



# Grain boundary detection in microstructure images using computational intelligence

Orhan Dengiz<sup>a,\*</sup>, Alice E. Smith<sup>a</sup>, Ian Nettleship<sup>b</sup>

<sup>a</sup> Auburn University, Department of Industrial and Systems Engineering,  
207 Dunstan Hall, Auburn, AL 36849, USA

<sup>b</sup> University of Pittsburgh, Department of Materials Science and Engineering,  
Pittsburgh, PA 15261, USA

Received 1 December 2004; received in revised form 31 March 2005; accepted 31 May 2005  
Available online 29 September 2005

## Abstract

Two computational intelligence approaches, a fuzzy logic algorithm and a neural network (NN) algorithm, for grain boundary detection in images of superalloy steel microstructure during sintering are presented in this paper. The images are obtained from an optical microscope and are quite noisy, which adversely affects the performance of common image processing tools. The only known way to accurately determine the grain boundaries is digitizing by hand. This is a very time-consuming process, causes operator fatigue, and it is prone to human errors and inconsistency. An automated system is therefore needed to complete as much work as possible and we consider a fuzzy approach and a neural approach. Both methods performed better than the widely available standard image processing tools with the neural approach superior on images similar to those trained while the fuzzy approach showed more tolerance of disparate images.

© 2005 Elsevier B.V. All rights reserved.

*Keywords:* Grain boundary detection; Fuzzy logic; Neural networks; Image processing

## 1. Introduction

### 1.1. Sintering

Sintering is a process of forming objects from metal, ceramic or composite powders by applying heat

at a temperature below the material's melting point for a certain period of time. To achieve parts with excellent creep life, fatigue resistance, high strength and durability, sintering is accompanied by high pressure (hot isostatic press or HIP), which greatly improves the final product but also increases the cost of manufacturing significantly. Hipping is a near-net shape manufacturing process. More than 97% of the raw material is used in the final product [11]. The compacted powder deforms during the sintering and

\* Corresponding author. Tel.: +1 334 524 1187;  
fax: +1 334 844 1381.

E-mail address: [dengior@auburn.edu](mailto:dengior@auburn.edu) (O. Dengiz).

hipping processes. This deformation is difficult to estimate because it is often anisotropic shrinkage, meaning, the deformation is not uniform in every direction [1]. Anisotropic shrinkage results in functional degradation of the product, proportional to the amount of deformation. If the shrinkage in the final product is overestimated, raw material will be wasted. If it is underestimated, the final product will be useless and may be scrapped. The mechanics of sintering have been studied for a long time and there are two different types of models: macroscopic and microscopic. Macroscopic models cannot represent the complex three-dimensional mechanics of particles and therefore are incapable of making accurate deformation predictions, resulting in high scrap rates. The microscopic approach has roots in sintering theory and concentrates on the local behavior of particle necks [1]. A better understanding of micro-structural mechanics will allow development of good prediction models for final product size estimation, which will increase production efficiency. Microscopic images of sintered material, taken during the sintering process, are very useful in understanding the dynamic behavior, or rearrangement of metal powder particles. In this study, the digital images of interest are the microscopic images of the sintered specimen. These images are to be used for the analysis of particle rearrangements, or particle cluster formations, and for calculations such as grain size distributions. Understanding how the metal particles behave individually and as a group will help explain the sintering phenomena. The first step of these analyses is the detection of grain boundaries, that is, converting the image to a binary grain boundary map.

### 1.2. Digital image processing

A gray-scale digital image is composed of discrete points of gray tones, or brightness, rather than continuously varying tones. A natural image is divided into a number of individual points of brightness, and in addition, each of those points is described by a digital data value. Each brightness point is a pixel of the digital image. A pixel is the most basic element of any digital image. The pixels of an image form a rectangular array. Each pixel has a coordinate  $(x, y)$  that corresponds to its location within the image,  $x$  being the vertical component, and  $y$  the horizontal

component. In general  $(0, 0)$  is the upper left corner of the image. For 256 gray-tone images, a pixel can have one of the 256 brightness values, ranging from 0 to 255. Black is represented by 0, and white is represented by 255.

Image processing is the management or manipulation of the image data to meet a user's needs [12]. Two different purposes of image processing can be stated as; improving the visual appearance to a human viewer, and preparing images for measurement of the features and structures present. The latter is the group that the algorithms presented in this study fall into. Modern fast computers with high memory capabilities are able to handle large amounts of data, which is usually needed for image processing applications. Image processing does not reduce the amount of data present but simply manages or rearranges it.

### 1.3. Problem definition

The images that are studied in this paper are of a magnified specimen. An optical microscope was used to acquire the images and the resulting images have sizeable variation of brightness and contrast within the image itself, along with high noise levels and local imperfections, which complicate the processing. Despite being easily identifiable by the human eye, it is almost impossible to accurately extract grain boundaries using readily available conventional tools, e.g. edge detection or threshold filters, in commercially available image processing software. A special tool is therefore needed for effective grain boundary detection.

The example image given in Fig. 1 clearly shows the uneven brightness and insufficient contrast. High level of noise and local imperfections can also be observed. There are several sources of local imperfections including optical problems (e.g. uneven lighting, physical obstructions) and external conditions such as dust in the environment. Noise, which is present throughout the entire image, is usually caused by the equipment and environment properties that are inherent to the experiment setup. For these images, a traditional or simple thresholding interval image processing approach performs very poorly. In the simple thresholding method, the user defines, or selects interactively, a certain threshold value, or a threshold interval. The pixels of the digital image with

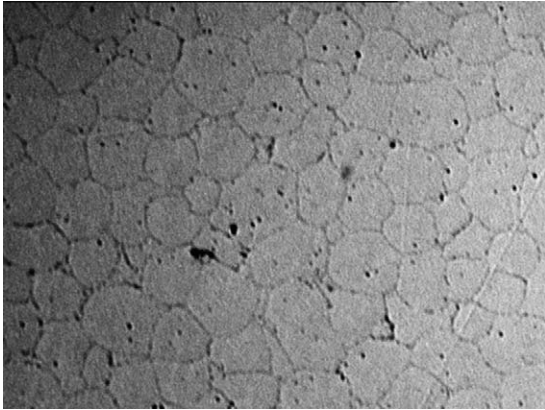


Fig. 1. An example image before processing.

a higher value than the threshold are extracted in the second step. In the case of a threshold interval, pixels that have a grayscale value within the interval are extracted. This method only works if the regions of interest in the image fall above, or below, a certain grayscale value, or fall within a certain interval.

#### 1.4. Computational intelligence in digital image processing

There have been a number of relevant approaches in the literature using computational intelligence and microscopic material images. Bigand et al. [3] developed a linear edge extraction algorithm that locates the edges and extracts line segments with a clustering method. Their algorithm detects linear shapes and then applies a clustering method to trace best linear edges. Friel et al. [5] developed an algorithm to detect grain boundaries accurately. They use thresholding to detect boundaries, followed by a post-processing step, which is a higher-level boundary completion process to complete the missing sections. Zaknich [16] used a neural network (NN) to isolate and identify aluminum hydroxide particles. He first preprocesses the image by smoothing for noise removal. The smoothed image is filtered by a NN and then converted to binary by thresholding. He then applies binary processing to eliminate smaller objects. Sinha and Karray [14] proposed a neuro-fuzzy algorithm for underground pipe inspection. Their algorithm was designed to recognize and classify pipe cracks, which appear as linear structures in inner-pipe images that are acquired by robots equipped with

digital cameras. Augusteijn and Clemens [2] used a NN to capture boundaries between different textured surfaces in a digital image. They trained the NN on boundaries between different textures and showed its ability to generalize to other textures that were not a part of the training set. Kim and Cho [9] presented a combined NN and fuzzy logic application to boundary detection in noisy images. They used a specially designed NN to perform fuzzy reasoning operations. They made use of spatial relationships among neighboring edges around the edge segments of interest to improve detection performance.

## 2. Fuzzy approach to boundary detection

### 2.1. Fuzzy logic

Fuzzy logic was developed and popularized by Zadeh [15]. He developed a variation of traditional set theory and binary logic that made it possible to model complex and imprecise systems. Fuzzy logic applications are used in control of mass transportation systems, automatic cookers, decision support systems, data mining and classification applications, and in many more areas. The members of fuzzy sets are defined by fuzzy membership functions. A fuzzy membership function measures the degree of membership to a fuzzy set and is normally between [0,1]. The fuzzy rules that are activated for a given instance of input values determine the final output value that is calculated by a chosen defuzzification method. Detailed information on fuzzy logic can be found in [4] and [13].

### 2.2. Fuzzy logic application to grain boundary detection

The fuzzy logic grain boundary detection algorithm (FLGBD) developed in this paper traces the entire image, pixel by pixel, and detects the pixels that belong to a boundary. The decision whether the point is on a boundary or not is made by evaluating that pixel's and its neighborhood's properties by using a fuzzy system. The pixel to be classified will be referred to as the *classified pixel* from this point on.

Fig. 2 represents a classified pixel and its neighborhood. The rule-base for classification is based on a few general properties of boundaries; first,

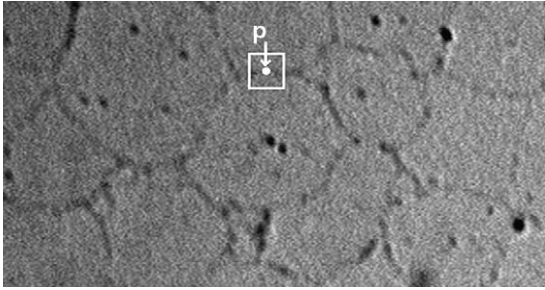


Fig. 2. Illustration of a classified pixel ( $p$ ) and its rectangular neighborhood.

the boundaries are always darker relative to their local neighborhood. Second, they should have adjacent “boundary” pixel groups at least within a single direction, at some angle. Another property is that both sides of a boundary towards its orthogonal directions have brighter pixels than the boundary pixel itself.

The stronger these properties for a pixel in the image are, the more likely that it belongs to a boundary. FLGBD quantifies the properties discussed above for every pixel to estimate the degree to which it belongs to a boundary. FLGBD uses three input variables to calculate one output variable.

2.2.1. Input 1: relative brightness

Pixels that belong to a boundary are darker relative to their local neighborhood. FLGBD concentrates on a local square region of  $21 \times 21$  pixels around the classified pixel. This size is selected because it is small enough that the change in image brightness is negligible, but large enough to rate the relative darkness of the classified pixel. A relative brightness ratio for the classified pixel at  $(i, j)$  is calculated by dividing its gray value to the average gray value of its neighborhood. This measure is the first input variable of FLGBD and it is given in Eq. (1).

$$\text{Input 1 } (i, j) = \frac{A(i, j)}{\left[ \sum_{k=i-((n-1)/2)}^{i+((n-1)/2)} \left( \sum_{l=j-((n-1)/2)}^{j+((n-1)/2)} A(k, l) \right) / n^2 \right]} \quad (1)$$

where  $A(x, y)$  is the gray value of the pixel at coordinates  $(x, y)$ ;  $n$  is the size of one side of square neighborhood ( $n = 21$ ).

The smaller the value of relative brightness is, the more likely the classified pixel belongs to a boundary. The membership functions for Input 1 were defined at five levels and their values were determined from preliminary experiments. The input 1 membership functions are as follows: *high* [0.0 0.0 0.63 0.82], *med high* [0.74 0.86 0.95], *med* [0.87 0.95 1.03], *med low* [0.99 1.07 1.11], *low* [1.08 1.16 2.0 2.0] and are shown in Fig. 3. (The trapezoidal functions are represented by four vertices and the triangular membership functions are represented by three vertices.)

2.2.2. Input 2: number of “dark” neighbors

A boundary is theoretically continuous, so any boundary point should have adjacent boundary points along some direction that are darker than the average color of its neighborhood. If there is a pattern of darker pixels adjacent to the classified pixel along some certain direction, it is very likely that the classified pixel belongs to a boundary. This is similar to co-occurrence measures, introduced by Haralick et al. [7]. Co-occurrence methods measure the frequencies of the occurrence of certain gray levels in pixels in different directions. Input 2 is the number of darker pixels around the classified pixel, calculated by searching eight different linearly directed search regions as shown in Fig. 4. The width of the directed search regions are set at three pixels, which is the average width of the boundaries in the source images.

Fig. 5 shows the area around the classified pixel ( $p$ ) used to search for adjacent dark pixels, which is an  $11 \times 11$  square region. This size is used when counting the number of darker adjacent pixels because boundary lines deviate from the linear search strips beyond this size area. Fig. 5b and c show the horizontal strips and a diagonal strip around  $p$ . If a pixel inside the gray region

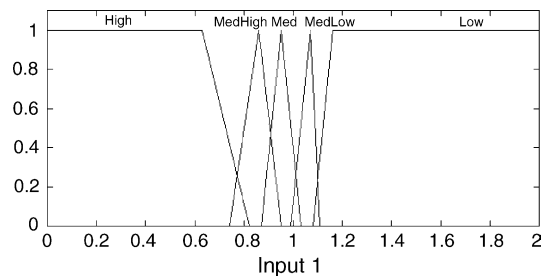


Fig. 3. Membership functions for Input 1.

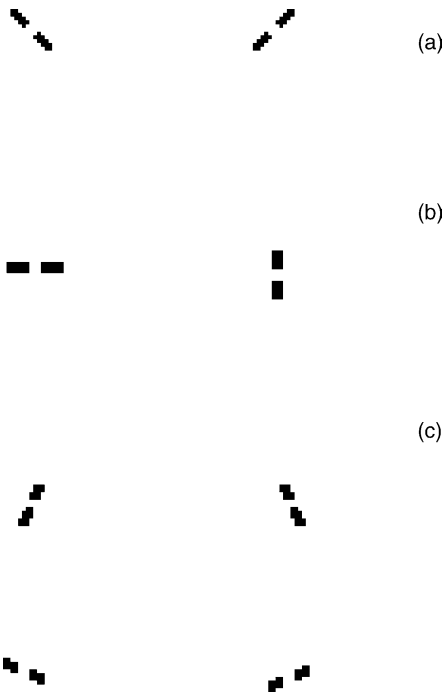


Fig. 4. The eight different strip regions around the pixel to be classified. (a) diagonal strips, (b) orthogonal strips, (c) inclined strips. The classified pixel is at the middle of each strip.

is darker than the average brightness, it is counted as a dark neighbor pixel. The numbers of dark pixels for all eight directions are calculated individually and the largest is set as the value of Input 2. Membership functions for Input 2 are determined at three levels as follows; *low* [0.0 0.0 10.2 12.2], *medium* [10.6 15.1 21.8 24.7], *high* [23.0 25.7 34.0 34.0], and are shown in Fig. 6.

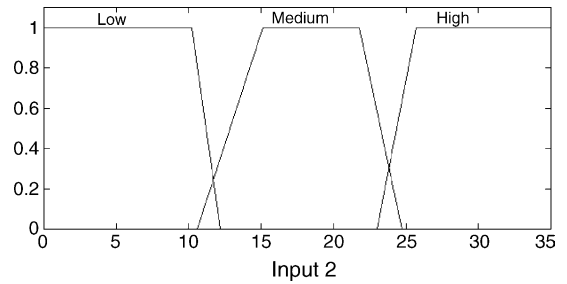


Fig. 6. Membership functions for Input 2.

### 2.2.3. Input 3: weighted average relative darkness

Input 3 is the ratio of the weighted average gray value of a smaller neighborhood of the classified pixel to the average of its region. The weighted average is calculated by averaging the gray values of the pixels in a  $5 \times 5$  square region around the classified pixel, giving more weight to adjacent pixels. Input 3 becomes larger for pixels surrounded by dark pixels and it becomes small for stand-alone dark pixels. It is designed to identify local imperfections in the images and helps reduce false-recognition of them as boundaries. Matrix  $W$  contains the weights that are used to calculate the weighted average brightness of the  $5 \times 5$  region around the classified pixel. Weights are determined by assigning the highest value to adjacent ones, and evenly decreasing as distance increases. The weighted average is calculated according to the equation given in Eq. (2). The membership functions are defined at four levels as follows: *high* [0.0 0.0 0.8 0.9], *medium high* [0.85 0.95 1.0], *low* [0.98 1.05 1.11], *very low* [1.05 1.2 2.0 2.0], and are shown in Fig. 7.

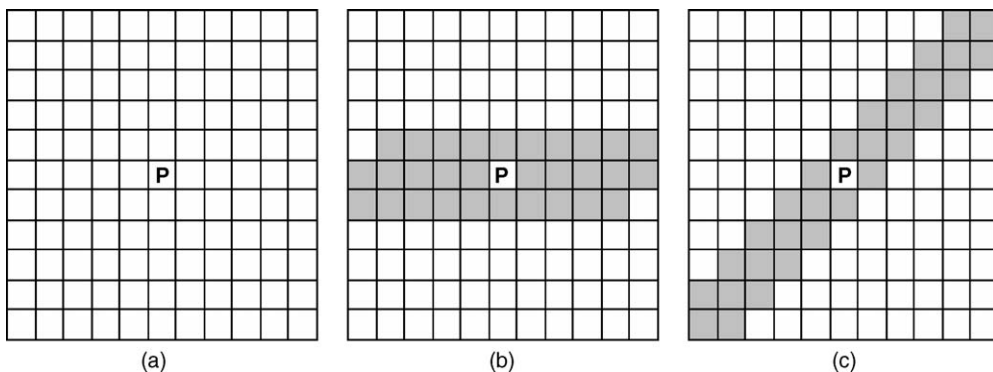


Fig. 5. Detailed representations of the local search region for darker neighbors for pixel  $p$  (a), the horizontal search direction (b), and the diagonal search direction (c).

$$\text{Input 3 } (i, j) = \frac{\sum_{k=i-((n-1)/2)}^{i+((n-1)/2)} \sum_{l=j-((n-1)/2)}^{j+((n-1)/2)} [W(k-i+2, l-j+2) \times A(k, l)]}{\sum_{k=i-((r-1)/2)}^{i+((r-1)/2)} \left( \sum_{l=j-((r-1)/2)}^{j+((r-1)/2)} A(k, l) \right) / r^2} \times \left[ \sum_{k=1}^n \sum_{l=1}^n W(k, l) \right] \quad (2)$$

where,

$$W = \begin{bmatrix} 1 & 2 & 3 & 2 & 1 \\ 2 & 4 & 5 & 4 & 2 \\ 3 & 5 & 0 & 5 & 3 \\ 2 & 4 & 5 & 4 & 2 \\ 1 & 2 & 3 & 2 & 1 \end{bmatrix}$$

A(x, y) is the gray value of the pixel at coordinates (x, y); n is the size of one dimension of square weight matrix (n = 5); r is the size of one dimension of the square region used for average gray value calculation (r = 21).

### 2.2.4. Output

The output is constructed at four levels; *low*, *medium*, *medium-high* and *high*. Every combination of input levels is assigned to an output level as explained in the next section. FLGBD uses the centroid defuzzification method, which is the geometric centroid of the output envelope. The output of FLGBD is designed to correlate with the degree that the classified pixel belongs to a boundary. Boundary points are expected to have higher output values than non-boundary points. A threshold value for output classification of 0.65 was established after some experimental runs. A pixel with an output >0.65 is recognized as a “boundary point”. A pixel with an output <0.60 is recognized as a “non-boundary point”. Pixels that fall between 0.65 and 0.60 are recognized as boundaries if

they are adjacent to another boundary point. The membership functions for the Output are defined at four levels as follows: *low* [0.0 0.03 0.38 0.48], *medium* [0.41 0.48 0.52 0.62], *medium high* [0.56 0.64 0.73], *high* [0.65 0.74 1.05 1.45] and are shown in Fig. 8.

### 2.2.5. Rule base

The rule base is formed by combining inputs by the AND (min) operator to produce an output response. All possible combinations of the different levels of the three inputs and the corresponding output levels are constructed. Table 1 shows the rule base and the descriptions of the variable levels are given in Table 2. In Table 1, each number refers to a fuzzy set for all three input variables and the output variable. The first column lists the five different fuzzy sets for Input 1, and the two rows above the mid section list three different fuzzy sets for Input 2 and four different fuzzy sets for Input 3, respectively. The middle section in Table 1 shows the fuzzy set that the output variable is a member of, for all possible combinations of the input variables’ levels. Table 2 shows the variable levels associated with the numbers used in Table 1. As an example, for a pixel to be classified, if Input 1 is “Medium High” (level 2), Input 2 is “Medium” (level 2), and Input 3 is “High” (level 1), then the level of Output is read from the mid-section of Table 1 as “Medium High” (level 3).

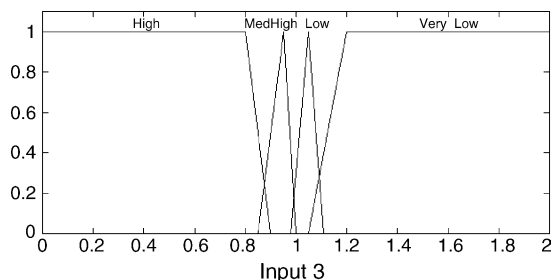


Fig. 7. Membership functions for Input 3.

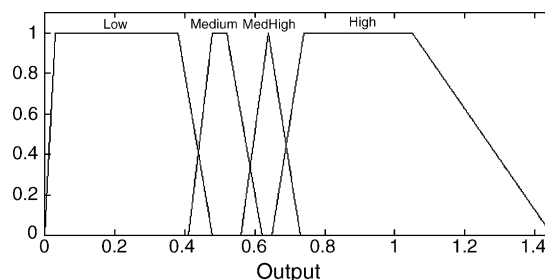


Fig. 8. Membership functions for the Output.

Table 1  
The FLGBD rule base

Input 1	Input 3												
	1	1	1	2	2	2	3	3	3	4	4	4	
	Input 2												
	1	2	3	1	2	3	1	2	3	1	2	3	
													Output
1	3	4	4	2	3	4	1	1	2	1	1	1	
2	3	3	4	2	3	3	1	1	2	1	1	1	
3	2	3	3	3	3	4	1	2	2	1	1	1	
4	1	2	4	1	2	4	1	1	1	1	1	1	
5	1	1	2	1	1	1	1	1	1	1	1	1	

### 3. Neural approach to boundary detection

Neural networks are connectionist systems, which consist of many primitive cells that work in parallel and are connected via directed links (or connections). Each link has a weight, which is initially random and converges to a final value at the end of an iterative training process. The processing principle is the transfer of the activation patterns across the links. The biological inspiration is the mechanism of the human brain, where information processing is done by transfer of activations from one group of neurons to others through synapses. NN's are well suited for pattern recognition and classification [10].

A typical feed forward NN has an input layer, to which the input data enters the NN, one or two hidden layers and an output layer. Hidden layers are required to define the non-linear relationships between the input and output layers. Generally more than two hidden layers are unnecessary [13,14]. Supervised training requires the set of input data and the corresponding output data. The network is able to learn the relationship between the input and output pairings by changing the weights to reach a smaller average output error. One of the most important properties of a NN is its generalization ability [6,2]. NN's are also known to perform well under the presence of noise [13].

Table 2  
Description of input and output levels

Level	Input 1	Input 2	Input 3	Output
1	High	Low	High	Low
2	Medium high	Medium	Medium high	Medium
3	Medium	High	Low	Medium high
4	Medium low		Very low	High
5	Low			

In this section of the paper, an alternative computational intelligence approach using a fully connected feed forward NN for grain boundary detection is presented, referred to as NNGBD. NNGBD is designed to accept the local neighborhood information of the classified pixel as inputs and learn the relation to pixel classification. NNGBD is trained by a standard backpropagation [4] method. This is a supervised training method, which requires the correct classifications of the pixels in the training set, done here by a human operator.

As an image processing application, the amount of data that is managed by the NN is enormous. An image that is 640 pixels by 480 pixels has a total of 307,200 pixels to be classified. The classification of boundary pixels in the training set to form the output values, i.e. the expert tracing of grain boundaries, is done by a human operator. Fig. 9 shows the expert detection of the grain boundaries for the image in Fig. 1. The binary images showing the boundaries that are created by the expert are referred to as the *expert outputs* for the rest of this paper.

Two fundamentally different approaches for NN input were investigated. The first is to preprocess the image data to calculate some descriptive measures and use them as inputs to the NN. Calculation of the descriptive measures is very similar to the FLGBD input calculation methods that were presented earlier. The second approach is to use the raw data, i.e. the gray values themselves, directly as inputs to the NN. This approach expects the NN to understand the

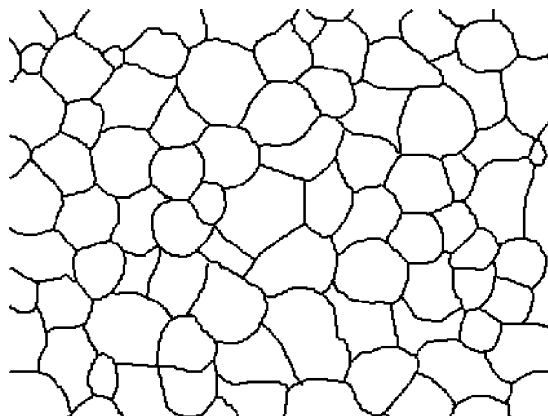


Fig. 9. Expert output traced by a human operator for the image in Fig. 1.

characteristic properties of boundary pixels, and also requires a larger network (more inputs and more trainable weights). The output value used for the training in both approaches is the binary value in the expert output that corresponds to the coordinates of the classified pixel.

### 3.1. First input approach for NN

The first approach was to generate the input data to train the NN similarly to the FLGBD. As with FLGBD, a square neighborhood region of size  $21 \times 21$  pixels, and strip regions in eight different directions were taken around each pixel to be classified. Descriptive properties of the classified pixel and its surroundings that are used as inputs to the NN are; the pixel's gray value, the average square neighborhood's gray value, the number of pixels in the eight strips that are darker than the average square neighborhood gray value, and the variances, the maximum, and the minimum values of the grayscale values of the surrounding strips. Although these properties were thought to be good indicators for boundary points, this approach (rather surprisingly) did not perform as well as the second approach.

### 3.2. Second input approach for NN

In this approach, the raw gray tone values from the image are fed into the NN as inputs. Use of the raw data has been tried before in some studies (e.g., [16]).

A square region of size  $n \times n$ , having the classified pixel at the center, defines the input vector to the NN. The input layer has  $n^2$  nodes, and input values are calculated by scaling the gray values of the corresponding pixels by the neighborhood average and subtracting 1.0, as shown in Eq. (3). Subtracting 1 yields negative values for the pixels that are darker than average and positive values for those that are brighter.

$$\text{Input} = \frac{\text{PixelValue}}{\text{RegionAverage}} - 1.0 \quad (3)$$

where *RegionAverage* is the average grayscale value in the a  $21 \times 21$  pixel square region with the classified pixel at the center.

Fig. 10a and b show two examples that represent the expert outputs for corresponding center pixels (*P*). Gray regions show the pixels that are recognized as boundaries by the human operator and they have an output value of 1. Non-boundary points have an output value of 0. Point *P* in Fig. 10a, as a boundary, has an expert output value of 1, and point *P* in Fig. 10b has an expert output value of 0.

Experimental runs showed that the second approach trained and performed much better than the first one for NNGBD. Many different NN architectures were tested including smaller and larger input and hidden layers and the following settings were chosen as the final topology:

*Input:* Although a square region of size  $21 \times 21$  pixels is used to calculate the average grayscale value

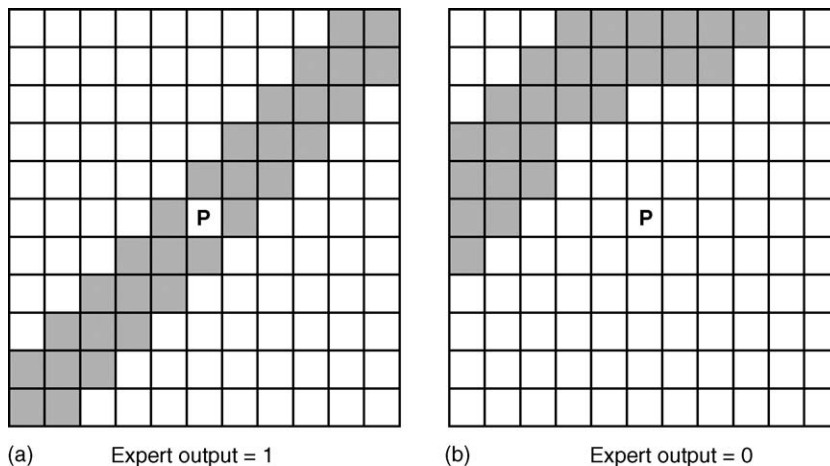


Fig. 10. (a) with an expert output of 1 and (b) with an expert output of 0.

around the classified pixel, using the same size region for the NN input resulted in too many input nodes, which adversely affects the NN performance. Thus, a smaller square input region of size  $11 \times 11$  pixels is used for the NN input, with the classified pixel at its center. This corresponds to a 121 node input layer. It is large enough to contain a representative portion of a boundary line, and small enough to allow effective and fast training. *Hidden Layer 1*: The first hidden layer has 121 nodes, which is the same size as the input layer. *Hidden Layer 2*: A 25 node hidden layer was used as the final hidden layer. *Output Layer*: An output layer of size 1 is used. Since output values used to train the NNGBD are binary the output is also converted into binary. A threshold value is set and output values less than the threshold are classified as non-boundaries, and others are classified as boundary pixels. A value of 0.6 was found to be a good threshold after experimenting with different values.

### 3.3. Data sets

Since there is a huge number of data points, 20,000 of them were randomly selected and used to train the NNGBD. A separate set of 1000 data points was used for validation during training. Training lasted for 1000 epochs with a learning rate of 0.1. After 1000 epochs, both the training set mean squared error (M.S.E.) and the validation set M.S.E. reached steady levels at 0.03 and 0.08, respectively. The NNGBD algorithm was also tested with images that did not contain any of the training or the validation data.

### 3.4. Post processing

Outputs of the NNGBD were refined by the use of binary image manipulation tools. Different algorithms are commonly available for noise reduction and boundary repair. Tools such as noise filtering, closing and edge thinning, or skeletonizing are used after the NN application to refine the final output [5]. Noise reduction, opening and closing are rank filters. Rank filters rank or sort a predefined number of pixels around a center pixel and replace the center pixel with the median, minimum, or maximum value [8]. In this paper, eight pixels around the center pixel, in a  $3 \times 3$  region are used for rank filters. For binary images, there are rank filters, or rank related filters, named in

ScionImage software as; dilation (D), erosion (E), and skeletonization (S). Dilation adds a black pixel if four or more of its eight neighbors are black. Dilation connects discontinuous objects and fills in holes. Erosion removes a black pixel if four or more of its eight neighbors are white. Erosion separates objects that are touching and removes isolated pixels. Skeletonization repeatedly removes pixels from the edges of boundary lines until they are reduced to single pixel wide skeletons. The best order and the required number of repetitions of these binary image operators can differ from one application to other. Selection and use of post-processing tools require an interactive operation, which can be done by a human operator. The following order of these operators perform the best in post-processing the binary output files; D, E, D, D, D, E, S.

## 4. Results

The graphical results of FLGBD, NNGBD, and conventional thresholding tools are shown in Figs. 11–14. The results of the thresholding method are the best results that could be generated using the thresholding tool interactively. The superior performance of the proposed algorithms to the conventional tools are clearly visible. For the images studied in this paper, NNGBD performed better than FLGBD. The disadvantage of NNGBD is that it requires the images to be processed be similar to the images used for training, i.e. they should be acquired under same conditions using the same equipment. The poor performance of NNGBD that is observed in Fig. 12 is due to the different character of the source image from the images used to train the NN. Fig. 12a shows an image on which the FLGBD algorithm performs better than NNGBD. It is necessary to emphasize that the quality of the input image is much higher than other ones studied in this paper. It has less variation in brightness and the noise level is low. For the source image in Fig. 12a, outputs of simple thresholding by hand, the FLGBD and the NNGBD algorithms are given in Fig. 12b–d, respectively. FLGBD performed better than the other two in this case. Figs. 13 and 14 show the effects of post processing (described in Section 3.4) for those images processed by the NNGBD.

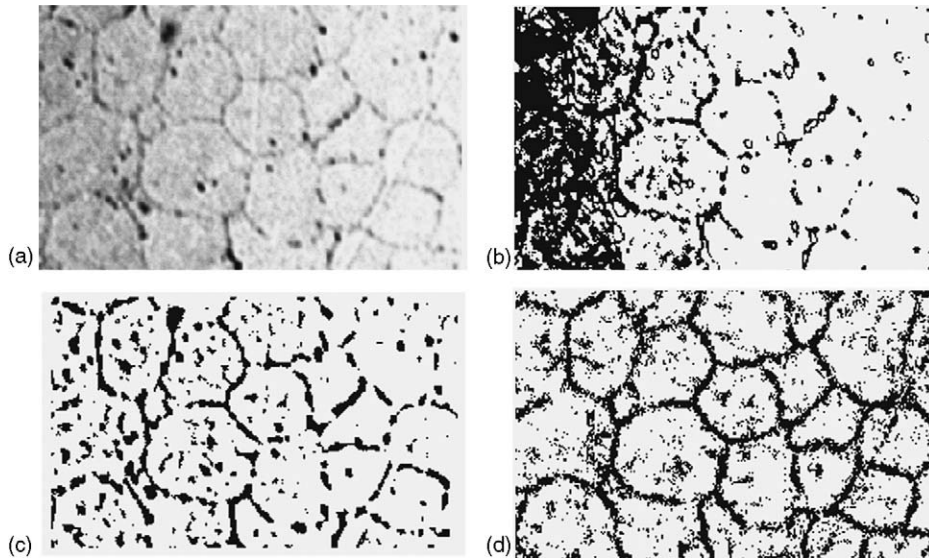


Fig. 11. An example smoothed and contrast enhanced source image (a), output of simple thresholding by hand (b), outputs of the FLGBD algorithm (c), and the NNGBD algorithm (d).

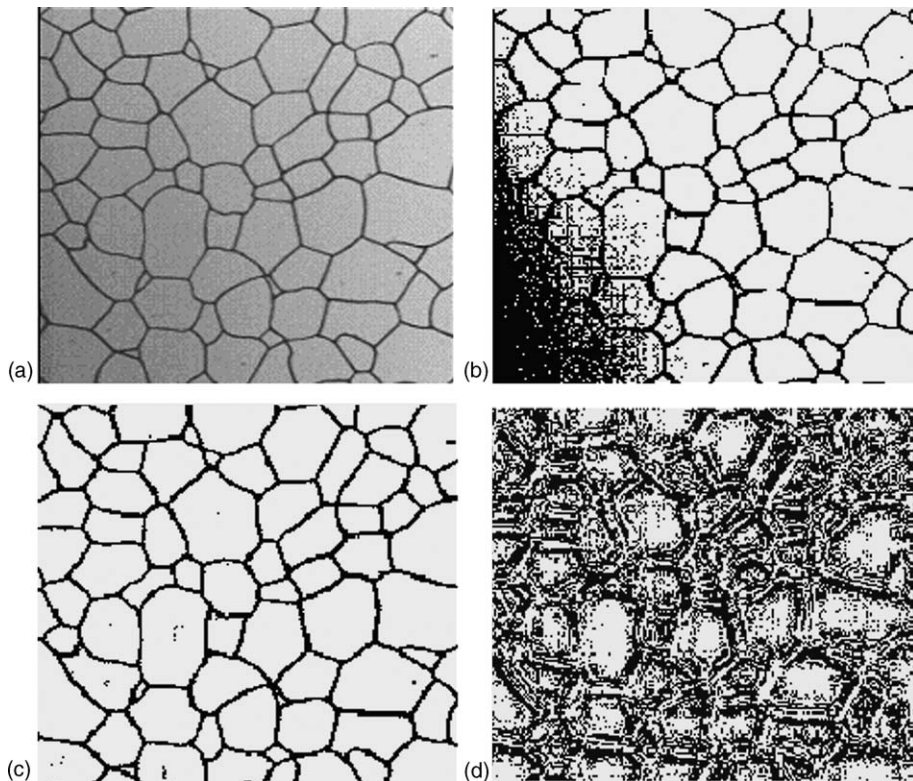


Fig. 12. An example source image (a), output of simple thresholding by hand (b), outputs of the FLGBD algorithm (c), and the NNGBD algorithm (d).

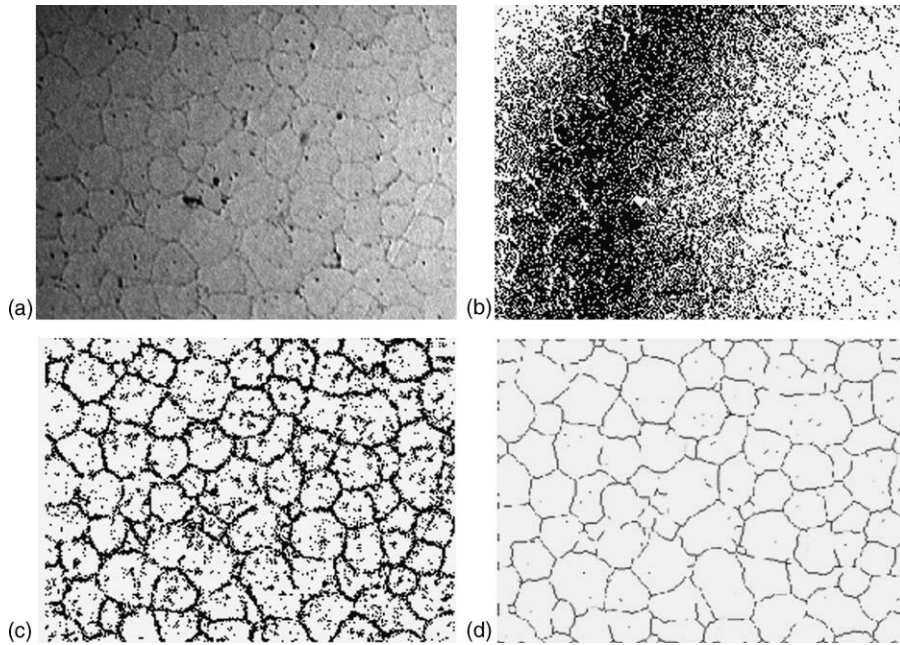


Fig. 13. A smoothed and contrast enhanced source image (a), output of simple thresholding by hand (b), output of the NNGBD algorithm (c), and the result of post-processing after the NNGBD algorithm (d).

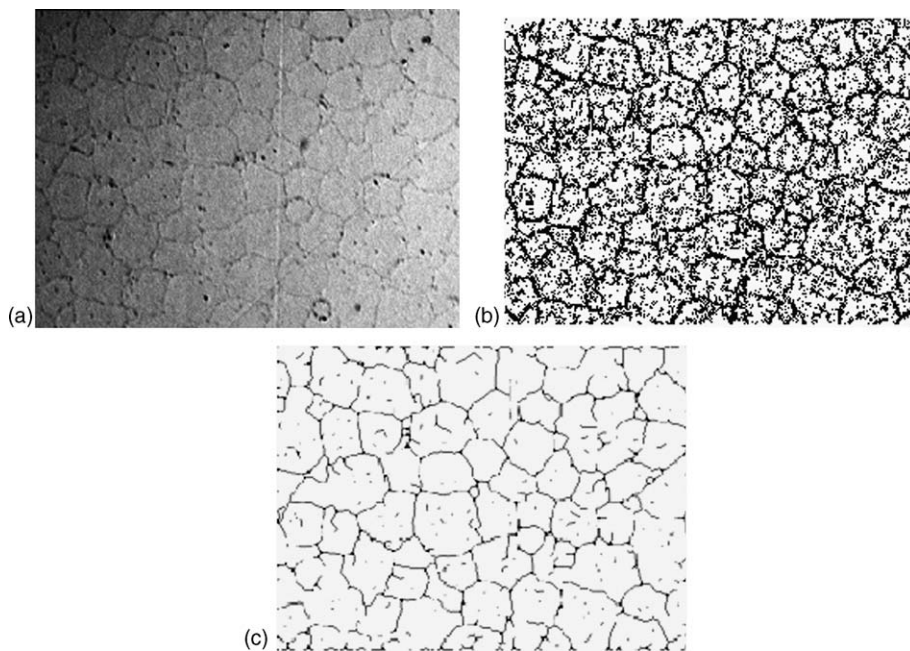


Fig. 14. A smoothed and contrast enhanced source image (a), outputs of the NNGBD algorithm (b), and post-processing after the NNGBD algorithm (c).

## 5. Conclusions

Today, pattern recognition with image processing is used in areas from medicine to quality control in manufacturing using machine vision. Reliable real-time filtering is essential for efficient and effective machine vision systems. The applications presented in this paper show that computational intelligence techniques can be used for effective image processing of steel microstructure data. Full automation may not be possible to achieve but these tools can help users as initial layers of perception and take over a sizeable portion of time-consuming tasks.

## Acknowledgement

This project was supported by US National Science Foundation grant DMI-9800430 and by considerable in kind support by an industrial partner.

## References

- [1] H.V. Atkinson, S. Davies, Fundamental aspects of hot isostatic pressing: an overview, *Metallurgical and Materials Transactions A: Physical Metallurgy and Materials Science* 31 (12) (2000) 2981–3000.
- [2] M.F. Augustejn, L.E. Clemens, A neural-network approach to the detection of texture boundaries, *Engineering Applications Artificial Intelligence* 9 (1) (1996) 75–81.
- [3] A. Bigand, T. Bouwmans, J.P. Dubus, Extraction of line segments from fuzzy images, *Pattern Recognition Letters* 22 (2001) 1405–1418.
- [4] R. Eberhart, P. Simpson, R. Dobbins, *Computational Intelligence Tools*, Academic Press Limited, London, UK, 1996.
- [5] J.J. Friel, E.B. Prestridge, F. Glazer, Grain boundary reconstruction for grain sizing, *MiCon90: advances in video technology for microstructural control*, in: F. George, Vander Voort (Eds.), ASTM STP 1094, American Society for Testing and Materials, Philadelphia, 1990.
- [6] L.K. Hansen, P. Solomon, Neural network ensembles, *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAM 1 (12) (1990) 903–1002.
- [7] R.M. Haralick, K. Shagmugam, I. Dinstein, Textural features for image classification, *IEEE transactions on systems, Man and Cybernetics* 3 (1973) 610–621.
- [8] G. Heygster, Rank filters in digital image processing, *Computer Graphics and Image Processing* 19 (2) (1982) 148–164.
- [9] J.S. Kim, H.S. Cho, A fuzzy logic and neural network approach to boundary detection for noisy imagery, *Fuzzy Sets and Systems* 65 (1994) 141–159.
- [10] C.G. Looney, *Pattern Recognition Using Neural Networks, Theory and Algorithms for Engineers and Scientists*, Oxford University Press, New York City, NY, 1997.
- [11] MPIF, <http://www.mpiif.org>, 2003.
- [12] J.C. Russ, *The Image Processing Handbook*, third ed., CRC Press & IEEE Press, Boca Raton, Florida (FL), 1998.
- [13] S. Russell, P. Norvig, *Artificial Intelligence A Modern Approach*, Prentice Hall, Upper Saddle River, NJ 07458, 2003.
- [14] S.K. Sinha, F. Karray, Classification of underground pipe scanned images using feature extraction and neuro-fuzzy algorithm, *IEEE Transactions on Neural Networks* 13 (2) (2002) 393–401.
- [15] L.A. Zadeh, Fuzzy sets, *Information and Control* 8 (1965) 338–353.
- [16] A. Zaknich, Characterization of aluminum hydroxide particles from the Bayer process using neural network and Bayesian classifiers, *IEEE Transactions on Neural Networks* 8 (4) (1997) 919–931.



**Alice E. Smith** is Professor and Chair of the Industrial and Systems Engineering Department at Auburn University. Previous to this position, she was on the faculty of the Department of Industrial Engineering at the University of Pittsburgh, which she joined in 1991 after ten years of industrial experience with Southwestern Bell Corporation. Dr. Smith has degrees in engineering and business from Rice University, Saint Louis University and University of Missouri - Rolla. Her research in analysis, modeling and optimization of manufacturing processes and engineering design has been funded by NASA, the National Institute of Standards (NIST), Lockheed Martin, Adtranz (now Bombardier Transportation), the Ben Franklin Technology Center of Western Pennsylvania and the National Science Foundation (NSF), from which she was awarded a CAREER grant in 1995 and an ADVANCE Leadership grant in 2001. Her industrial partners on sponsored research projects have included DaimlerChrysler Electronics, Eljer Plumbingware, Extrude Hone, Ford Motor, PPG Industries and Crucible Compaction Metals. International research collaborations have been sponsored by the federal governments of Japan, Turkey, United Kingdom and the US. Dr. Smith holds one US patent and several international patents and has authored over 50 publications in books and journals including articles in *IIE Transactions*, *IEEE Transactions on Reliability*, *INFORMS Journal on Computing*, *International Journal of Production Research*, *IEEE Transactions on Systems, Man, and Cybernetics*, *Journal of Manufacturing Systems*, *The Engineering Economist*, and *IEEE Transactions on Evolutionary Computation*. Dr. Smith holds editorial positions on *INFORMS Journal on Computing*, *Computers & Operations Research*, *International Journal of General Systems*, *IEEE Transactions on Evolutionary Computation* and *IIE Transactions*. Dr. Smith is a fellow of IIE, a senior member of IEEE and SWE, a member of Tau Beta Pi, INFORMS and ASEE, and a Registered Professional Engineer in Industrial Engineering in Alabama and Pennsylvania.



**Ian Nettleship** is an Associate Professor of Materials Science and Engineering at the University of Pittsburgh, USA. He completed both his BSc in Materials Science and Engineering (1983) and his PhD in Ceramics Science and Engineering (1987) at the University of Leeds in the UK. He then worked as a postdoctoral researcher in the Materials Science and Engineering Departments at the University of California at Santa Barbara (1988 to 1989) and the University of Illinois at Urbana-Champaign (1989 to 1992). He became an Assistant Professor in Materials Science and Engineering at the University of Pittsburgh in 1992 and was promoted to his present position in 1998. Dr. Nettleship's research interests include quantitative analysis of microstructure evolution in materials and its application to ceramics processing and the properties of porous ceramics.



**Orhan Dengiz** was born in Ankara, Turkey. He received his BS degree in 2000 from Civil Engineering at Middle East Technical University, Ankara, Turkey, and received an MISE degree in 2002 from Industrial and Systems Engineering at Auburn University, Auburn, AL, USA. Dengiz is currently a PhD student at Industrial and Systems Engineering, Auburn University. He is a student member of INFORMS, IEEE, and IIE. Dengiz's current research interests involve using computational intelligence or heuristic optimization methods in automated image processing and analysis, finite element mesh generation, and in reliability and performance optimization of autonomous telecommunication networks.