JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 35, 555-575 (2019)
DOI: 10.6688/JISE.201905_35(3).0005

# Automatic Text Recognition in Natural Scene Using Neural Network Classifier with Dynamic-group-based Hybrid Particle Swarm Optimization*

KUANG-HUI TANG[1,2], CHUAN-KUEI HUANG[1] AND CHENG-JIAN LIN[3,+]
[1]*Department of Industrial Education and Technology*
*National Changhua University of Education*
*Changhua, 500 Taiwan*
[2]*Department of Electronic Engineering*
[3]*Department of Computer Science and Information Engineering*
*National Chin-Yi University of Technology*
*Taichung, 411 Taiwan*
*E-mail: tkhf14@ncut.edu.tw; ckhuang@cc.ncue.edu.tw; cjlin@ncut.edu.tw*

This paper presents a two-stage algorithm for automatic text detection and recognition. In the first stage, using a stroke width transform and an improved connected component, an edge analysis method detects a candidate character region. Subsequently, a text region is located by filtering and linking characters with similar font sizes and colors. For the second stage, a histogram of oriented gradient is employed as a feature descriptor, and a neural network classifier is built with dynamic-group-based hybrid particle swarm optimization (DGHPSO) for character recognition. In DGHPSO, each group's threshold value of similarity depends on the threshold values of fitness and distance. In addition, a local search algorithm is used to improve the search for a global optimum. The proposed algorithm was experimentally validated; it outperformed a number of recently published studies in terms of the text recognition rate when tested on the ICDAR 2003 database and the Street View Text database.

***Keywords:*** text detection, text recognition, neural network classifier, particle swarm optimization, natural scene

## 1. INTRODUCTION

In recent years, natural scene text detection has drawn increasing research attention, and has been successfully applied to a wide variety of disciplines, such as visual impairment assistance [1], multimedia retrieval [2], and industrial automation [3]. It is difficult to detect and recognize characters against complicated backgrounds, and a number of algorithms have been proposed to address this problem. Text detection approaches can be generally categorized into two types, namely, texture-based [4, 5] and region-based detection methods [6, 7]. With the first type, textual units or even symbols can be detected because their textures are considerably different from that of their background. Images are scanned for textures using multiscale windows. Several texture-based detection methods have been reported, such as discrete cosine transform, Fourier transform, wavelet coefficients, spatial variance, and Gabor filters. Li *et al.* [4] reported on image classi-

Received January 2, 2018; revised January 23, 2018; accepted March 1, 2018.
Communicated by Shyi-Ming Chen.
* This work was supported in part by the Ministry of Science and Technology, Taiwan, under Grant MOST 105-2221-E-167-028.
+ Corresponding author.

fication that used a neural network as well as the mean, second-, and third-order central moments of wavelet coefficients. By contrast, Lee *et al.* [5] combined features from gradients, Gabor filters, variances of Wavelet coefficients, and edge intervals, and then classified images by means of AdaBoost. In the latter case, pixels with similar colors and intensities are grouped together with connected components, and nontext components are filtered. Jain and Yu [6] employed color reduction to generate color layers as the syntactic rules of connected components and then connected candidate character regions with similar colors. As presented by Koo and Kim [7], each component was split into eight square subregions, and the following features were chosen: (1) the number of foreground pixels; (2) the number of vertical white-black transitions; and (3) the number of horizontal black-white transitions. Characters and noncharacters were classified according to the features of all the subregions. Although region-based detection methods can effectively locate candidate character regions, a major disadvantage is that empirical rules are required.

Feature extraction is a very important process within text recognition. If an inappropriate feature extraction method is used, the result of the text recognition can be seriously affected. To overcome the problem, several researchers [8-14] have considered different regions of interest with the aim of extracting features at different resolutions. Dalal and Triggs [8] proposed the Histogram of Oriented Gradients (HOG). This technique counts occurrences of gradient orientation in localized rectangular areas of an image. Ahonen *et al.* [9] presented local binary patterns (LBP) to encode each pixel in terms of the magnitude relation with the surrounding neighbors. In contrast to LBP, Jun and Kim [10] proposed local gradient patterns (LGP), which consider the pixel gradient instead of its gray value. Unlike LBP, LGP representation is insensitive to global intensity variations. Local ternary patterns (LTP), proposed by Tan and Triggs [11], are an extension of LBP. Unlike LBP, LTP does not divide the pixels into 0 and 1 values but into three values. Instead of using a histogram with a large number of bins, the ternary pattern is commonly split into two binary codes known as LTP low and LTP high. Chen *et al.* [12] proposed the Weber local descriptor (WLD) for image texture classification. In particular, WLD comprises two components: differential excitation and orientation. The differential excitation component is a function of the ratio between the relative intensity differences of a pixel and its neighbors and the pixel intensity. The orientation component is the pixel gradient orientation. Ojansivu and Heikkil [13] proposed the Local Phase Quantization (LPQ) for quantizing the phase information of the local Fourier transform. In [14], Castrillon-Santana *et al.* analyzed the aforementioned feature extraction methods and obtained the similar the results of the text recognition. Because the HOG technique is simple and easy to implement, we adopt HOG technique as the feature extraction method in this study.

Text recognition techniques can be mainly classified into two types, optical character recognition (OCR)-based and object-based recognition techniques. Traditional OCR methods are devoted to binary image processing, by which characters can be separated from the background, and the extracted characters can be recognized using an OCR engine. Gllavata *et al.* [15] first employed a color quantizer to determine the colors of text samples and background samples, and then pixels were classified as either text or background using a modified K-means algorithm. A poor recognition rate is caused by image noise or complicated backgrounds because the recognition performance of color-based clustering methods [8] is susceptible to color consistency. Chen *et al.* [16] built gray-

scale images using a Gaussian mixture model (GMM), and assigned pixels to a Gaussian layer based on a Markov random field. Ye *et al.* [17] used pixel intensities and hue components in the HSI color space as training pixels to train a GMM, and thus extracted text from image backgrounds. Recognition techniques, involving binarization or preprocessing, provide a higher character recognition rate than conventional binarization approaches such as that proposed by Otsu [18] and Niblack [19]. However, those binarization approaches underperformed due to their constraints on illumination, character font size, and style. By contrast, object-based recognition methods are conducted under the assumption that scene characters are recognized in a manner similar to object recognition; that is to say, there exists a high degree of intra-class variation for both scene character recognition and object recognition. For characters in a single font, linear discriminant analysis [20] is frequently employed for character classification; however, typical practical applications have multiple character fonts. Therefore, this problem must be modeled in such a way that characters in multiple fonts can be recognized. For instance, Sheshadri *et al.* [21] employed a support vector machine to recognize characters. Yao *et al.* [22] proposed a Bag-of-Strokelets character feature extraction method and used a random forest as a character classifier.

Recently, several researchers [23-30] have adopted the granular computing-based approach for solving cluster and classification problems. A multilayer perceptron utilizing a supervised learning technique called backpropagation (BP) can be used for training the network [25]. However, the error curve may converge very slowly or the solution may fall into a local minimum when it has been trained by BP. In recent years, several researchers have adopted evolutionary computation to avoid falling into local minima. Observations on animals' foraging behaviors have resulted in a number of algorithms, such as Genetic algorithms [31-36], Ant Colony Optimization [37], Differential Evolution algorithms [38], Bacterial Foraging Optimization [39], and Particle Swarm Optimization (PSO) [40-43]. PSO is inspired by the foraging behavior of birds, and has been applied to a wide variety of disciplines over recent years. Representing an individual in a flock of birds, a particle carries, and is able to reference the information provided by others to determine the direction of its next move in an optimization process. Traditional PSO has the advantages of few parameters, fast speed of convergence, easy implementation, and global search ability, and has been applied to various fields successfully. Recently, academic PSO research has mainly focused on improving its parameters through theoretical analysis, mathematical inference, and empirical research. For example, Lovbjerg *et al.* [44] introduced the concepts of subpopulations and reproduction into PSO to achieve faster convergence. Higashi *et al.* [45] used Gaussian mutation to redesign the rules of particle position and speed. Baskar *et al.* [46] designed two particle swarms, which worked in parallel and exchanged information to overcome premature convergence. For the inertia weight in PSO, Chatterjee *et al.* [47] proposed an inertial weight with nonlinear variation, and Nickabadi *et al.* [48] proposed an adaptive inertial weight. Moreover, Bansal *et al.* [49] conducted experiments for comparing different methods for inertia weight calculation. The results revealed that the chaotic inertial weight had the optimal effect, and the random inertial weight had the highest efficiency. When PSO systems are applied to highly complex engineering problems, those systems exhibit several disadvantages, such as poor accuracy and a tendency to become trapped in local optima. Therefore, a novel learning algorithm is presented herein to overcome the afore-

mentioned problems.

In this paper, a two-stage algorithm is proposed for automatic text detection and recognition. An edge-based text detection method is proposed for automatic text detection, while a neural network classifier based on dynamic-group-based hybrid particle swarm optimization (DGHPSO) is proposed for automatic text recognition. An edge analysis method is presented to improve the text detection rate. Candidate character regions are detected using a stroke width transform and an improved connected component, and text regions are located by means of filtering and linking candidate character regions. With reference to text recognition, a neural network classifier is developed on the basis of DGHPSO for character classification. In DGHPSO, based on the average of the distance difference and the average of the fitness difference between these ungrouped particles and group leader, the threshold value of similarity comprises the threshold values of fitness and distance. Therefore, in this study the distance and fitness threshold values of every group are determined automatically. Finally, with ICDAR 2003 and Street View Text (SVT) as the test datasets, the work is validated experimentally to provide a higher text recognition rate than the counterparts.

The rest of this paper is outlined as follows. Section 2 presents an edge-based text detection technique, Section 3 presents a DGHPSO-based neural network classifier as a prerequisite of text recognition, Section 4 presents the experimental results and a performance comparison, and Section 5 presents the conclusions.

## 2. EDGE-BASED TEXT DETECTION

This section presents an edge analysis approach as a prerequisite for text detection. In an image, a significant contrast exists between characters, symbols, and background, according to which a text region can be identified in an efficient and accurate manner. The edge-based text extraction approach is illustrated as a flowchart in Fig. 1.

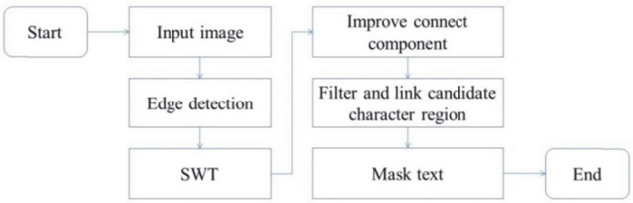


Fig. 1. Brief flowchart of the proposed text detection approach.

### 2.1 Stroke Width Transform

As explicitly stated in [50], a character in an image is composed of several strokes, each having a pair of parallel edges. First, a scene image is converted into a gray-scale image, and an edge is detected using the Canny algorithm [51]. The gradient, including the direction and the magnitude, is evaluated at each edge pixel. Pixel tracking is performed until another edge pixel is hit, at which juncture the algorithm considers whether the gradient points roughly in the opposite direction. For example, an edge pixel has a gradient pointing to 30°, while a hit pixel has a gradient between 180° and 240°. Subse-

quently, each pixel over the tracking path is assigned the length of the path as the pixel value. Fig. 2 is an illustration of the stroke width transform.

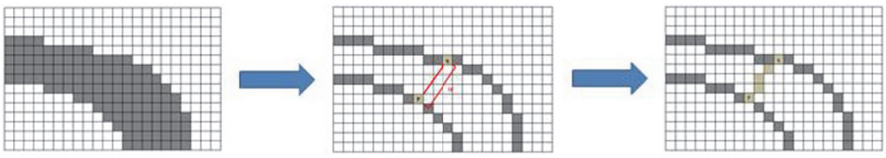

Fig. 2. Illustration of the stroke width transform.

### 2.2 Improved Connected Component

Following the stroke width transform, the system updates a subset of pixel values in an image, according to which the pixels are separated into a large number of potential character regions using an improved connected component. In most cases, a connected component is employed to separate a binary image into character regions; each character region is assigned a flag; all neighboring regions are assigned the same flag. Conventionally, the two most common types of connectedness are 4-connectivity (Fig. 3) and 8-connectivity, which require neighboring pixels to determine whether the same pixel value is shared. Herein, binary images are not considered; instead, the system considers whether the ratio between neighboring pixel values is between 1/3 and 3.

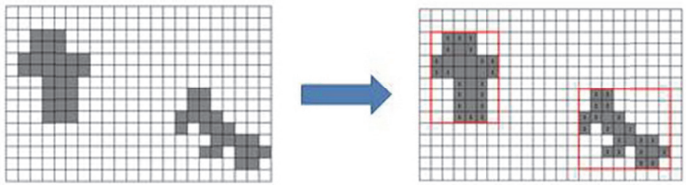

Fig. 3. Illustration of 4-connectivity.

### 2.3 Candidate Character Region Filtering and Link

Two rules are involved to filter character regions out of connected regions. A character region is known to have pixels with similar pixel values. The first, and the most important, rule is designed to identify and remove most noncharacter regions according to the standard deviation of pixel values over each region. The second rule is to filter extremely thin or wide regions because a region longer than 300 pixels wide or shorter than 10 pixels is considered to be a noncharacter region. Consequently, the remaining regions are treated as the candidate character regions, which are then linked as follows. First, the first candidate character region is paired with another in a horizontal direction on the condition that both regions have similar font sizes and colors. Subsequently, the distances between the centers of paired character regions and adjacent regions are evaluated, and then the region with the greatest distance is divided into different groups.

## 3. TEXT RECOGNITION USING A NEURAL NETWORK CLASSIFIER

Text is recognized using a neural network classifier with a weight optimized by a

dynamic-group-based hybrid particle swarm optimization algorithm. Text recognition is illustrated as a brief flowchart in Fig. 4.

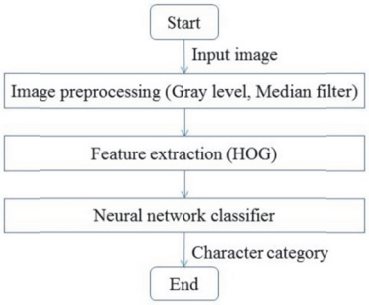


Fig. 4. Text recognition flowchart using a neural network classifier.

### 3.1 Image Preprocessing and Character Feature Extraction

Each character image has distinct features, such as symmetry and size. Hence, image preprocessing, including filtering and size standardization, must be performed prior to feature extraction. In feature extraction, a character image is first converted into a gray-scale image, and then a 3×3 median filter is employed to filter the unwanted noise. Subsequently, the character is resized to a square that measures 100×100 pixels. A HOG is adopted for feature extraction due to multiple advantages. An HOG has geometric and photometric features because it operates on a local, *i.e.* 100×50 pixel, mask in an image. First, an image is split into small-sized connected regions, referred to as cells. The gradient direction at each single pixel contained in a cell is collected, accumulated using bilinear interpolation, and then illustrated as a histogram. The range of 0~360° is divided into a number of bins, say 18 bins; that is, each bin covers a 20° sector, and an 18-bin HOG is built as a feature descriptor. As illustrated in Fig. 5, a character image is split into two regions of size 100×50 pixels, each with 18 orientation bins. That is, there are a total of 36 features.

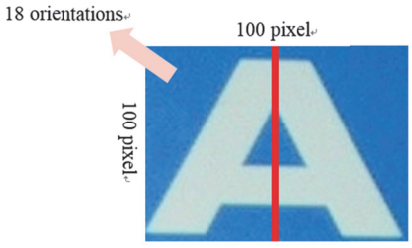


Fig. 5. Illustration of a character image split.

### 3.2 Neural Network Classifier

Recently, neural network and fuzzy theory methods [62, 66, 67] are usually used as a classifier. Because a neural network is simple and easy implementation, we adopt it as

a classifier in this paper. Features of a character are considered as inputs to a neural network classifier, as illustrated in Fig. 6, involving a great number of neurons. A single neuron sums the weighted input signals, namely the output signals of the previous layer, which is formulated as $y_{NO} = f(x) = \sum_{i=1}^{NI} w_{ij} x_i$. The weights are initially specified as a random number between +1 and −1, and the aim is to tune the weights of the neural network classifier.

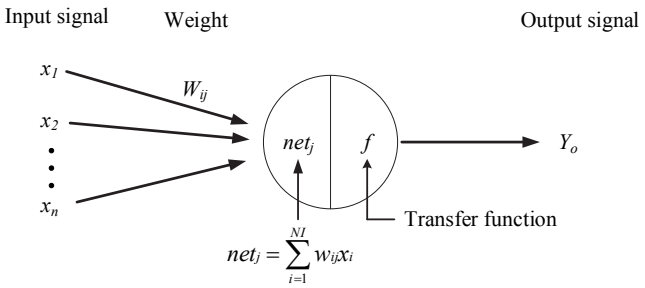


Fig. 6. Model of a single neuron.

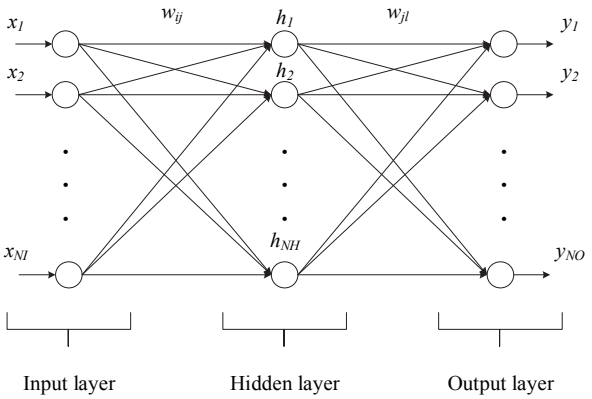


Fig. 7. Framework of a neural network.

As illustrated in Fig. 7, a multilayer neural network involves an input layer, a hidden layer, and an output layer.

(1) Input Layer:

Input and output signals are related by

$$o_i^{(I)}(k) = x_i(k) \tag{1}$$

(2) Hidden Layer:

The input signal at node $k$ is expressed as

$$net_j^{(H)}(k) = \sum_{i=1}^{NI} o_i^{(I)}(k) \cdot w_{ij} \tag{2}$$

where $w_{ij}$, $i = 1, 2, \ldots, NI$, $j = 1, 2, \ldots, NH$ represent the weights between the input and hidden layers. A sigmoid transfer function transfers the input to the output of the hidden layer. Hence, the output at node $k$ is expressed as

$$o_j^{(H)}(k) = 1/(1 + \exp(-net_j^{(H)}(k))) \tag{3}$$

(3) Output Layer:

Operation in the output layer is given as

$$net_l^{(O)}(k) = \sum_j^{NH} o_i^{(H)}(k) \cdot \boldsymbol{w}_{jl} \tag{4}$$

where $w_{jl}$, $l = 1, 2, \ldots, NO$, represent the weights between the hidden and the output layers. The output is expressed as

$$y_l = o_l^{(O)}(k) = 1/(1 + \exp(-net_l^{(O)}(k))). \tag{5}$$

The BP algorithm has a high possibility of being trapped in a local minimum. Therefore, in this paper a novel hybrid evolutionary algorithm is presented herein to adjust the weights of a neural network classifier for text recognition, which is detailed as follows.

### 3.3 Dynamic-group-based Hybrid Particle Swarm Optimization

In the section, the DGHPSO algorithm is presented. To have a superior reference position, a particle is dynamically grouped because all the particles of a swarm continue moving during an evolutionary process. In general, the similarity measure method in [52] is usually used. In this study, we consider the fitness and distance threshold values. The DGHPSO algorithm is detailed as follows:

**Step 1:** Particle ranking

As illustrated in Fig. 8 (a), ungrouped particles are sorted according to fitness. Performance superiority is reflected by a greater fitness.

**Step 2:** Similar threshold value

As illustrated in Fig. 8 (b), a particle is grouped according to the similar threshold value, evaluated prior to particle grouping, between two particles. The similar threshold value consists of the fitness and the distance threshold values. The former is defined as

$$Threshold_{Fit} = \frac{\sum_{i=1}^{N-1} \left| Fit_{Leader} - Fit_{P_i} \right|}{N-1} \tag{6}$$

$$FIT_i = |Fit_{Leader} - Fit_{Pi}| \tag{7}$$

where $N$ represents the number of ungrouped particles, $Fit_{Leader}$ represents the ungrouped particle with the greatest fitness, $FIT_i$ represents the difference between the fitness of particle $i$ and the greatest fitness, and $P_i$ represents the $i$th ungrouped particle. The distance between $P_i$ and the particle with the greatest fitness $L_{Leader}$ is expressed as

$$D_i = \sqrt{(\boldsymbol{L}_{\boldsymbol{Leader}} - \boldsymbol{P}_i)^2} \tag{8}$$

and particles are grouped according to entropy. The distance threshold value is defined as

$$Threshold_{EM} = \frac{-\sum_{i=1}^{N-1} \boldsymbol{D}_i \log_2 \boldsymbol{D}_i}{\boldsymbol{N}-1}. \tag{9}$$

**Step 3:** Particle grouping

As illustrated in Fig. 8 (c), an ungrouped particle $i$ is assigned to a group on the condition that $Fit_{Pi}$ and $D_i$ are lower than $Threshold_{Fit}$ and $Threshold_{EM}$, respectively; that is,

If ($Threshold_{Fit} > Fit_{P_i}$) and ($Threshold_{EM} > D_i$), then join the group.

**Step 4:** Determine whether to terminate particle grouping.

Determine whether there is any ungrouped particle. If yes, skip to Step 1 for a new run or terminate, as illustrated in Fig. 8 (d).

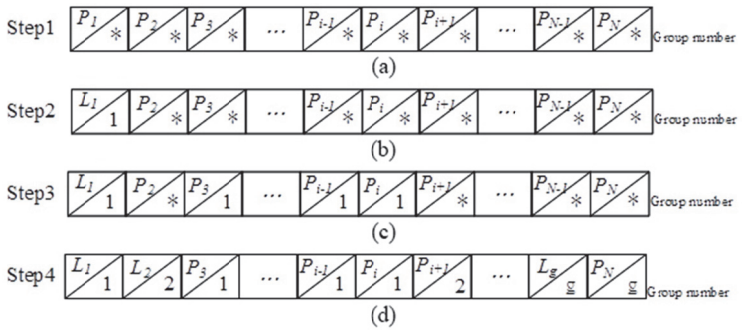


Fig. 8. Illustrations for (a) particle ranking; (b) similar threshold value; (c) particle grouping; and (d) decision on grouping termination.

In this paper, a particle refers to the optimal position of its group leader $L_g$, and the velocity is updated as

$$v_i(\boldsymbol{n}+1) = \omega \times v_i(\boldsymbol{n}) + \boldsymbol{c}_1 \times \boldsymbol{rand}_1 \times (\boldsymbol{P}_{\boldsymbol{Lg}} - \boldsymbol{P}_i(\boldsymbol{n})) + \boldsymbol{c}_2 \times \boldsymbol{rand}_1 \times (\boldsymbol{P}_{\boldsymbol{Gbest}} - \boldsymbol{P}_i(\boldsymbol{n})). \tag{10}$$

A typical PSO algorithm is frequently criticized for its tendency of getting trapped in a local optimum. Therefore, a local search (LS) mechanism is presented herein to overcome the aforementioned problem. This is achieved by using the position information and passing/sharing between two particles. More precisely, a particle with a low fitness is guided toward the right direction by another with a high fitness, or a particle can be rescued out of a trap with the help of a distant particle. In LS, $P_i$ refers to the $i$th particle, while $P_j$ is a randomly selected particle, and a mutualistic relationship is built to improve respective positions. The particle positions can be updated as

$$P_i(\boldsymbol{n}+1) = \boldsymbol{P}_i(\boldsymbol{n}) + \boldsymbol{rand}(-1, 1) \times (\boldsymbol{P}_{\boldsymbol{best}} - \boldsymbol{Mutual\_vector} \times \boldsymbol{BF}_1) \tag{11}$$

$$P_j(n+1) = \boldsymbol{P_j(n)}+\boldsymbol{rand}(-1, 1)\times(\boldsymbol{P_{best}} - \boldsymbol{Mutual\_vector}\times\boldsymbol{BF_2}) \tag{12}$$

$$\boldsymbol{Mutual\_vector} = \frac{\boldsymbol{P_i} + \boldsymbol{P_j}}{2} \tag{13}$$

where $rand(-1, 1)$ represents a random number between −1 and 1, $P_{best}$ is the particle with the optimal position, and $BF_1$ and $BF_2$ are the benefit factors specified as either 1 or 2. In a mutualistic relationship, a particle may not benefit from the other as much as the other way around. For example, particle $i$ may receive a large benefit from particle $j$, while particle $j$ may merely receive an adequate or insignificant benefit from its counterpart. In other words, $BF_1$ and $BF_2$ reflect the amount of the received benefit. The relationship between two particles is characterized by Eq. (13), while the quantity ($P_{best} - Mutual\_vector\times BF_1$) in Eqs. (11)-(12) is concerned with the improvement in particle positions. The LS mechanism is illustrated as a flowchart in Fig. 9, and the particle positions are finally updated only if an improved fitness is observed.

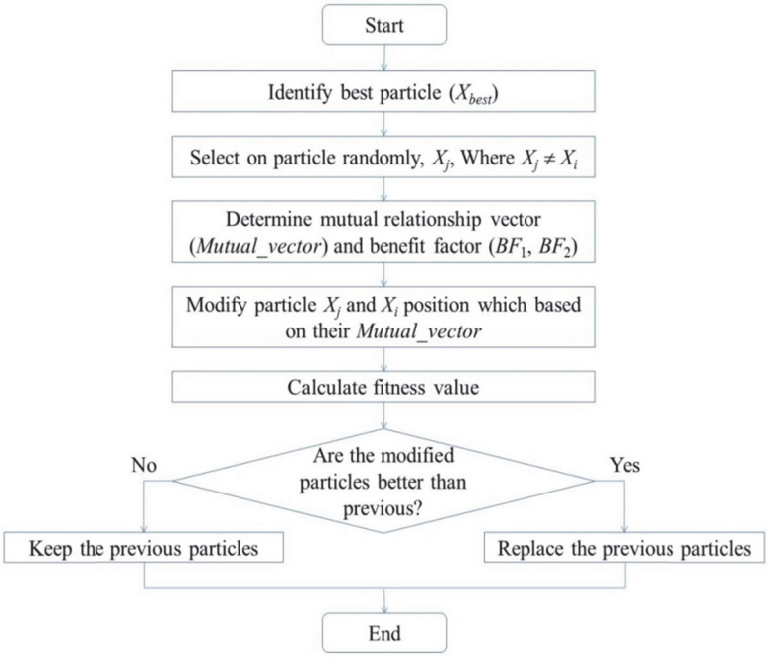


Fig. 9. Flowchart of the presented local search algorithm.

As illustrated in Fig. 10, first initialize all the particles, evaluate $Threshold_{EM}$ and $Threshold_{FM}$, and then group all the particles accordingly in this presented PSO algorithm. The position and velocity of a particle are updated according to the global optimal position and the group leader's optimal position. The presented LS algorithm is then used to identify better particle positions, update the personal and global best positions, and determine whether to terminate an optimum search.

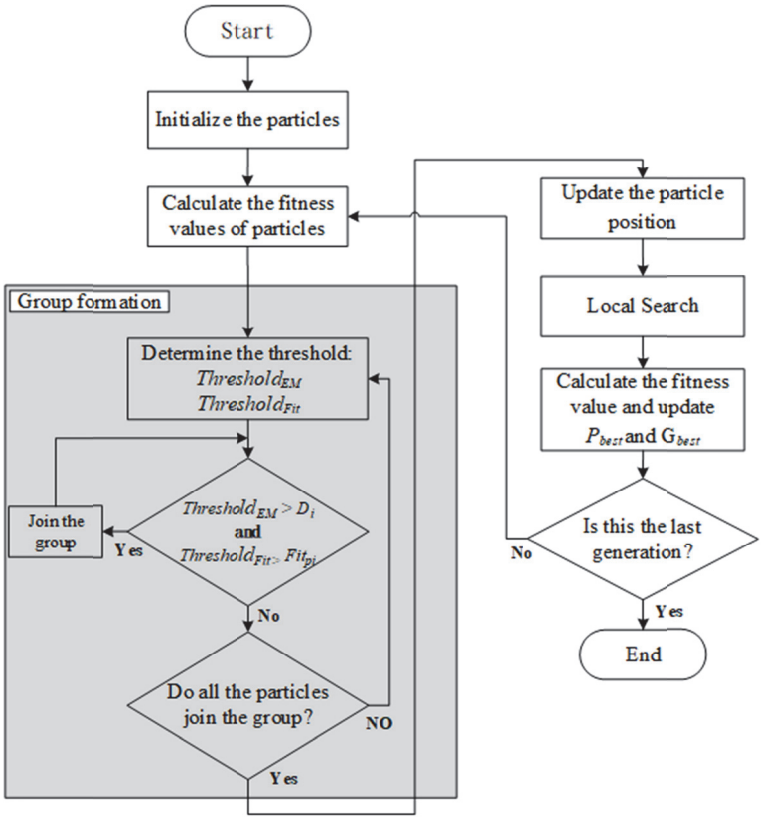


Fig. 10. Flowchart of the proposed DGHPSO algorithm.

## 4. EXPERIMENTAL RESULTS AND DISCUSSION

The Chars74K, ICDAR 2003, and SVT databases were used to test the performance of this work. More precisely, Chars74K was used as the training data for the neural network classifier, while the others were used as the test data.

### 4.1 Text Detection Results

As explicitly stated previously, text detection is performed on the ICDAR 2003 and SVT database using the presented edge analysis method. The ICDAR 2003 database comprises 249 unequal-sized images with different colors, fonts, and complicated backgrounds. The precision, recall, and *f*-score are the common measures of text detection performance, defined as

$$\textbf{\textit{Precision}} = \frac{\sum_{r_e \in E} \boldsymbol{m}(r_e, \boldsymbol{T})}{|\boldsymbol{E}|} \tag{14}$$

$$\textbf{\textit{Recall}} = \frac{\sum_{r_t \in T} \boldsymbol{m}(r_t, \boldsymbol{E})}{|\boldsymbol{T}|} \tag{15}$$

$$f = \frac{1.0}{\frac{a}{Precision} + \frac{1.0 - a}{Recall}} \tag{16}$$

where $T$ and $E$ represent all the ground-truth elements and estimated rectangles, respectively; $r_t$ and $r_e$ represent a ground-truth and a detected rectangle, respectively; $f$ symbolizes the harmonic mean of the precision and recall, and the parameter $a$ is frequently specified as 0.5. As presented in Fig. 11, this work can successfully detect text out of a complicated background. Table 1 compares the performance of this method with other works [53-56]. The proposed method outperforms the others in all aspects, except for the superior recall and $f$-score provided by the last counterpart.


Fig. 11. Text detection results using ICDAR 2003 as a test database.

**Table 1. Performance comparison among various approaches using the test database ICDAR 2003.**

| Quantitative indicators / Method | Accuracy | Recall | $f$-score |
|---|---|---|---|
| Proposed method | 0.64 | 0.58 | 0.61 |
| Qiang Zhu [53] | 0.40 | 0.33 | 0.36 |
| Kim *et al.* [54] | 0.28 | 0.22 | 0.25 |
| Ezaki *et al.* [55] | 0.36 | 0.18 | 0.27 |
| Yu *et al.* [56] | 0.62 | 0.79 | 0.69 |

**Table 2. Performance comparison among various approaches using the test database SVT.**

| Quantitative indicators / Method | Accuracy | Recall | $f$-score |
|---|---|---|---|
| Proposed method | 0.52 | 0.48 | 0.50 |
| Wang *et al.* [57] | 0.40 | 0.36 | 0.38 |
| Nermann [58] | 0.33 | 0.19 | 0.26 |
| Yu *et al.* [56] | 0.52 | 0.40 | 0.45 |

SVT, in contrast to ICDAR 2003, comprises color Google street views with complicated backgrounds. As shown in Fig. 12, texts are successfully detected again using the presented edge analysis method. Table 2 compares the precision, recall, and $f$-score; the proposed method outperforms the counterparts in all respects without any exceptions.


Fig. 12. Texts are successfully detected by the proposed method.

### 4.2 Text Recognition Results

As mentioned previously, Chars74K is used as the training data for the neural network classifier, while ICDAR 2003 and SVT are used as the test data for performance validation. The classifier is equipped with a total of 36 inputs, $x_1 - x_{36}$, and a total of 43 outputs, $y_1 - y_{43}$, represented as $Y = [y_1, y_2, y_3, \ldots, y_{42}, y_{43}]$ for brevity. Accordingly, the characters of interest, namely the target outputs, are defined as

character A: target output d = [0 0 0 L 0 0 0 L 0 0 1]
character B: target output d = [0 0 0 … 0 0 0 … 0 1 0]
character C: target output d = [0 0 0 … 0 0 0 … 1 0 0] ❑

character a: target output d = [0 0 0 … 0 0 1 … 0 0 0]
character b: target output d = [0 0 0 … 0 1 0 … 0 0 0]
character d: target output d = [0 0 0 … 1 0 0 … 0 0 0] ❑

character u: target output d = [0 1 0 … 0 0 0 … 0 0 0]
character y: target output d = [1 0 0 … 0 0 0 … 0 0 0] ❑

Table 3 shows the parameter settings for particle initialization. The numbers of input and output nodes of the proposed neural network represent the numbers of features and classifications, respectively. The parameters of PSO are adopted based on the experiments of most researchers [40, 41, 59]. In addition, evolution is performed 10 times for testing the stability, and the fitness value is defined as

$$Fitness\_value = \frac{1}{1 + MSE} \tag{17}$$

where *MSE* represents the mean square error. Optimizing the number of particles is difficult. Fig. 13 shows the learning curves representing the performance by considering the number of particles, that is, 150, 250, and 300, as a parameter. Table 4 compares the fitness versus number of particles in DGHPSO.

**Table 3. Parameter settings for an initialization process.**

| Parameters | Value |
|---|---|
| The number of input nodes | 36 |
| The number of hidden nodes | 10 |
| The number of output nodes | 43 |
| The amount of training data | 430 |
| The number of generations | 100000 |
| The number of particles | 150, 250, 300 |
| $w$(max) | 0.6 |
| $w$(min) | 0.2 |
| $c_1$, $c_2$ | 2.0 |
| $BF_1$, $BF_2$ | 1 |

**Table 4. Fitness versus the number of particles in DGHPSO.**

| Fitness \ Number of Particle | 150 | 250 | 300 |
|---|---|---|---|
| Best | 0.7099 | 0.7301 | 0.7918 |
| Worst | 0.6186 | 0.6909 | 0.7226 |
| Average | 0.6618 | 0.6740 | 0.7497 |

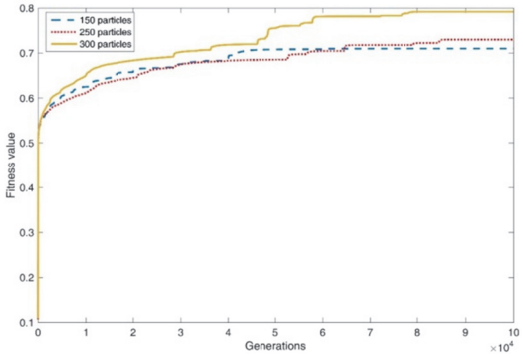


Fig. 13. Learning curves with the number of particles as a parameter in DGHPSO.

**Table 5. Fitness comparison among various algorithms.**

| Algorithm / Fitness | Proposed method | PSO [40] | QPSO [59] | ABC [60] | MABC [61] | IABC [62] | DE [38] | Rank-DE [63] |
|---|---|---|---|---|---|---|---|---|
| Best | 0.791 | 0.704 | 0.708 | 0.685 | 0.743 | 0.712 | 0.663 | 0.673 |
| Worst | 0.740 | 0.638 | 0.620 | 0.648 | 0.701 | 0.650 | 0.614 | 0.616 |
| Average | 0.750 | 0.672 | 0.652 | 0.658 | 0.730 | 0.688 | 0.633 | 0.632 |
| STD | 0.011 | 0.021 | 0.019 | 0.009 | 0.012 | 0.014 | 0.009 | 0.015 |

As shown in Table 5, an inferior performance is observed in the case of 150 particles, although it requires a short processing time as well as a small amount of memory. By contrast, the case of 300 particles outperforms but requires considerable processing time. Therefore, a population of 300 particles is used in an initialization process at all events for performance comparison. Fig. 14 compares the fitness versus number of

generations, and Table 6 compares the best, worst, average, and STD fitness values. DGHPSO outperforms its counterparts in all respects, except for the STD in ABC [60] and DE [38].

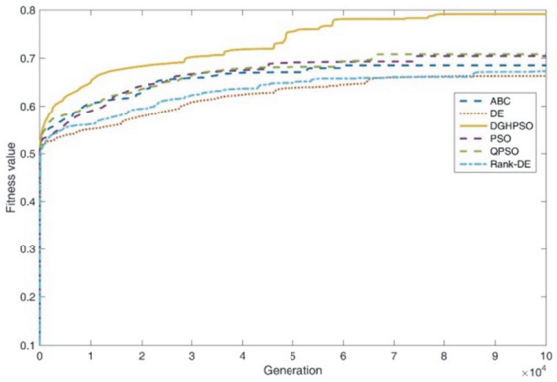


Fig. 14. Learning curve comparison among various algorithm.

Using the proposed algorithm, colored texts with different fonts can be successfully recognized (shown right below the respective photos) from a wide variety of complicated backgrounds, as shown in Figs. 15-16. Table 6 compares the various algorithms in term of accuracy by using the IDCAR 2003 and SVT test datasets. A combined use of DGHPSO and LS provides the highest accuracy, while using only DGHPSO offers the second-highest accuracy as expected. That is, LS is validated as an effective way to substantially improve the recognition accuracy. As can be seen in Table 7, this work outperforms several existing works in terms of recognition accuracy.

**Table 6. Accuracy comparison among various algorithms.**

| Algorithms \ Database | IDCAR 2003 database (Accuracy) | SVT database (Accuracy) |
|---|---|---|
| Proposed method (with LS) | 73.7% | 67.1% |
| Proposed method (without LS) | 66.1% | 57.2% |
| PSO[40] | 58.1% | 49.0% |
| QPSO [59] | 58.8% | 42.7% |
| ABC[60] | 55.3% | 46.5% |
| MABC[61] | 64.2% | 56.8% |
| IABC[62] | 60.2% | 51.7% |
| DE[38] | 49.5% | 41.3% |
| Rank-DE[63] | 53.0% | 45.1% |

**Table 7. Accuracy comparison among various algorithms.**

| Method \ Database | IDCAR 2003 database (Accuracy) | SVT database (Accuracy) |
|---|---|---|
| Proposed method | 73.7% | 67.1% |
| Wang *et al.* [57] | 62% | 57% |
| Feild *et al.* [64] | 73.43% | 54.2% |
| ABBYY9.0[65] | 55% | 35% |


Fig. 15. Text detection results using the ICDAR 2003 database.


Fig. 16. Text detection results using the SVT database.

## 5. CONCLUSIONS

In this paper, a two-stage algorithm for automatic text detection and recognition is proposed. An edge analysis method is presented to improve the text detection rate. A candidate character region is obtained by a stroke width transform and improved connection components, and a text region is acquired by filtering and linking candidate character regions. Characters are classified herein using a neural network classifier with the proposed DGHPSO learning algorithm. In DGHPSO, the threshold value of similarity consists of the threshold values of fitness and distance. Therefore, the distance and the fitness threshold values of every group are determined automatically. In addition, a local search algorithm is validated as an effective way to considerably improve the search accuracy for a global optimum. The superior performance of this work is demonstrated using the presented edge analysis method with ICDAR 2003 and SVT as test databases, and it is concluded that this work provides a high recognition accuracy from unconstrained scene images.

Recently, a convolutional neural network (CNN) [57] was used to detect and recognize characters, resulting in a high recognition accuracy. However, a traditional CNN utilizes the BP learning technique for training its network, and such systems are easily trapped in local minima. Therefore, an evolutionary computation method for tuning the CNN parameters will be proposed in future works.

## REFERENCES


1. J. Chen, "A method to extract character strings from scene images by eliminating non-character strings: Towards development of a visual assistance system for people with visual impairments," in *Proceedings of International Conference on Signal-Image Technology and Internet-Based Systems*, 2013, pp. 848-853.
2. Q. Ye, Q. Huang, W. Gao, and D. Zhao, "Fast and robust text detection in images and video frames," *Image and Vision Computing*, Vol. 23, 2005, pp. 565-576.
3. Z. He, J. Liu, H. Ma, and P. Li, "A new automatic extraction method of container identity codes," *Transactions on Intelligent Transportation Systems*, Vol. 6, 2005, pp. 72-78.
4. H. Li, D. Doermann, and O. Kia, "Automatic text detection and tracking in digital video," *IEEE Transactions on Image Processing*, Vol. 9, 2000, pp. 147-156.
5. J. Lee, P. Lee, S. Lee, A. Yuille, and C. Koch, "AdaBoost for text detection in natural scene," in *Proceedings of International Conference on Document Analysis and Recognition*, 2011, pp. 429-434.
6. A. K. Jain and B. Yu, "Automatic text location in images and video frames," *Pattern Recognition*, Vol. 31, 1998, pp. 2055-2076.
7. H. Koo and D. H. Kim, "Scene text detection via connected component clustering and non-text filtering," *IEEE Transactions on Image Processing*, Vol. 22, 2013, pp. 2296-2305.
8. N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of International Conference on Computer Vision and Pattern Recognition*, Vol. 2, 2005, pp. 886-893.
9. T. Ahonen, A. Hadid, and M. Pietikäinen, "Face description with local binary patterns: application to face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 28, 2006, pp. 2037-2041.
10. B. Jun and D. Kim, "Robust face detection using local gradient patterns and evidence accumulation," *Pattern Recognition*, Vol. 45, 2012, pp. 3304-3316.
11. X. Tan and B. Triggs, "Enhanced local texture feature sets for face recognition under difficult lighting conditions," *IEEE Transactions on Image Processing*, Vol. 19, 2010, pp. 1635-1650.
12. J. Chen, S. Shan, C. He, G. Zhao, M. Pietikainen, X. Chen, and W. Gao, "WLD: a robust local image descriptor," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 32, 2010, pp. 1705-1720.
13. V. Ojansivu and J. Heikkila, "Blur insensitive texture classification using local phase quantization," in A. Elmoataz, O. Lezoray, F. Nouboud, D. Mammass (eds.), *Image and Signal Processing*, LNCS 5099, Springer, 2008, pp. 236-243.
14. M. Castrillon-Santana, J. Lorenzo-Navarro, C. M. Travieso-Gonzalez, D. Freire-Ob-

regon, J. B. Alonso-Hernandez, "Evaluation of local descriptors and CNNs for non-adult detection in visual content," *Pattern Recognition Letters*, http://dx.doi.org/10.1016/j.patrec.2017.03.016, 2017.
15. J. Gllavata, R. Ewerth, T. Stefi, and B. Freisleben, "Unsupervised text segmentation using color and wavelet features," *Image and Video Retrieval*, Springer-Verlag, NY, 2004, pp. 216-224.
16. D. Chen, J. Olobez, and H. Bourlard, "Text segmentation and recognition in complex background based on Markov random field," in *Proceedings of the 16th International Conference on Pattern Recognition*, Vol. 4, 2002, pp. 227-230.
17. Q. Ye, W. Gao, and Q. Huang, "Automatic text segmentation from complex background," in *Proceedings of International Conference on Image Processing*, Vol. 5, 2004, pp. 2905-2908.
18. N. Otsu, "A threshold selection method from gray-level histograms," *Automatica*, Vol. 11, 1975, pp. 285-296.
19. W. Niblack, *An Introduction to Digital Image Processing*, Strandberg, IL, 1985.
20. X. Chen, J. Yang, J. Zhang, and A. Waibel, "Automatic detection and recognition of signs from natural scenes," *IEEE Transactions on Image Processing*, Vol. 13, 2004, pp. 87-99.
21. K. Sheshadri and S. K. Divvala, "Exemplar driven character recognition in the wild," in *Proceedings of British Machine Vision Conference*, 2012, pp. 1-10.
22. C. Yao, X. Bai, B. Shi, and W. Liu, "Strokelets: A learned multiscale representation for scene text recognition," in *Proceedings of IEEE International Conference Computer Visual and Pattern Recognition*, 2014, pp. 4042-4049.
23. G. Peters and R. Weber, "DCC: A framework for dynamic granular clustering," *Granular Computing*, Vol. 1, 2016, pp. 1-11.
24. P. Lingras, F. Haider, and M. Triff, "Granular meta-clustering based on hierarchical, network, and temporal connections," *Granular Computing*, Vol. 1, 2016, pp. 71-92.
25. M. Song and Y. Wang, "A study of granular computing in the agenda of growth of artificial neural networks," *Granular Computing*, Vol. 1, 2016, pp. 247-257.
26. M. Antonelli, P. Ducange, B. Lazzerini, and F. Marcelloni, "Multi-objective evolutionary multiplicative aggregation in group decision making design of granular rule-based classifiers," *Granular Computing*, Vol. 1, 2016, pp. 37-58.
27. H. Liu and M. Cocea, "Granular computing-based approach for classification towards reduction of bias in ensemble learning," *Granular Computing*, Vol. 2, 2017, pp. 131-139.
28. H. Liu and M. Cocea, "Fuzzy Information granulation towards interpretable sentiment analysis," *Granular Computing*, Vol. 2, 2017, pp. 289-302.
29. H. Liu and M. Cocea, "Semi-random partitioning of data into training and test sets in granular computing context," *Granular Computing*, Vol. 2, 2017, pp. 357-386.
30. F. Min and J. Xu, "Semi-greedy heuristics for feature selection with test cost constraints," *Granular Computing*, Vol. 1, 2016, pp. 199-211.
31. D. B. Fogel, "Evolutionary optimization," in *Proceedings of IEEE International Conference on Signals*, *Systems and Computers*, Vol. 1, 1992, pp. 409-414.
32. S. M. Chen and C. Y. Chien, "Parallelized genetic colony systems for solving the traveling salesman problem," *Expert Systems with Applications*, Vol. 38, 2011, pp. 3873-3883.

33. S. M. Chen and Y. C. Chang, "Weighted fuzzy rule interpolation based on GA-based weight-learning techniques," *IEEE Transactions on Fuzzy Systems*, Vol. 19, 2011, pp. 729-744.
34. S. M. Chen, Y. C. Chang, and J. S. Pan, "Fuzzy rules interpolation for sparse fuzzy rule-based systems based on interval type-2 Gaussian fuzzy sets and genetic algorithms," *IEEE Transactions on Fuzzy Systems*, Vol. 21, 2013, pp. 412-425.
35. S. M. Chen and C. M. Huang, "Generating weighted fuzzy rules from relational database systems for estimating null values using genetic algorithms," *IEEE Transactions on Fuzzy Systems*, Vol. 11, 2003, pp. 495-506.
36. S. M. Chen and N. Y. Chung, "Forecasting enrollments of students using fuzzy time series and genetic algorithms," *International Journal of Information and Management Sciences*, Vol. 17, 2006, pp. 1-17.
37. M. Dorigo and G. D. Caro, "Ant colony optimization: A new meta-heuristic," in *Proceedings of Congress on Evolutionary Computation*, Vol. 2, 1999, pp. 1470-1477.
38. R. Storn and K. Price, "Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, Vol. 11, 1997, pp. 341-359.
39. K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," *IEEE Control Systems Magazine*, Vol. 22, 2002, pp. 52-67.
40. J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of IEEE International Conference on Neural Networks*, Vol. 4, 1995, pp. 1942-1948.
41. S. M. Chen and P. Y. Kao, "TAIEX forecasting based on fuzzy time series, particle swarm optimization techniques and support vector machines," *Information Sciences*, Vol. 247, 2013, pp. 62-71.
42. P. W. Tsai, J. S. Pan, S. M. Chen, B. Y. Liao, and S. P. Hao, "Parallel cat swarm optimization," in *Proceedings of International Conference on Machine Learning and Cybernetics*, Vol. 6, 2008, pp. 3328-3333.
43. S. M. Chen and T. H. Chang, "Finding multiple possible critical paths using fuzzy PERT," *IEEE Transactions on Systems*, *Man*, *and Cybernetics – Part B: Cybernetics*, Vol. 31, 2001, pp. 930-937.
44. M. Lovbjerg, T. K. Rasmussen, and T. Krink, "Hybrid particle swarm optimizer with breeding and subpopulations," in *Proceedings of the 3rd Genetic and Evolutionary Computation Conference*, 2001, pp. 469-476.
45. N. Higashi and H. Iba, "Particle swarm optimization with Gaussian mutation," in *Proceedings of Congress on Evolutionary Computation*, 2003, pp. 72-79,
46. S. Baskar and P. N. Suganthan, "A novel concurrent particle swarm optimization," in *Proceedings of Congress Evolutionary Computation*, 2004, pp. 792-796.
47. A. Chatterjee and P. Siarry, "Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization," *Computer & Operation Research*, Vol. 33, 2006, pp. 859-871.
48. N. Ahmad, M. Mehdi Ebadzadeh, and R. Safabakhsh, "A novel particle swarm optimization algorithm with adaptive inertia weight," *Applied Soft Computing*, Vol. 11, 2011, pp. 3658-3670.
49. J. C. Bansal, P. K. Singh, M. Saraswat, A. Verma, S. S. Jadon, and A. Abraham, "Inertia weight strategies in particle swarm optimization," in *Proceedings of the 3rd World Congress on Nature and Biologically Inspired Computing*, 2011, pp. 633-640.

50. B. Epshtein, E. Ofek, and Y. Wexler, "Detecting text in natural scenes with stroke width transform," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2963-2970.
51. J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1986, pp. 679-698.
52. S. M. Chen and J. H. Chen, "Fuzzy risk analysis based on similarity measures between interval-valued fuzzy numbers and interval-valued fuzzy number arithmetic operators," *Expert Systems with Applications*, Vol. 36, 2009, pp. 6309-6317.
53. S. Lucas, "ICDAR 2005 text locating competition results," in *Proceedings of International Conference on Document Analysis and Recognition*, 2005, pp. 80-84.
54. J. Kim, S. Park, and S. Kim, "Text locating from natural scene images using image intensities," in *Proceedings of the 8th International Conference on Document Analysis and Recognition*, 2005, Vol. 2, pp. 655-659.
55. N. Ezaki, K. Kiyota, B. T. Minh, M. Bulacu, and L. Schomaker, "Improved text-detection methods for a camera-based text reading system for blind persons," in *Proceedings of IEEE International Conference on Document Analysis and Recognition*, 2005, pp. 257-261.
56. C. Yu, Y. H. Song, Q. Meng, Y. L. Zhang, and Y. Liu, "Text detection and recognition in natural scene with edge analysis," *IET Computer Vision*, Vol. 9, 2015, pp. 603-613.
57. T. Wang, D. J. Wu, A. Coates, and A. Y. Ng, "End-to-end text recognition with convolutional neural networks," in *Proceedings of International Conference on Pattern Recognition*, 2012, pp. 3304-3308.
58. L. Neumann and J. Matas, "Real-time scene text localization and recognition," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3538-3545.
59. S. Jun, W. Xu, and B. Feng, "A global search strategy of quantum-behaved particle swarm optimization," in *Proceedings of IEEE Conference on Cybernetics and Intelligent Systems*, Vol. 1, 2004, pp. 111-116.
60. D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, Vol. 39, 2007, pp. 459-471.
61. C. J. Lin and M. L. Huang, "Modified artificial bee colony algorithm for scheduling optimization for printed circuit board production," *Journal of Manufacturing Systems*, Vol. 4, 2017, pp. 1-11.
62. Y. R. Chen, S. C Kuo, and C. J. Lin, "Mobile robot navigation control using recurrent fuzzy CMAC based on improved dynamic artificial bee colony," in *Proceedings of International Conference on Applied System Innovation*, 2016, pp. 496.
63. W. Gong and Z. Cai, "Differential evolution with ranking-based mutation operators," *IEEE Transactions on Cybernetics*, Vol. 43, 2013, pp. 2066-2081.
64. J. Feild and E. G. Learned-Miller, "Scene text recognition with bilateral regression," Technical Report UM-CS-2012-021, Department of Computer Science, University of Massachusetts Amherst, 2012.
65. ABBYY FineReader 9.0, http://www.abbyy.com, 2014.
66. Y. C. Chang, S. M. Chen, and C. J. Liau, "Multilabel text categorization based on a new linear classifier learning method and a category-sensitive refinement method,"

*Expert Systems with Applications*, Vol. 34, 2008, pp. 1948-1953.
67. S. M. Chen, S. H. Lee, and C. H. Lee, "A new method for generating fuzzy rules from numerical data for handling classification problems," *Applied Artificial Intelligence*, Vol. 15, 2001, pp. 645-664.

**Kuang-Hui Tang (唐光輝)** received the M.S. degree in Telecommunication Engineering from the Da-Yeh University, Taiwan, in 2005. Currently, he is an Assistant Professor of National Chin-Yi University of Technology, Taiwan. His current research interests are image processing and recognition, communication electronics, electromagnetic compatibility and solar control systems.

**Chuan-Kuei Huang (黃川桂)** received the Ph.D. degree in graduate school of Engineering Science and Technology from the National Yunlin University of Science and Technology, Taiwan, in 2004. Currently, he is a Professor of Department of Industrial Education and Technology, National Changhua University of Education, Taiwan. His current research interests are image processing, Chaotic system application, power electronics, and electric vehicle control.

**Cheng-Jian Lin (林正堅)** received the Ph.D. degree in Electrical and Control Engineering from the National Chiao-Tung University, Taiwan, in 1996. Currently, he is a Distinguished Professor of Computer Science and Information Engineering Department, National Chin-Yi University of Technology, Taichung County, Taiwan. His current research interests are machine learning, intelligent control, image processing and recognition, deep leaning, and intelligent manufacture.