)$, $f_{\rm r}(x,y)$, $f_{\rm u}(x,y)$, and $f_{\rm d}(x,y)$ next to the interpolation region.

Let the original $M\times N$ 2-D signal $I_s(i,j)$ be interpolated during resampling periods N_x and N_y in the x- and y-directions, respectively. In region $R_{i,j}$ we can then calculate the new coordinates (x_i,y_j) of the given sample points and the new coordinate (x,y) of the interpolated point as follows (see Fig. 1): $x_i=iN_x$, $i=0,1,\dots,M-1$, $y_j=jN_y$, $j=0,1,\dots,N-1$; $x=x_i+u$, $u=0,1,\dots,N_x-1$, $y=y_j+v$, $v=0,1,\dots,N_y-1$. In what follows, the procedures for deriving the main-surface, sub-planes, and for constructing the fuzzy rules are described.

2.1. Deriving the interpolation main-surface

Function $F_{\rm c}(x,y)$ is bilinearly interpolated from the four corner signals of region $R_{i,j}$ and we call function $F_{\rm c}(x,y)$ the main-surface. Suppose $F_{\rm c}(x,y)$ is represented by the following:

$$F_{\rm c}(x,y)=ax+by+cxy+d,\quad x_i\leqslant x\leqslant x_{i+1},\ \ y_j\leqslant y\leqslant y_{j+1}. \tag{2}$$

In (2), the four unknown coefficients, a, b, c, and d, can easily be determined from the following four equations using these four signal samples. I.e.,

$$\begin{aligned} I_s(i,j) &= ax_i + by_j + cx_i y_j + d,\\ I_s(i,j+1) &= ax_i + by_{j+1} + cx_i y_{j+1} + d,\\ I_s(i+1,j) &= ax_{i+1} + by_j + cx_{i+1} y_j + d,\\ I_s(i+1,j+1) &= ax_{i+1} + by_{j+1} + cx_{i+1} y_{j+1} + d. \end{aligned} \tag{3}$$

Thus, we have

$$\begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} x_i & y_j & x_i y_j & 1 \\ x_i & y_{j+1} & x_i y_{j+1} & 1 \\ x_{i+1} & y_j & x_{i+1} y_j & 1 \\ x_{i+1} & y_{j+1} & x_{i+1} y_{j+1} & 1 \end{pmatrix}^{-1} \begin{pmatrix} I_s(i,j) \\ I_s(i,j+1) \\ I_s(i+1,j) \\ I_s(i+1,j+1) \end{pmatrix}. \tag{4}$$

After mathematical manipulations (see Appendix A), (2) takes the following form:

$$F_c(x,y) = \alpha(x,y) I_s(i,j) + \beta(x,y) I_s(i,j+1) + \gamma(x,y) I_s(i+1,j) + \eta(x,y) I_s(i+1,j+1), \tag{5}$$

where

$$\begin{aligned} \alpha(x,y) &= (1-(x-x_i)/N_x)(1-(y-y_j)/N_y),\\ \beta(x,y) &= (1-(x-x_i)/N_x)((y-y_j)/N_y),\\ \gamma(x,y) &= ((x-x_i)/N_x)(1-(y-y_j)/N_y), \end{aligned} \tag{6}$$

and

$$\begin{aligned} &\eta(x,y) = ((x-x_i)/N_x)((y-y_j)/N_y); \quad x_i \leqslant x \leqslant x_{i+1}, \;\; y_j \leqslant y \leqslant y_{j+1};\\ &\qquad\qquad i = 0,1,\ldots,M-2, \;\; j = 0,1,\ldots,N-2. \end{aligned} \tag{7}$$

2.2. Deriving the four sub-planes

To derive the sub-plane, using the left-hand surrounding plane f_l as an illustrative example, we need to know the relationship between the local signal average of the interpolation region and the local signal averages of the two consecutively contiguous regions on the left-hand side of the interpolation region. These local signal averages are calculated as follows:

$$\begin{aligned} &G_{i,j} = \tfrac{1}{4}\{I_s(i,j) + I_s(i,j+1) + I_s(i+1,j) + I_s(i+1,j+1)\},\\ &\qquad i = 0,1,\ldots,M-2, \;\; j = 0,1,\ldots,N-2. \end{aligned} \tag{8}$$

In (8), $G_{i,j}$ is the local signal average of the interpolation region $R_{i,j}$. The local signal averages, $G_{i-2,j}$ and $G_{i-1,j}$, of these two contiguous regions, $R_{i-2,j}$ and $R_{i-1,j}$ can be calculated in a similar manner.

Sub-plane $f_l(x,y)$ may be inclined or flat. If it is inclined, it is parallel to the inclined plane included in the cuboid Cub_l of the contiguous region $R_{i-1,j}$. Cuboid Cub_l is bounded by the maximum and minimum of the sampled noisy data, $I_s(i-1,j)$, $I_s(i-1,j+1)$, $I_s(i,j)$, and $I_s(i,j+1)$, in the region of interest $R_{i-1,j}$,

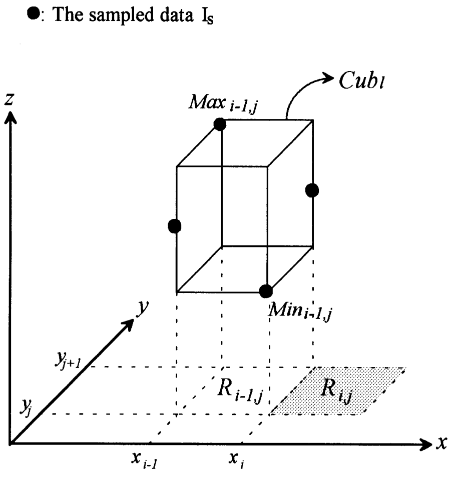

Fig. 2. The cuboid, Cub_{l}, in $R_{i-1,j}$.

as shown in Fig. 2; hence, the cuboid will enclose these sampled data. To this end, the maximum $Max_{i-1,j}$ and minimum $Min_{i-1,j}$ of $R_{i-1,j}$ are defined as

$$\begin{aligned} Max_{i-1,j} &= \max\{I_s(i-1,j), I_s(i-1,j+1), I_s(i,j), I_s(i,j+1)\}, \\ Min_{i-1,j} &= \min\{I_s(i-1,j), I_s(i-1,j+1), I_s(i,j), I_s(i,j+1)\}, \\ & i=1,2,\ldots,M-1,\ \ j=0,1,\ldots,N-2. \end{aligned} \tag{9}$$

Sub-plane $f_{\mathrm{l}}(x,y)$ is determined according to the geometric perspective among the following three contiguous regions, $\{R_{i-2,j},\ R_{i-1,j},$ and $R_{i,j}\}$. To derive $f_{\mathrm{l}}(x,y)$, the following parameters, G_1, G_2, G_3, $(x_{\mathrm{c}},y_{\mathrm{c}})$, *max*, and *min*, must be defined. G_1, G_2, and G_3 are, respectively, the local signal averages of the regions $R_{i-2,j}$, $R_{i-1,j}$, and $R_{i,j}$. $(x_{\mathrm{c}},y_{\mathrm{c}})$ denotes the coordinate of the center of $R_{i-1,j}$; whereas *max* and *min* represent the maximum, $Max_{i-1,j}$, and the minimum, $Min_{i-1,j}$, of $R_{i-1,j}$, respectively. In summary, to derive $f_{\mathrm{l}}(x,y)$, these parameters are defined as follows (see Fig. 3(a)): $G_1 = G_{i-2,j}$, $G_2 = G_{i-1,j}$, $G_3 = G_{i,j}$; $x_{\mathrm{c}} = (x_{i-1}+x_i)/2$, $y_{\mathrm{c}} = (y_j + y_{j+1})/2$; $max = Max_{i-1,j}$, $min = Min_{i-1,j}$. Next we use these parameters to derive the sub-plane $f_{\mathrm{l}}(x,y)$. When deriving $f_{\mathrm{l}}(x,y)$, we must distinguish among three cases.

Case 1: $G_1 < G_2 < G_3$. In this case, pixel values tend to increase from the left-hand side of $R_{i,j}$, and sub-plane $f_{\mathrm{l}}(x,y)$ is introduced to reflect this tendency. Sub-plane $f_{\mathrm{l}}(x,y)$ is an inclined plane that passes through the point $((x_{\mathrm{c}},y_{\mathrm{c}}),G_2)$ and parallels plane P_{lh} in the cuboid Cub_{l}, where the subscript lh indicates the tendency from lower gray levels toward higher ones as the pixel moves from the left-hand side of $R_{i,j}$. Plane P_{lh} and its normal vector V_{lh} are shown in Fig. 4(a). After the coordinates of the vertices of Cub_{l} have been taken, the equation of the normal vector V_{lh} is given by

$$\begin{aligned} V_{\mathrm{lh}} &= (v_1,v_2,v_3) = (N_x, 0, max-min) \times (0, N_y, 0) \\ &= \left(\begin{vmatrix} 0 & max-min \\ N_y & 0 \end{vmatrix}, \begin{vmatrix} max-min & N_x \\ 0 & 0 \end{vmatrix}, \begin{vmatrix} N_x & 0 \\ 0 & N_y \end{vmatrix} \right), \end{aligned} \tag{10}$$

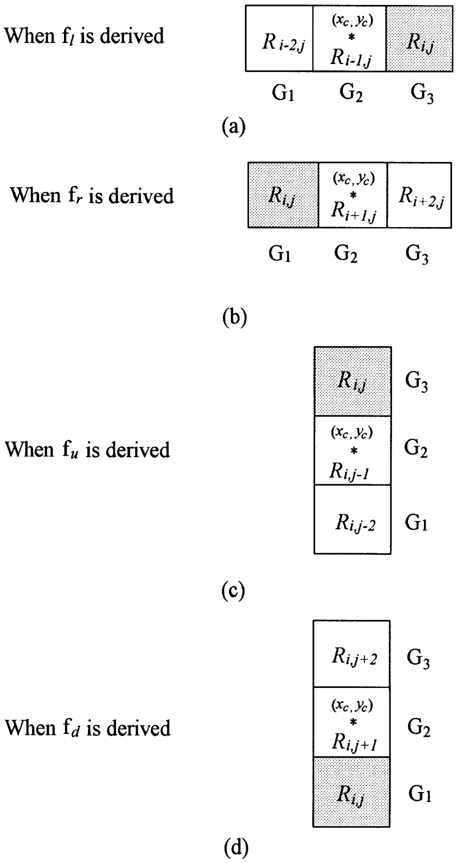

Fig. 3. The positional relationships among the parameters G_1, G_2, and G_3, referred to as the three-region sets, $\{R_{i-2,j}, R_{i-1,j}, R_{i,j}\}$, $\{R_{i,j}, R_{i+1,j}, R_{i+2,j}\}$, $\{R_{i,j}, R_{i,j+1}, R_{i,j+2}\}$, and $\{R_{i,j-2}, R_{i,j-1}, R_{i,j}\}$, of the three regions.

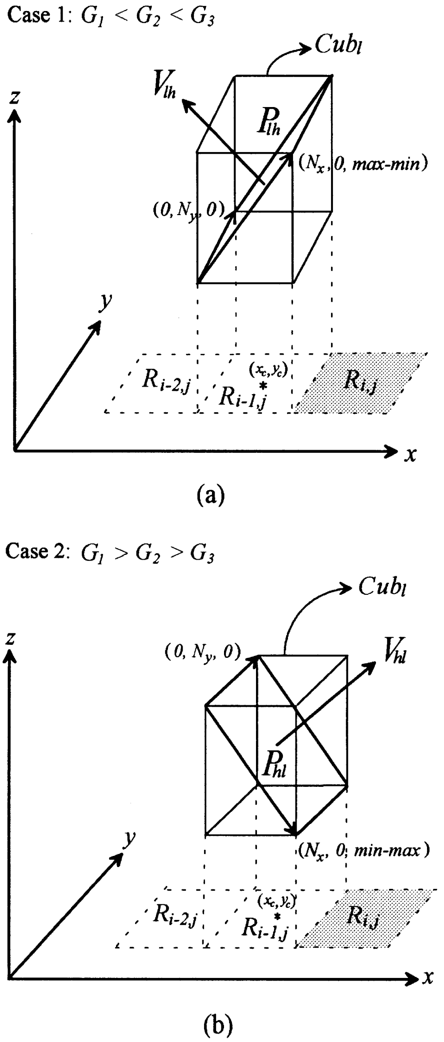

Fig. 4. When f_l is derived, (a) case 1 is satisfied; (b) case 2 is satisfied.

where × is the vector cross product. Let $f_l(x, y) = z$ be represented implicitly as

$$v_1 x + v_2 y + v_3 z + d = 0. \tag{11}$$

Since $((x_c, y_c), G_2)$ is on the plane $f_l(x, y)$, it follows that $((x_c, y_c), G_2)$ satisfies (11). Hence,

$$d = -v_1 x_c - v_2 y_c - v_3 G_2. \tag{12}$$

With vector V_{lh} and d given above, the left-hand interpolation sub-plane $f_l(x, y)$ is given by

$$f_l(x, y) = (-d - v_1 x - v_2 y)/v_3. \tag{13}$$

Case 2: $G_1 > G_2 > G_3$. In this case, pixel values tend to decrease from the left-hand side of $R_{i,j}$, and sub-plane $f_l(x, y)$ is introduced to reflect this tendency. Sub-plane $f_l(x, y)$ is again an inclined plane that passes through point $((x_c, y_c), G_2)$ and parallels plane P_{hl} in the cuboid Cub_l, where the subscript hl indicates a tendency from higher gray levels toward lower ones as the pixel moves from the left-hand side of R_{ij}. Fig. 4(b) shows plane P_{hl} and its normal vector V_{hl}. Normal vector V_{hl} is given by

$$\begin{aligned} V_{hl} &= (v_1, v_2, v_3) = (N_x, 0, min - max) \times (0, N_y, 0) \\ &= \left(\begin{vmatrix} 0 & min - max \\ N_y & 0 \end{vmatrix}, \begin{vmatrix} min - max & N_x \\ 0 & 0 \end{vmatrix}, \begin{vmatrix} N_x & 0 \\ 0 & N_y \end{vmatrix} \right). \end{aligned} \tag{14}$$

Case 3: *Neither case* 1 *nor case* 2 *condition holds*. No pixel tendency can be inferred in this case. Plane $f_l(x, y)$ is a horizontal plane that passes through point $((x_c, y_c), G_2)$. It is represented as follows:

$$f_l(x, y) = G_2. \tag{15}$$

Referring to Figs. 3(b)–(d), we can derive $f_r(x, y)$, $f_u(x, y)$, and $f_d(x, y)$ in a manner similar to that used to derive $f_l(x, y)$.

2.3. Constructing the fuzzy sets and inference rules for interpolation

Since the 2-D signal is interpolated via fuzzy inference on the main-surface and sub-planes, the membership functions for the main-surface and sub-planes are defined first. As shown in Fig. 5, $W_c(x, y)$ and $W_{i,j}(x, y)$ are the respective membership functions assigned to the main-surface and sub-planes. $W_c(x, y)$ is a cubic of height “1”. $W_{i,j}(x, y)$ is a pyramid of height “1” and is given by (see Fig. 6)

$$\begin{aligned} W_{i,j}(x, y) &= 1 - max\{2|x - x_c|/Length_x, 2|y - y_c|/Length_y\}, \\ &\qquad i = 0, 1, \ldots, M - 2, \; j = 0, 1, \ldots, N - 2, \end{aligned} \tag{16}$$

where $(x_c, y_c) = ((x_i + x_{i+1})/2, (y_j + y_{j+1})/2)$ is the center coordinate of each region and $Length_x$ and $Length_y$ represent the lengths of the pyramid in the x- and y-directions, respectively. The values of $Length_x$ and $Length_y$ can be chosen freely and they determine relative weighting of sub-planes on computing $\hat{f}(x, y)$. In this paper, we set $Length_x = 3N_x$ and $Length_y = 3N_y$. We use $W_{i-1,j}(x, y)$, $W_{i+1,j}(x, y)$, $W_{i,j-1}(x, y)$, and $W_{i,j+1}(x, y)$ to represent the membership functions assigned to $f_l(x, y)$, $f_r(x, y)$, $f_u(x, y)$, and $f_d(x, y)$, respectively, of which only $W_{i-1,j}(x, y)$ and $W_{i+1,j}(x, y)$ are shown in Fig. 5 for simplicity.

Finally, the interpolation of $\hat{f}(x, y)$ in the region $R_{i,j}$ is carried out according to the following rules:

Rule 1: If the interpolation membership function is W_c,
then the corresponding interpolation surface is $F_c(x, y)$.

Rule 2: If the interpolation membership function is $W_{i-1,j}$,
then the corresponding interpolation plane is $f_l(x, y)$.

Rule 3: If the interpolation membership function is $W_{i+1,j}$,
then the corresponding interpolation plane is $f_r(x, y)$. (17)

Rule 4: If the interpolation membership function is $W_{i,j-1}$,
then the corresponding interpolation plane is $f_u(x, y)$.

Rule 5: If the interpolation membership function is $W_{i,j+1}$,
then the corresponding interpolation plane is $f_d(x, y)$.

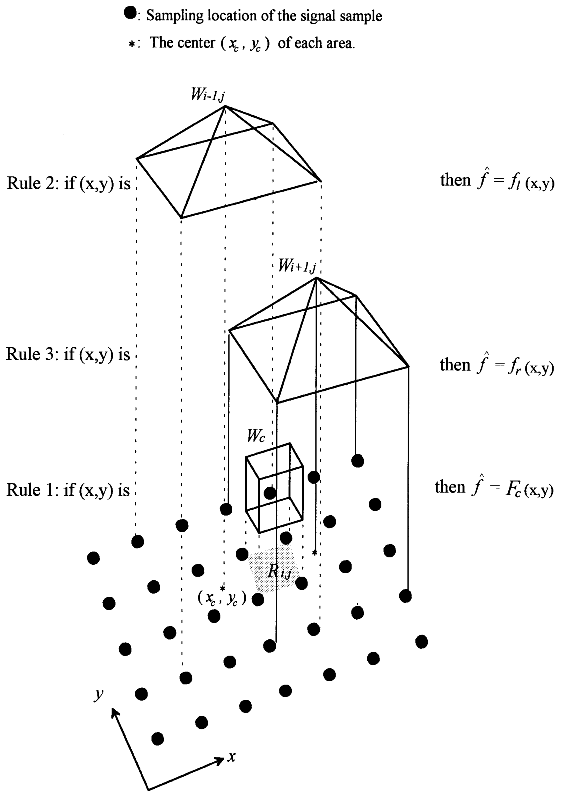

Fig. 5. Membership functions assigned to the main-surface and the sub-planes, and the fuzzy rules for interpolation in region $R_{i,j}$.

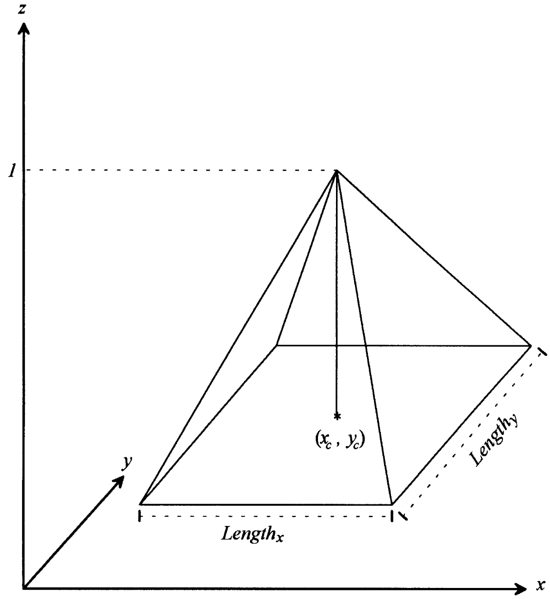

Fig. 6. The pyramid membership function $W_{i,j}(x,y)$.

Thus the interpolated value, $\hat{f}(x,y)$, in the region $R_{i,j}$ is given as follows:

$$\hat{f}(x,y) = \frac{W_{i-1,j}(x,y)f_{\mathrm{l}}(x,y) + W_{i+1,j}(x,y)f_{\mathrm{r}}(x,y) + 4F_{\mathrm{c}}(x,y) + W_{i,j-1}(x,y)f_{\mathrm{u}}(x,y) + W_{i,j+1}(x,y)f_{\mathrm{d}}(x,y)}{W_{i-1,j}(x,y) + W_{i+1,j}(x,y) + 4 + W_{i,j-1}(x,y) + W_{i,j+1}(x,y)},$$

$$x_i \leqslant x \leqslant x_{i+1}, \quad y_j \leqslant y \leqslant y_{j+1}, \quad i = 0,1,\ldots,M-2, \; j = 0,1,\ldots,N-2. \tag{18}$$

In (18), the weighting of the main-surface is four times greater than those of the sub-planes. This is to balance the effect of the main-surface with the effects of the remaining four sub-planes. Note that, at the edges of the images, we may eliminate some items from (18) in case G_1, G_2, or G_3 cannot be defined. For example, sub-plane $f_{\mathrm{l}}(x,y)$ and $W_{i-1,j}(x,y)$ must be eliminated from interpolation equation (18), when interpolating the left edge of the image; sub-planes $f_{\mathrm{l}}(x,y)$, $f_{\mathrm{u}}(x,y)$, and their corresponding weighting membership functions have to be ignored when interpolating the left-upper corner of the image.

3. Simulations and results

In order to confirm the validity of the proposed 2-D fuzzy rule-based inference interpolation algorithm, we conducted computer simulations on synthetic signals, as shown in the following examples. We used five simulations to test the proposed algorithm. In these simulations, performance in terms of reconstruction accuracy of the present algorithm was compared with bilinear interpolation and the cubic convolution interpolation proposed by Keys [8]. With respect to reconstruction accuracy, we used the mean square error (MSE) and the mean absolute error (MAE) as the comparison indices. These two error indices are given by

$$MSE=\frac{1}{MN}\left(\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}[f(i,j)-\hat{f}(i,j)]^2\right),\quad MAE=\frac{1}{MN}\left(\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}|f(i,j)-\hat{f}(i,j)|\right), \tag{19}$$

where $f(i,j)$ and $\hat{f}(i,j)$ represent the original and the resampled signals, respectively, and both of them consist of $M\times N$ pixel arrays.

Then, we applied the proposed algorithm to image resampling examples and compared the results with the other two interpolation algorithms.

3.1. Synthetic signal interpolation

3.1.1. Simulation 1

In this synthetic signal interpolation simulation, the original signal was given as follows:

$$S_1(x,y)=20x\exp(-x^2-y^2),\quad -2\leqslant x\leqslant 2,\ -2\leqslant y\leqslant 2. \tag{20}$$

A 45×45 point array $S_1(x_k,y_l)$, from which $k=0,1,\ldots,44$, $l=0,1,\ldots,44$, was created by uniformly sampling $S_1(x,y)$ in the defined region. The noisy point array $S_1'(x_k,y_l)$, was created by adding Gaussian noise of zero mean and unit variance to $S_1(x_k,y_l)$, and its *MSE* and *MAE* were 1.105 and 0.8, respectively. The 12×12 point sampled noisy data array $I_s(i,j)$ was obtained from $I_s(i,j)=S_1'(x_k,y_l)$, where $i=0,1,\ldots,11$, $j=0,1,\ldots,11$; $k=4i$, $l=4j$. We then put $I_s(i,j)$ with $N_x=4$ and $N_y=4$ into the proposed algorithm, to derive the resampling function $\hat{f}(x,y)$. And, putting $I_s(i,j)$ into the bilinear interpolation

Table 1
The MSEs for the resampling signals $\hat{f}(x,y)$ from the three interpolation algorithms applied in five simulations

	Noise	Fuzzy	Bilinear	Cubic
Simulation 1	1.015	0.355	0.549	0.696
Simulation 2	8.429	3.669	6.021	7.199
Simulation 3	31.596	6.420	7.825	9.987
Simulation 4	1.684	0.876	1.095	1.313
Simulation 5	0.966	0.394	0.504	0.656

Note: Simulations 2 and 3 were calculated based on the loss region only, and the others on the whole signal region, $-2\leqslant x\leqslant 2$, $-2\leqslant y\leqslant 2$.

Table 2
The MAEs for the resampling signals $\hat{f}(x,y)$ from the three interpolation algorithms applied in five simulations

	Noise	Fuzzy	Bilinear	Cubic
Simulation 1	0.801	0.479	0.591	0.668
Simulation 2	2.534	1.686	2.174	2.43
Simulation 3	5.515	2.311	2.489	2.764
Simulation 4	0.915	0.662	0.751	0.838
Simulation 5	0.789	0.500	0.565	0.649

Note: Simulations 2 and 3 were calculated based on the loss region only, and the others on the whole signal region, $-2\leqslant x\leqslant 2$, $-2\leqslant y\leqslant 2$.

and cubic convolution interpolation algorithms, we respectively obtained the corresponding resampling signals $\hat{f}(x,y)$. To compare reconstruction accuracy, the *MSEs* of the resampling signals $\hat{f}(x,y)$ obtained from the bilinear, cubic spline, and the proposed algorithm were 0.549, 0.696, and 0.355, respectively. And the *MAEs* were, respectively, 0.591, 0.668, and 0.479. The results of these algorithms are shown in Tables 1 and 2. From these *MSEs* and *MAEs*, we can see the reconstruction accuracy of the fuzzy rule-based interpolation algorithm was better than those of the other two interpolation algorithms. The proposed algorithm not only generated reconstructed signals from the region of interest but also kept track of pixel variation trends from the four surrounding sub-planes, and hence yielded better noise removal results.

3.1.2. Simulation 2

We assumed that the noisy signal $S_1'(x,y)$ was due to a complete loss of the original signal $S_1(x,y)$ in (20) from the following region, $\{(x,y)\,|\,-0.2\leqslant x\leqslant 0.25,\ -0.29\leqslant y\leqslant 0.16\}$. *MSEs* and *MAEs* for $\hat{f}(x,y)$ were calculated *only in the lost* 6×6 *region*, by the three interpolation algorithms, these results are also shown in Tables 1 and 2. From these two tables, it is evident that the proposed method demonstrated the best signal recovery ability.

3.1.3. Simulation 3

In this simulation, the noisy signal $S_1'(x,y)$ was also assumed to be due to the loss of signal $S_1(x,y)$ in the 6×6 region, $\{(x,y)\,|\,0.61\leqslant x\leqslant 1.06,\ -0.83\leqslant y\leqslant -0.38\}$. *MSEs* and *MAEs* for the resampled signal $\hat{f}(x,y)$ calculated only in the 6×6 lost region, are also shown in the two tables mentioned above. The fuzzy inference approach led to the smallest error.

3.1.4. Simulation 4

In Simulation 4, noisy signal $S_1'(x,y)$ suffered from losses in regions of Simulation 2 and 3, i.e., $\{(x,y)\,|\,-0.2\leqslant x\leqslant 0.25,\ -0.29\leqslant y\leqslant 0.16\}$ and $\{(x,y)\,|\,0.61\leqslant x\leqslant 1.06,\ -0.83\leqslant y\leqslant -0.38\}$. The *MSEs* and *MAEs* for the

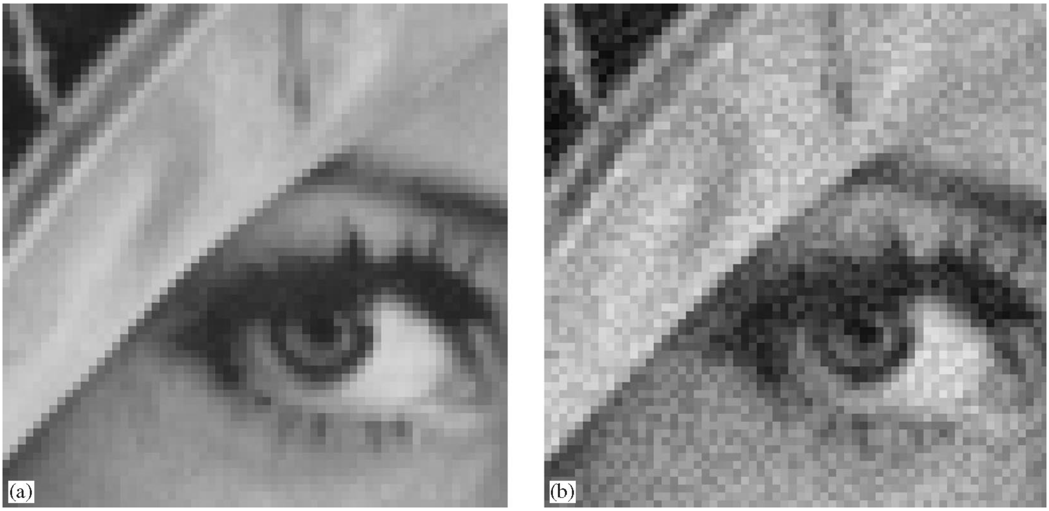

Fig. 7. Lena's eye images. The primary data was 64×64, with display dimensions of 256×256 using zero-order hold. (a) The original image; (b) The noisy image.

45×45 point array $\hat{f}(x, y)$ at $-2 \leqslant x \leqslant 2$ and $-2 \leqslant y \leqslant 2$ are also shown in Tables 1 and 2, respectively. The smallest error was still recorded by the proposed approach.

3.1.5. Simulation 5

In the last synthetic signal simulation, we considered another signal S_2, which was rougher than S_1, given by

$$S_2(x, y) = 3(1 - x)^2 \exp[-x^2 - (y + 1)^2] - 10(x/5 - x^3 - y^5) \exp[-x^2 - y^2]$$
$$- (1/3) \exp[-(x + 1)^2 - y^2], \quad -2 \leqslant x \leqslant 2, \ -2 \leqslant y \leqslant 2.$$

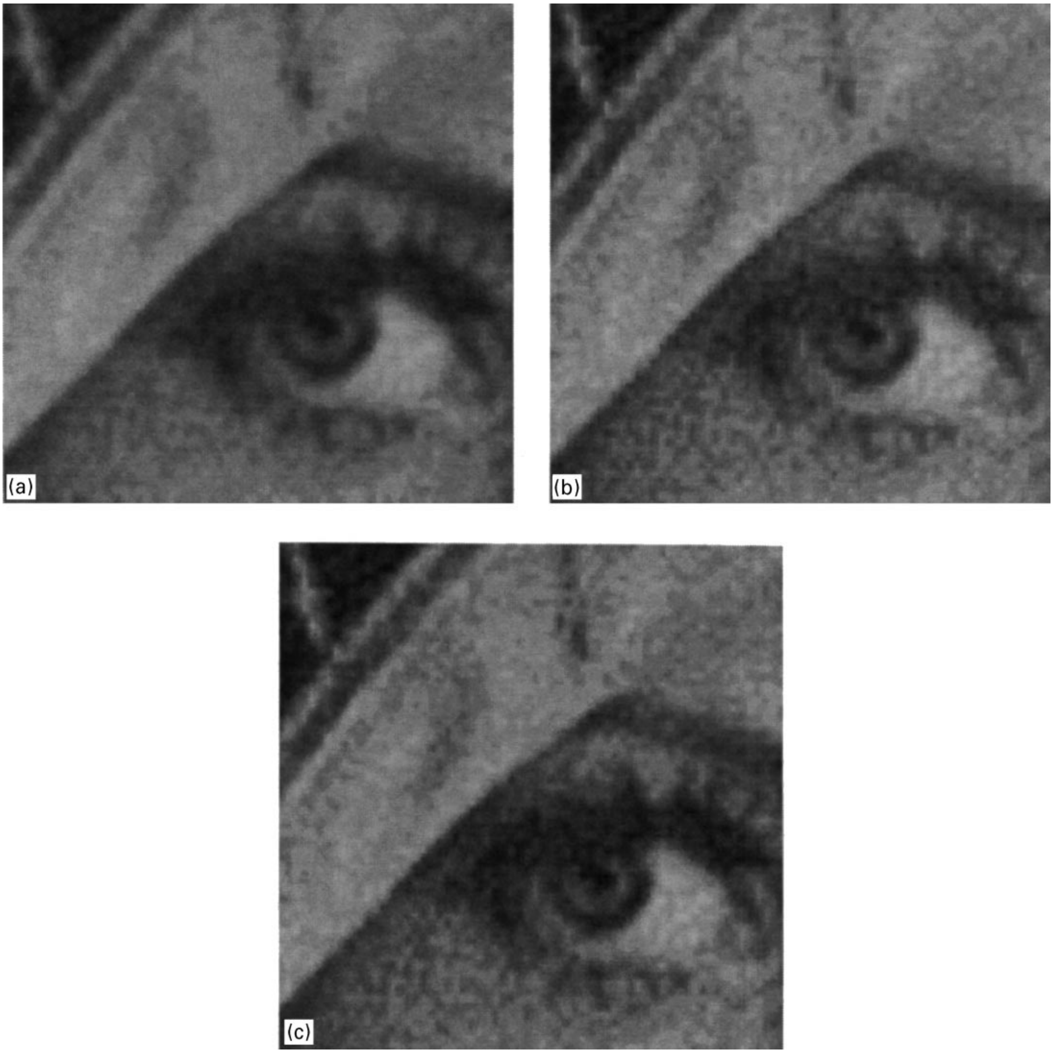

Fig. 8. Resampling using three different methods. (a) Fuzzy rule-based interpolation algorithm; (b) bilinear interpolation algorithm; (c) cubic convolution interpolation algorithm.

A 49×49 point array $S_2(x_k,y_l)$, where $k=0,1,\dots,48$, $l=0,1,\dots,48$, was also created by uniform sampling $S_2(x,y)$. And a 49×49 point noisy array $S_2'(x_k,y_l)$, was created similar to that in Simulation 1. The noisy signal's *MSE* and *MAE* are also shown in Tables 1 and 2. The 13×13 point sampled noisy data array $I_s(i,j)$ was sampled from the noisy point array $S_2'(x_k,y_l)$, i.e., $I_s(i,j)=S_2'(x_k,y_l)$, where $i=0,1,\dots,12$, $j=0,1,\dots,12$; $k=4i$, $l=4j$. The sampled noisy data was resampled to a 49×49 point array $\hat{f}(x,y)$, and the *MSEs* and *MAEs* for $\hat{f}(x,y)$ are also shown in Tables 1 and 2. As in the other simulations, the fuzzy inference approach produced the smallest errors.

In these simulations and their error measures, the proposed fuzzy rule-based interpolation algorithm demonstrated the best noise removal and signal recovery abilities.

3.2. Image resampling

2-D signal interpolation algorithms have a wide range of application areas in many image processing systems. For example, 2-D interpolation algorithms are used for image resampling to change the sizes of digital images interactively. To demonstrate the power of our algorithm on image resampling, we applied the 2-D interpolation scheme on a 256×256 image of Lena's eye. The result was compared with results obtained from the bilinear and cubic convolution interpolation algorithms. The original image, as shown in Fig. 7(a), had primary data of 64×64, with display dimensions of 256×256 obtained from zero-order hold. Fig. 7(b) shows the noisy image from zero-order hold, which was created by adding zero-mean Gaussian noise to the primary image at a signal-to-noise ratio (SNR) of 11.7 dB. Next, the 64×64 noisy image was put into the three algorithms and resampled to 256×256 images. The resampled images from the fuzzy rule-based interpolation, bilinear, and cubic convolution interpolation algorithms are shown in Figs. 8(a)–(c). Comparing these three figures, we can see that our algorithm removed noise better than the other two algorithms and outperformed the bilinear interpolation algorithm in reducing blocky effect. Moreover, the interpolated image of the proposed algorithm is comparable to that of the cubic convolution interpolation algorithm, in spite of the much simpler scheme used by our method, as compared to the third-order scheme of the cubic convolution algorithm.

4. Conclusion

An interpolation algorithm for 2-D signal using fuzzy rule-based inference has been proposed in this paper. The proposed interpolation scheme first finds a bilinearly interpolated main-surface that passes through four signal samples in the region to be interpolated, and finds four sub-planes, surrounding the interpolation region, that reflect the pixel tendencies of the interpolation region. Some fuzzy if–then rules for these five functions were presented for carrying out 2-D signal interpolation. Because of the inference rules employed for interpolation, the original signal can be interpolated and resampled very well despite the original signal having been corrupted by noise. When we applied the proposed algorithm to image resampling, the result was also very successful.

Appendix A. The deriving of $F_c(x,y)$

In this appendix, the equivalence of (2) and (5) is proved. From (4),

$$\begin{pmatrix} a\\ b\\ c\\ d \end{pmatrix} = \begin{pmatrix} x_i & y_j & x_i y_j & 1\\ x_i & y_{j+1} & x_i y_{j+1} & 1\\ x_{i+1} & y_j & x_{i+1} y_j & 1\\ x_{i+1} & y_{j+1} & x_{i+1} y_{j+1} & 1 \end{pmatrix}^{-1} \begin{pmatrix} I_s(x_i,y_j)\\ I_s(x_i,y_{j+1})\\ I_s(x_{i+1},y_j)\\ I_s(x_{i+1},y_{j+1}) \end{pmatrix}. \tag{A.1}$$

Since $N_x = x_{i+1} - x_i$ and $N_y = y_{j+1} - y_j$, the inverse of the matrix in (A.1) can be derived by

$$\begin{pmatrix} x_i & y_j & x_i y_j & 1 \\ x_i & y_{j+1} & x_i y_{j+1} & 1 \\ x_{i+1} & y_j & x_{i+1} y_j & 1 \\ x_{i+1} & y_{j+1} & x_{i+1} y_{j+1} & 1 \end{pmatrix}^{-1} = \begin{pmatrix} -y_{j+1}/N_xN_y & y_j/N_xN_y & y_{j+1}/N_xN_y & -y_j/N_xN_y \\ -x_{i+1}/N_xN_y & x_{i+1}/N_xN_y & x_i/N_xN_y & -x_i/N_xN_y \\ 1/N_xN_y & -1/N_xN_y & -1/N_xN_y & 1/N_xN_y \\ x_{i+1}y_{j+1}/N_xN_y & -x_{i+1}y_j/N_xN_y & -x_i y_{j+1}/N_xN_y & x_i y_j/N_xN_y \end{pmatrix}. \tag{A.2}$$

By substituting (A.2) into (A.1), we have

$$\begin{aligned} a &= -\frac{y_{j+1}}{N_x N_y} I_s(i,j) + \frac{y_j}{N_x N_y} I_s(i,j+1) - \frac{y_{j+1}}{N_x N_y} I_s(i+1,j) - \frac{y_j}{N_x N_y} I_s(i+1,j+1), \\ b &= -\frac{x_{i+1}}{N_x N_y} I_s(i,j) + \frac{x_{i+1}}{N_x N_y} I_s(i,j+1) + \frac{x_i}{N_x N_y} I_s(i+1,j) - \frac{x_i}{N_x N_y} I_s(i+1,j+1), \\ c &= \frac{1}{N_x N_y} I_s(i,j) - \frac{1}{N_x N_y} I_s(i,j+1) - \frac{1}{N_x N_y} I_s(i+1,j) + \frac{1}{N_x N_y} I_s(i+1,j+1), \\ d &= \frac{x_{i+1} y_{j+1}}{N_x N_y} I_s(i,j) - \frac{x_{i+1} y_j}{N_x N_y} I_s(i,j+1) - \frac{x_i y_{j+1}}{N_x N_y} I_s(i+1,j) + \frac{x_i y_j}{N_x N_y} I_s(i+1,j+1). \end{aligned} \tag{A.3}$$

By substituting the above four coefficients into $F_c(x, y) = ax + by + cxy + d$ and after a few mathematical manipulations, we have

$$\begin{aligned} F_c(x,y) = {} & \frac{(x_{i+1}-x)(y_{j+1}-y)}{N_x N_y} I_s(i,j) + \frac{(x_{i+1}-x)(y-y_j)}{N_x N_y} I_s(i,j+1) \\ & + \frac{(x-x_i)(y_{j+1}-y)}{N_x N_y} I_s(i+1,j) + \frac{(x-x_i)(y-y_j)}{N_x N_y} I_s(i+1,j+1). \end{aligned} \tag{A.4}$$

Since $x_{i+1} - x = N_x - (x - x_i)$ and $y_{j+1} - y = N_y - (y - y_j)$, it follows that

$$\begin{aligned} F_c(x,y) = {} & \frac{[N_x - (x-x_i)]}{N_x} \frac{[N_y - (y-y_j)]}{N_y} I_s(i,j) + \frac{[N_x - (x-x_i)]}{N_x} \frac{(y-y_j)}{N_y} I_s(i,j+1) \\ & + \frac{(x-x_i)}{N_x} \frac{[N_y - (y-y_j)]}{N_y} I_s(i+1,j) + \frac{(x-x_i)}{N_x} \frac{(y-y_j)}{N_y} I_s(i+1,j+1), \end{aligned} \tag{A.5}$$

which is exactly (5).

References

[1] K. Arakawa, Y. Arakawa, A nonlinear digital filter using fuzzy clustering, Proc. IEEE Internat. Conf. Acoust. Speech, Signal Process. IV, San Francisco, March 1992, pp. 309–312.

[2] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum Press, New York, 1981.

[3] T.C. Chen, R.J.P. deFigueiredo, Two-dimensional interpolation by generalized spline filters based on partial differential equation image models, IEEE Trans. Acoust. Speech, Signal Process. 33 (1985) 631–642.

[4] I. Enbustsu, K. Baba, N. Hara, Fuzzy rule extraction from a multilayered neural network, Proc. Internat. Joint Conf. Neural Networks, Seattle, July 1991, pp. 461–465.

[5] R.C. Gonzalez, R.E. Woods, Digital Image Processing, Addison-Wesley, Reading, MA, 1992.

[6] I. Hayashi, H. Nomura, N. Wakami, Construction of fuzzy inference rules by NDF and NDFL, Internat. J. Approx. Reason. 6 (1992) 241–266.

[7] H.S. Hou, H.C. Andrews, Cubic splines for image interpolation and digital filtering, IEEE Trans. Acoust. Speech, Signal Process. 26 (1978) 508–517.

[8] R.G. Keys, Cubic convolution interpolation for digital image processing, IEEE Trans. Acoust. Speech, Signal Process. 29 (1981) 1153–1160.

[9] R. Krishnapuram, J.M. Keller, Fuzzy set theoretic approach to computer vision: an overview, Proc. IEEE Internat. Conf. Fuzzy Systems, San Diego, March 1992, pp. 135–142.

[10] C.C. Lee, Fuzzy logic in control systems: fuzzy logic controller – Part I and Part II, IEEE Trans. Systems Man, Cybernet. 20 (1990) 404–435.

[11] J.S. Lim, Two-Dimensional Signal and Image Processing, Prentice-Hall, Englewood Cliffs, NJ, 1990.

[12] C.T. Lin, C.S.G. Lee, Real-time supervised structure/parameter learning for fuzzy neural network, Proc. IEEE Internat. Conf. Fuzzy Systems, San Diego, March 1992, pp. 1283–1291.

[13] S.K. Pal, R.A. King, Image enhancement using smoothing with fuzzy sets, IEEE Trans. Systems Man, Cybernet. 11 (1981) 494–501.

[14] S.K. Pal, S. Mitra, Multilayer perceptron, fuzzy sets, and classification, IEEE Trans. Neural Networks 3 (1992) 683–697.

[15] S.K. Park, R.A. Showengerdt, Image reconstruction by parametric convolution, Comput. Vision, Graphics, Image Process. 23 (1983) 258–272.

[16] W.K. Pratt, Digital Image Processing, Wiley, New York, 1991.

[17] F. Russo, G. Ramponi, Combined FIRE filters for image enhancement, Proc. IEEE Conf. Fuzzy Systems I, Orlando, June 1994, pp. 260–264.

[18] M. Sugeno, An introductory survey of fuzzy control, Inform. Sci. 36 (1985) 59–83.

[19] E. Uchino, T. Yamakawa, T. Miki, S. Nakamura, Fuzzy rule-based simple interpolation algorithm for discrete signal, Fuzzy Sets and Systems 59 (1993) 259–270.

[20] M. Unser, A. Aldroub, M. Eden, Fast B-spline transforms for continuous image representation and interpolation, IEEE Trans. Pattern Anal. Mach. Intell. 13 (1991) 277–285.