S & M 3059

Floor Area Detection by Multiple-classifier Method Based on Improved Fuzzy Integral

Chi-Huang Shih,[1] Cheng-Jian Lin,[1,2*] and Jyun-Yu Jhang[3]

[1]Department of Computer Science & Information Engineering,
National Chin-Yi University of Technology, Taichung 411, Taiwan
[2]College of Intelligence, National Taichung University of Science and Technology, Taichung 404, Taiwan
[3]Department of Computer Science & Information Engineering,
National Taichung University of Science and Technology, Taichung 404, Taiwan

(Received April 3, 2022; accepted September 2, 2022)

Keywords: image segmentation, fuzzy integral (FI), artificial bee colony (ABC), fully convolutional network (FCN), area contour detection (ACD).

The areas of application of mobile robots are becoming increasingly extensive. Most mobile robots use visual images to obtain information about the surrounding environment, including information about the floor area that can be traversed or obstacles to be avoided. In this study, we propose a multiple-classifier method based on an improved fuzzy integral (IFI) for floor area detection using charge-coupled device sensors. The multiple classifiers include deep learning networks (DLNs) and an area contour detection (ACD) algorithm. The IFI in the proposed method is different from the traditional fuzzy integral in that its fuzzy density value is optimized using the artificial bee colony algorithm. The proposed method takes the outputs of several classifiers and the ACD algorithm as the input. The IFI is used to evaluate each classifier to leverage its advantages for the proposed method. Finally, the classification results are calculated to achieve accurate floor area detection. To verify the performance of the proposed method, we used the public MIT scene dataset and images of various indoor environments as the study dataset, with 200, 48, and 48 images used as training data, verification data, and test data, respectively. The experimental results revealed that the DLN–ACD–IFI combination achieved an average floor area detection accuracy of 97.8%, thus improving the overall recognition rate.

1. Introduction

In recent years, mobile robots have been widely used in different fields. In some mobile robot applications, visual images from charge-coupled device (CCD) sensors have been used to obtain information about the surrounding environment, thus enabling the mobile robot to avoid obstacles and move autonomously. Accordingly, image segmentation is a key technology that enables a mobile robot system to automatically segment object areas from an image and identify objects.

Numerous traditional image-sensing-based segmentation techniques have been proposed. For example, Chun *et al.*[(1)] proposed a novel method for calculating depth information from a single

*Corresponding author: e-mail: cjlin@ncut.edu.tw
https://doi.org/10.18494/SAM4043

indoor image through nonlinear diffusion and image segmentation. In this method, nonlinear diffusion is used to detect floor candidates, and image segmentation is then used to detect the floor area and calculate the depth information. However, this method cannot be applied to complex environments. Kumar *et al.*(2) used superpixels to segment some smaller obstacles. Because obstacles with a similar appearance to the floor area cannot be accurately distinguished, they developed a line segment detection method for detecting obstacles on the floor. However, this method cannot accurately segment the floor area. Aggarwal *et al.*(3) proposed a framework for estimating floor regions in cluttered indoor environments and used a generic classifier that was trained on appearance cues and floor density estimates using various indoor images. Moreover, Boykov and Kolmogorov(4) experimentally compared the efficiency of mincut/maxflow algorithms for applications in image restoration and segmentation.

Although the aforementioned traditional image segmentation methods can detect the floor area, they can perform only rough segmentation and cannot accurately segment objects. Therefore, many researchers have applied deep learning to image segmentation. A convolutional neural network (CNN) can effectively implement image classification. However, because a CNN has a fully connected layer, the original 2D matrix becomes 1D, resulting in the loss of spatial information.

To obtain a segmentation map as the output, Long *et al.*(5) proposed a fully convolutional network (FCN) for generalizing an end-to-end convolutional network for image segmentation. Badrinarayanan *et al.*(6) proposed SegNet, which transfers the maximum pooling index to the decoder to improve the recognition rate. The Google team(7) proposed DeepLabv2, which uses atrous convolution to increase the receptive field without increasing the number of parameters and uses fully connected conditional random fields (CRFs) to optimize the prediction results. However, atrous convolution has a high computational cost and requires substantial memory. Lin *et al.*(8) proposed RefineNet and the corresponding encoder–decoder structure.

Optimization methods have been proposed to improve the performance of deep learning networks (DLNs). The most common optimization method is minibatch stochastic gradient descent (SGD), which randomly selects a certain number of training samples for training each time. This method usually learns efficiently; however, it relies excessively on the learning rate setting and requires a long training time. Therefore, some researchers have proposed different optimization methods(9–13) to address these drawbacks; of these methods, Adam(14) is the most widely used.

In this study, we propose a multiple-classifier method based on an improved fuzzy integral (IFI) for floor area detection using CCD sensors. The multiple classifiers included for combination in the proposed method are DLNs and an area contour detection (ACD) algorithm. The proposed method uses multiple trained DLNs and ACD as inputs to the fuzzy integral (FI) and computes classification results using Sugeno or Choquet output rules. Moreover, the multiple trained DLNs and ACD are combined with the IFI, and the corresponding fuzzy density value is optimized using the artificial bee colony (ABC) algorithm, thus overcoming the drawback of majority decision rules in traditional voting methods. The IFI evaluates each classifier to leverage its advantages for the proposed method. Finally, for the input data, the classification results are derived to achieve a more accurate floor area detection process.

The remainder of this paper is organized as follows. In Sect. 2, we describe a previously proposed FCN and the DeepLabv2 network. In Sect. 3, the proposed multiple-classifier method based on the IFI is presented. In Sect. 4, we present the experimental results obtained using the proposed method. Finally, in Sect. 5, conclusions and future work are given.

2. Image Segmentation Methods

In this section, we present three image segmentation methods, namely, a previously proposed FCN, DeepLabv2, and the ACD algorithm, that are used in the proposed method. These three methods are detailed as follows.

2.1 FCN algorithm

In this subsection, we introduce the FCN algorithm proposed by Long *et al.*(5) for performing pixel-level image classification and end-to-end pixel-to-pixel image segmentation. This FCN is trained through a supervised learning approach. An FCN differs from a general CNN in that an FCN is built on a “full convolution” network. An FCN can take an input image of any size from a CCD sensor and use a deconvolution layer to upsample the feature map of the last convolutional layer. Subsequently, the network output is restored to the same size as the input image. Each pixel in the image is predicted. Finally, pixel classification is performed on the upsampled feature map.

This FCN mainly comprises convolutional, pooling, upsampling, activation function, and skip layers. The operating process of each layer is described below.

2.1.1 Convolutional layers

Convolutional layers comprise several convolution kernels and are used to extract input image features. Multiple convolutional layers can extract more complex features. Each convolutional layer uses a convolution kernel mask of different weight combinations to perform a convolution operation (i.e., an inner product operation) with a sliding window and generate a feature map. Figure 1 displays a schematic of the convolution operation

$$Y_{IJ} = \sum_{i=0}^{c_1}\sum_{j=0}^{c_1} X_{ij} * K_{ij}, \tag{1}$$

where Y_{IJ} is the output matrix, c_1 represents the width and height of the convolution kernel, X_{ij} represents the input matrix, and K_{ij} represents the weight in the convolution kernel. The classifier must learn K_{ij} during the training process.

2.1.2 Pooling layer

The pooling layer reduces the dimension and the subsequent large number of parameter operations without losing crucial feature information. The pooling process involves the use of a

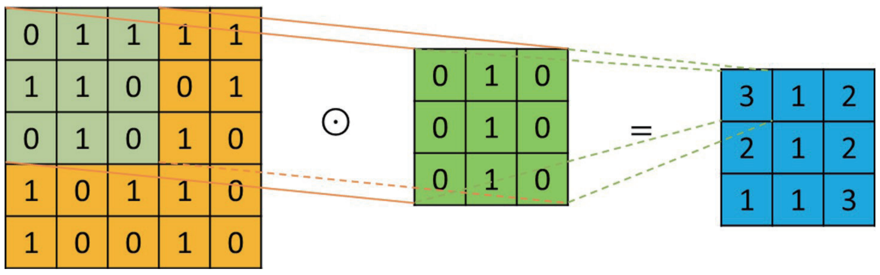

Fig. 1. (Color online) Schematic of convolution operation.

sliding window to perform masking operations on the input image. However, a nonoverlapping method is adopted in the masking process, which means that each pixel of the input matrix only undergoes one pooling operation. The most commonly used nonlinear pooling functions are max pooling and average pooling (Fig. 2). The proposed method uses max pooling, and the maximum value in the mask is used as the output; that is, the most prominent features are retained.

$$Y_j = \max_{x_2 * y_2}\left(X_i^{p_1 * p_1}\right) \tag{2}$$

Here, Y_j is the output, X_i is the input, and p_1 is the size of the pooling mask. However, because the masks do not overlap with each other, the output dimension of the pooling layer is $1/(p_1 \times p_1)$ times the input dimension.

2.1.3 Upsampling layer

The upsampling layer is a fully connected layer that replaces the convolutional network. It is used to restore the pooled output to a segmentation map of the input image size. This process is called deconvolution or transposed convolution. Similar to convolution, deconvolution involves multiplication and addition operations; it is schematically illustrated in Fig. 3.

2.1.4 Activation function layer

The activation function layer enables a network with only a linear combination to employ nonlinear expressions for solving more complex nonlinear problems. Common activation functions include sigmoid, tanh, and rectified linear unit (*ReLU*). Previously, the sigmoid function was mostly used as the activation function; however, the gradient disappears in the process of backward transfer learning. Therefore, *ReLU* is currently used as the activation function to solve this problem and reduce the degree of overfitting. Figure 4 presents the ReLU activation function.

$$ReLU(x) = \max(0, x) \tag{3}$$

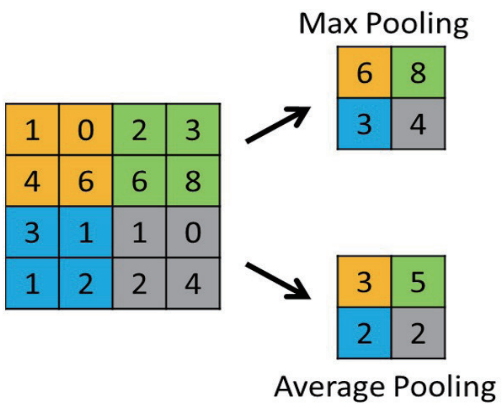

Fig. 2. (Color online) Schematic of pooling operation.

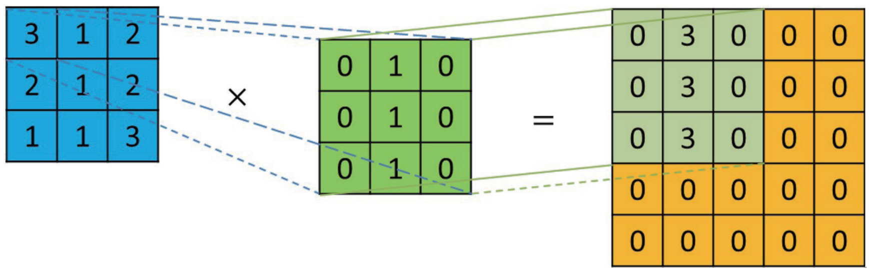

Fig. 3. (Color online) Schematic diagram of deconvolution.

2.1.5 Skip layer

The results obtained after full convolution can be upsampled to obtain dense prediction results. However, segmentation results are relatively rough. Therefore, the skip layer in the FCN proposed by Long *et al.*(5) is included, which has the primary function of optimizing the results. The results of different pooling layers are upsampled to optimize the output; that is, the sum of the outputs of the last few layers and the final output is calculated.

2.2 DeepLabv2 network

Here, DeepLabv2(7) proposed by the Google team in 2017 is introduced. It uses atrous convolution as a powerful tool for dense prediction tasks. Furthermore, it applies atrous spatial pyramid pooling (ASPP), in which multiscale information is used to obtain more robust segmentation results. Finally, the segmentation boundary results are improved by combining deep CNNs (DCNNs) and dense CRFs, which afford advantages including high speed, high accuracy, and easy implementation. Figure 5 shows a schematic of DeepLabv2.

DeepLabv2 involves the following steps.

Step 1: Use atrous convolution in the DCNN in the ASPP module to obtain rough prediction results for the input image.

Step 2: Enlarge the image to its original size through bilinear interpolation.

Step 3: Refine the prediction results through the fully connected CRF.

The components of DeepLabv2 are described as follows.

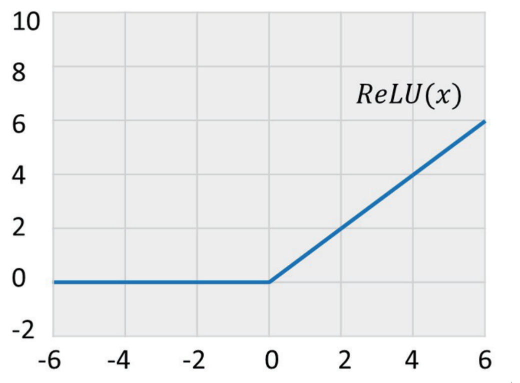

Fig. 4. (Color online) ReLU activation function.

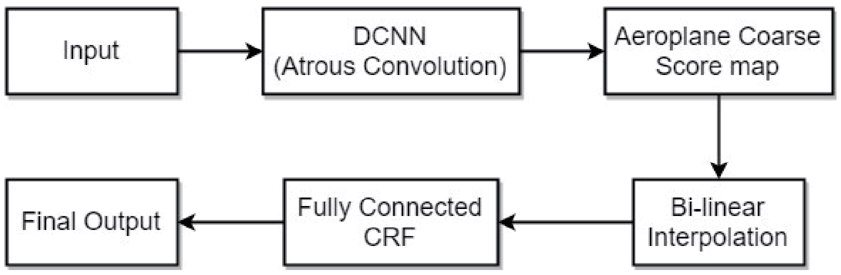

Fig. 5. Schematic of DeepLabv2.

2.2.1 Atrous convolution

Atrous convolution is used for dense feature extraction and for enlarging the receptive field without increasing the number of parameters or computations. The feature map of any layer can be calculated at any feature response resolution. Figure 6 shows a schematic of atrous convolution. This convolution process can be expressed as

$$y[i] = \sum_{k=1}^{K} x[i + r \cdot k]\omega[k], \tag{4}$$

where $y[i]$ is the hole convolution output, $x[i]$ represents the 1D input signal, r is the atrous rate (the stride of the sampled input signal), and $\omega[k]$ represents the filter of length k.

2.2.2 ASPP module

The ASPP module represents multiscale images that can be used to improve the DCNN accuracy in segmenting objects of different sizes. DeepLabv2 uses multiple sampling rates in parallel to extract features. The features are then combined in a configuration similar to a spatial pyramidal structure. Figure 7 shows a schematic of the ASPP module.

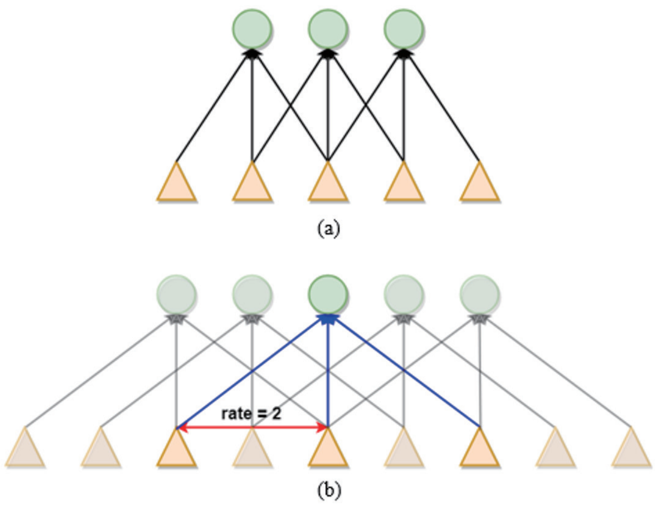

Fig. 6. (Color online) Schematic of atrous convolution: (a) sparse feature extraction and (b) dense feature extraction.

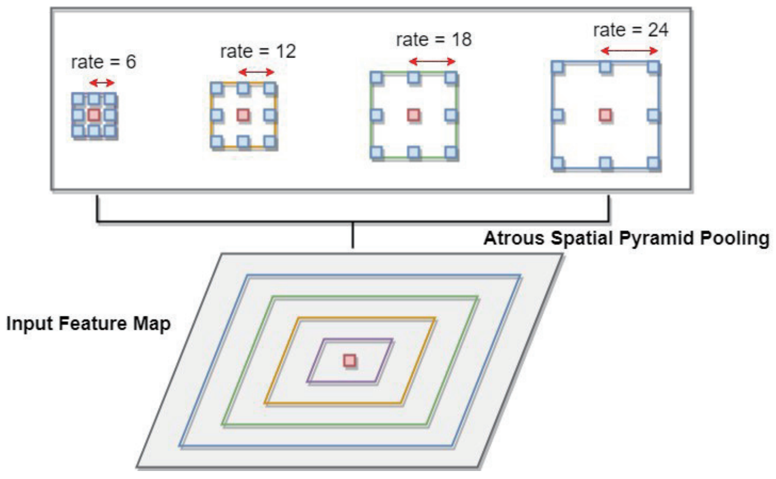

Fig. 7. (Color online) Schematic of ASPP module.

2.2.3 Fully connected CRF

Owing to the inaccuracy of classification and boundary cutting in the DCNN, a fully connected CRF[7] is used for structure prediction to improve the boundary accuracy. The fully connected CRF finds the relationship between image pixels and assigns labels by considering the probability of the pixels. After several iterations to refine the result, a more accurate boundary can be obtained.

2.3 ACD algorithm

Because of the weak representation ability of DLNs for floor details, the proposed method applies the ACD algorithm[15] to improve its representation of floor details. The ACD algorithm

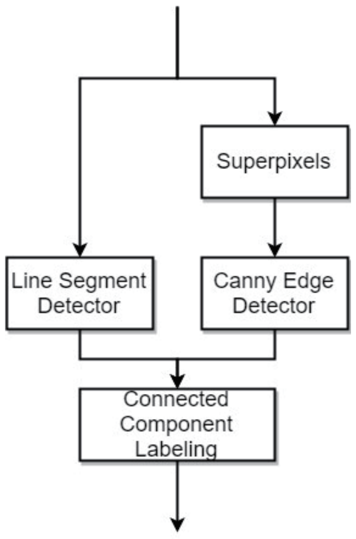

Fig. 8. Schematic of ACD algorithm.

identifies the rough areas of a floor through two methods: object edge detection in an image and line segment intersection (Fig. 8). Edge detection is executed to detect the floor and objects by extracting the edge. To reduce the number of calculations, the image is first processed using the simple linear iterative clustering (SLIC) algorithm—a superpixel segmentation algorithm—to group all adjacent and similar pixels. The pixels are then processed in blocks. Line segment intersection is executed, and the areas created after the intersection are calculated. Finally, the floor area is identified through object labeling.

2.3.1 SLIC algorithm

The SLIC algorithm uses local clustering. Specifically, it calculates the similarity between each pixel in an image and its adjacent pixels for clustering. Compared with processing pixels individually, processing grouped superpixels is conducive to improving operational speed and affords the advantages of easy implementation and high efficiency.

2.3.2 Canny edge detector

The Canny edge detector[16] both reduces noise and provides precise edge locations. After the input image is processed by the SLIC algorithm, the average value of the RGB components in the segmented superpixels is calculated and used as the input. Therefore, the detected edge is closer to the boundary of the object and removes some unnecessary noise.

2.3.3 Line segment detector

Gioi *et al.*[17,18] proposed the line segment detector (LSD) for extracting line segment features in an image. The LSD can quickly detect line segments in an image; subsequently, an error control method is used to enhance the detection accuracy. In the LSD, the input is a grayscale image, and the outputs are the coordinates and attributes of the detected line segment. Li and Birchfield(19) proposed an improved LSD that can operate on longer diagonal lines. In the improved LSD, lines are divided into three categories, after which the intersections of the diagonal lines with all other line segments are calculated. An adaptive threshold is set to identify the diagonal line with the most intersections and extend it. In this study, the adaptive threshold was set to 20% of the maximum number of intersections.

2.3.4 Connected-component labeling

After the edge and line segment features of the object are detected by the Canny edge detector and the LSD, connected-component labeling (CCL)(20) is used to label the object. Therefore, an appropriate area is selected as the rough floor area.

3. Proposed Method

Each DLN has advantages and disadvantages. To leverage the advantages of DLNs, in this study, we propose a multiple-classifier method based on the IFI for detecting floor areas. The multiple classifiers include DLNs and the ACD algorithm. The method involves three combination schemes: combining the same DLN architectures, combining different DLN architectures, and combining different DLN architectures and the ACD algorithm.

The traditional FI can combine multiple classifiers in the same task to improve performance.(21,22) Consider, for example, traditional feature extraction methods. In such methods, several features are extracted from the same image and then used as the input in trained classifiers; the derived output is used as the input of the FI. The FI evaluates the output of each classifier through a fuzzy measure, and the final output is then calculated.

In this study, to accurately detect the location of the floor area in an image, we propose a method in which the IFI is combined with several DLNs and the ACD algorithm. First, different DLNs, each with its own performance capability or advantage, are trained using the same set of training data. Subsequently, each DLN with good representation ability is combined with the IFI to obtain favorable accuracy. Figure 9 shows a flowchart of the proposed method.

Suppose $X = \{x_i\}_{i=1:n}$ represents a set of n classifiers. Accordingly, $g(x_i)$ is a fuzzy measure that represents the confidence level of each output of the classifier and can be used to evaluate the credibility of a subset. If the maximum value of the output is 1, then the output of this set can be completely trusted. If the minimum value of the output is 0, then the output of this set has no reference value. The FI is mainly based on a fuzzy measure, and the final output is determined by various rules. The fuzzy measure must meet the following three conditions.

Condition 1: $g(X) = 1$

When the outputs of all classifiers are consistent, the results must be trusted.

Condition 2: $g(\varnothing) = 0$

The results are meaningless when the outputs of all classifiers are not consistent.

Condition 3: The fuzzy measure must be an increasing monotonic function,

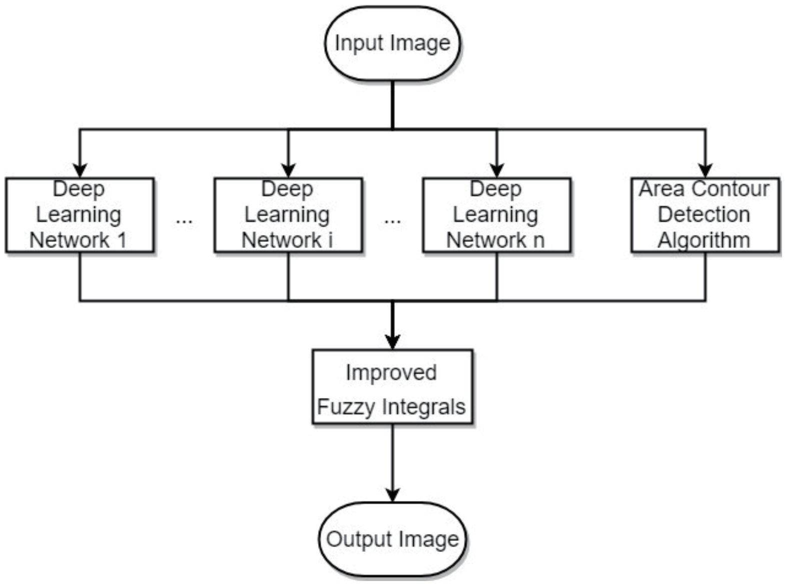

Fig. 9. Flowchart of proposed method.

$$\text{If } A \subset B \subset X, \text{ then } 0 \le g(A) \le g(B) \le 1. \tag{5}$$

Before calculating the fuzzy measure, the fuzzy density must be determined. When set g contains only one element, the fuzzy measure is called the fuzzy density; Fig. 10 displays the relationship between the two. However, in the traditional FI, the fuzzy density is determined by experts. Specifically, the fuzzy density is intuitively the confidence level of each output of the classifier. Therefore, some researchers used the accuracy of the classifier for nontraining data as the parameter for determining the fuzzy density.(23–25) Nevertheless, this determination approach is usually not optimal.

Accordingly, we apply the IFI to determine the fuzzy density through an optimization algorithm. The optimization algorithm is based on the ABC algorithm proposed by Karaboga and Basturk.(26) The ABC algorithm simulates the foraging behavior of bee colonies in nature and finds the best solution after several iterations of evolution. Figure 11 shows a flowchart of the ABC algorithm.

The fuzzy density parameter in the IFI is $n \times k$ floating-point numbers in the range [0, 1], where n is the number of classifiers and k is the number of classes in the classification problem. Therefore, the dimension of the solution space is $n \times k$ and the range is [0, 1]. Each bee in the ABC algorithm represents a set of fuzzy density solutions; Fig. 12 shows its code. The algorithm calculates the fitness value, which represents the quality of the solution. Specifically, the obtained fitness value represents the degree of excellence of the bee.

The overall evolution steps of the ABC algorithm are as follows.

Step 1: Initialization

The initial parameters are set, including the number of food sources (SN), maximum number of iterations, and number of continuous unimproved food sources ("limit"). The locations of the food source are generated by random numbers in the solution space:

$$x_i^j = x_{min}^j + rand\left[0,1\right]\left(x_{max}^j - x_{min}^j\right), \tag{6}$$

where x_i^j is the initial value of the *j*th dimension of the *i*th employed bee, x_{min}^j is the minimum value of the *j*th dimension of the solution space, and x_{max}^j is the maximum value of the *j*th dimension of the solution space.

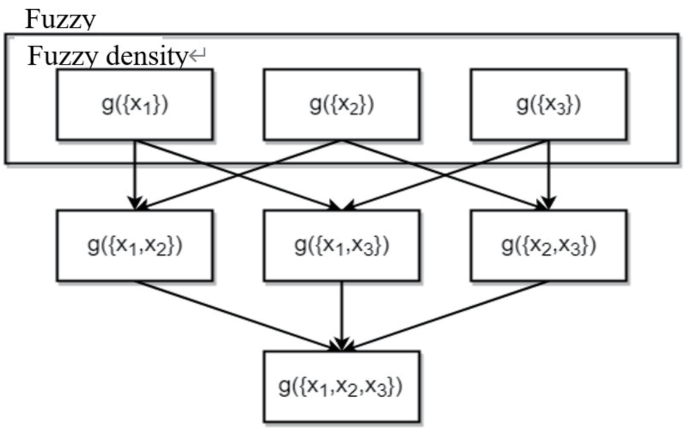

Fig. 10. Relationship between fuzzy measure and fuzzy density.

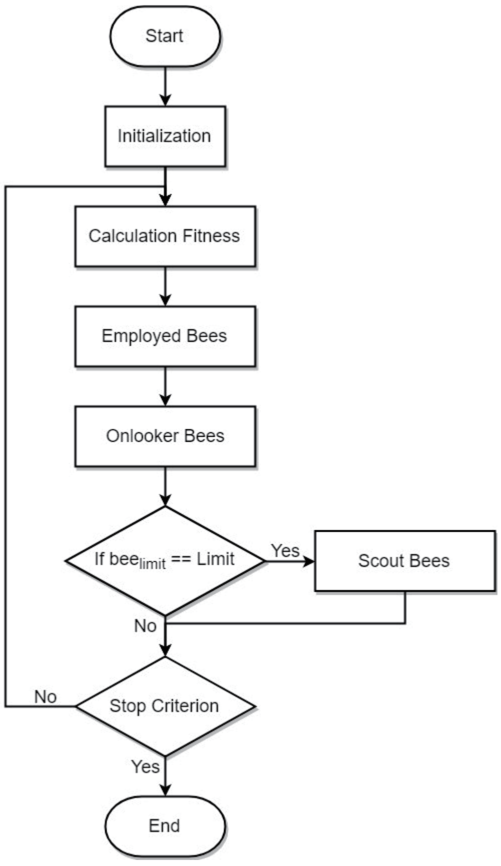

Fig. 11. Flowchart of ABC algorithm.

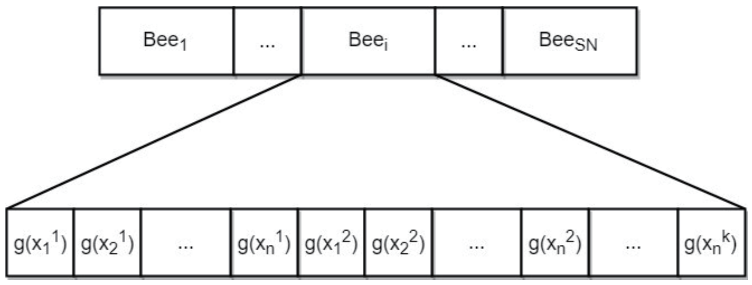

Fig. 12. Coding diagram of fuzzy density solution.

Step 2: Fitness function calculation

After determining the location information of the employed bees, each onlooker uses the roulette method to determine which food source to search for and calculate the benefit of the food source. The probability P_i that the ith food source is selected is given by

$$P_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_n}, \tag{7}$$

where fit_i is the profitability of the ith food source and SN is the total number of food sources.

Step 3: Swarm location update

The number of employed bees and onlookers in the algorithm is usually equal to half the population. The initial position of each employed bee is moved to the new food source position in accordance with the following formula:

$$v_i^j = x_{ij} + rand\left[-1,1\right]\left(x_i^j - x_k^j\right), \tag{8}$$

where v_i^j is the new position of the ith employed bee after the jth dimension moves, x_i^j is the position of the ith employed bee before the jth dimension moves, and x_k^j is the position of the jth dimension of another employed bee k that is randomly selected.

Step 4: Check "limit" value

To present the algorithm from falling into a regional optimal solution, if the profitability of any food source has not been improved after a set number of searches ("limit" value), the food source is discarded, the employed bee becomes a scout bee, and the abandoned food source is replaced by another new food source in accordance with Eq. (6).

Step 5: Loop until termination conditions are met

An assessment is conducted to determine whether the termination conditions are met or the maximum number of iterations is reached. If the conditions are met, the algorithm ends and the optimal solution is output. Otherwise, the algorithm returns to Step 2 to continue its execution.

After the fuzzy density is determined, the fuzzy measure is calculated as follows:

$$g\left(A \cup B\right) = g(A) + g(B) + \lambda g(A) g(B),\ A, B \subset X, \tag{9}$$

where λ is calculated as

$$\lambda + 1 = \prod_{i=1}^{n}\left(1+\lambda g^{i}\right). \tag{10}$$

The value of λ in the IFI is equal to the output dimension, which indicates that each output category can calculate one λ, and $\lambda \in (-1, \infty)$ with three characteristics: (1) if $\sum_{i=1}^{n} g^{i} = 1$, then $\lambda = 0$; (2) if $\sum_{i=1}^{n} g^{i} < 1$, then $\lambda > 0$; (3) if $\sum_{i=1}^{n} g^{i} > 1$, then $-1 \leq \lambda < 0$.

In the proposed method, two output rules, the Sugeno FI and Choquet FI, are selected. The two FIs are calculated simultaneously, and the higher value is regarded as the final output of the FI. The IFI outputs of the Sugeno and Choquet FIs are described as follows.

(1) Sugeno FI:

$$Y_s = \vee_{i=1}^{N}\left(h\left(x_{\pi i}\right) \wedge g\left(A_i\right)\right). \tag{11}$$

(2) Choquet FI:

$$Y_c = \sum_{i=1}^{n} h\left(x_{\pi i}\right)\left(g\left(A_i\right) - g\left(A_{i-1}\right)\right), \tag{12}$$

where π_i denotes the classifier with the *i*th highest output value.

4. Experimental Results

To verify the performance of the proposed method, we used the public MIT scene dataset and images of various indoor environments as the study dataset. In the experiment, this dataset was divided into three subsets for floor image segmentation: training data, verification data, and test data. Subsequently, the performances of the various image segmentation methods were compared. Evaluation indicators were used to verify that the proposed IFI method can combine the advantages of various classifiers for image segmentation to improve the recognition rate.

To objectively evaluate the performance of each classifier in the proposed method, we used a confusion matrix—a commonly used tool for measuring classification performance—as a quantitative indicator of its performance. This matrix includes the following evaluation indices: accuracy, G-Mean, precision, recall, and F1-score. In the matrix, *TP* means that the classifier accurately predicted an actual positive value to be positive, *TN* means that it accurately predicted an actual negative value to be negative, *FP* means that it inaccurately predicted an actual negative value to be positive, and *FN* means that it inaccurately predicted an actual positive value to be negative. Each evaluation index is described below.

Accuracy

$$Accuracy = \frac{TN + TP}{TN + FP + FN + TP} \tag{13}$$

G-Mean

$$G\text{-}Mean = \sqrt{\frac{TP}{TP+FN} \times \frac{TN}{TN+FP}} \tag{14}$$

Precision (P)

$$P = \frac{TP}{TP+FP} \tag{15}$$

Recall (R)

$$R = \frac{TP}{TP+FN} \tag{16}$$

FI-score

$$FI = \frac{1}{\frac{1}{P}+\frac{1}{R}} = \frac{2 \times P \times R}{P+R} \tag{17}$$

A total of 200, 48, and 48 images were used as training data, verification data, and test data, respectively. The following parameters were set during the training process: number of iterations = 100000, *limit* = 10, and *SN* = 100. Table 1 presents the performances of the proposed method involving various combinations and the performances of the other algorithms (i.e., FCN and DeepLabv2). The experimental results revealed that in the proposed method, the FCN–DeepLabv2–ACD–IFI combination provided an accuracy, G-Mean, precision, recall, and F1-score of 97.83, 96.98, 95.86, 95.65, and 95.75%, respectively, outperforming the FCN, FCN–FI, DeepLabv2, and DeepLabv2–FI algorithms.

Figure 13 presents ground-truth data with the floor area detection results obtained using the proposed method and the IFI, DeepLabv2, and FCN algorithms. As illustrated in Figs. 13(c) and

Table 1
Performance results for different methods.

Methods	Accuracy (%)	G-Mean (%)	Precision (%)	Recall (%)	F1-score (%)
FCN	97.38	96.26	95.40	95.17	95.18
DeepLabv2	97.13	95.80	95.82	93.12	94.45
FCN (SGD + Adagrad + Adam) + FI	97.58	96.67	96.49	94.61	95.54
FCN (SGD + Adagrad + Adam) + IFI	97.74	96.97	95.96	95.40	95.68
DeepLabv2 (SGD + Adagrad + Adam) + FI	96.44	93.73	97.67	88.51	92.87
DeepLabv2 (SGD + Adagrad + Adam) + IFI	97.37	96.22	95.98	93.90	94.92
FCN + DeepLabv2+ ACD + FI	97.48	96.00	96.22	94.04	95.08
FCN + DeepLabv2+ ACD + IFI	97.83	96.98	95.86	95.65	95.75

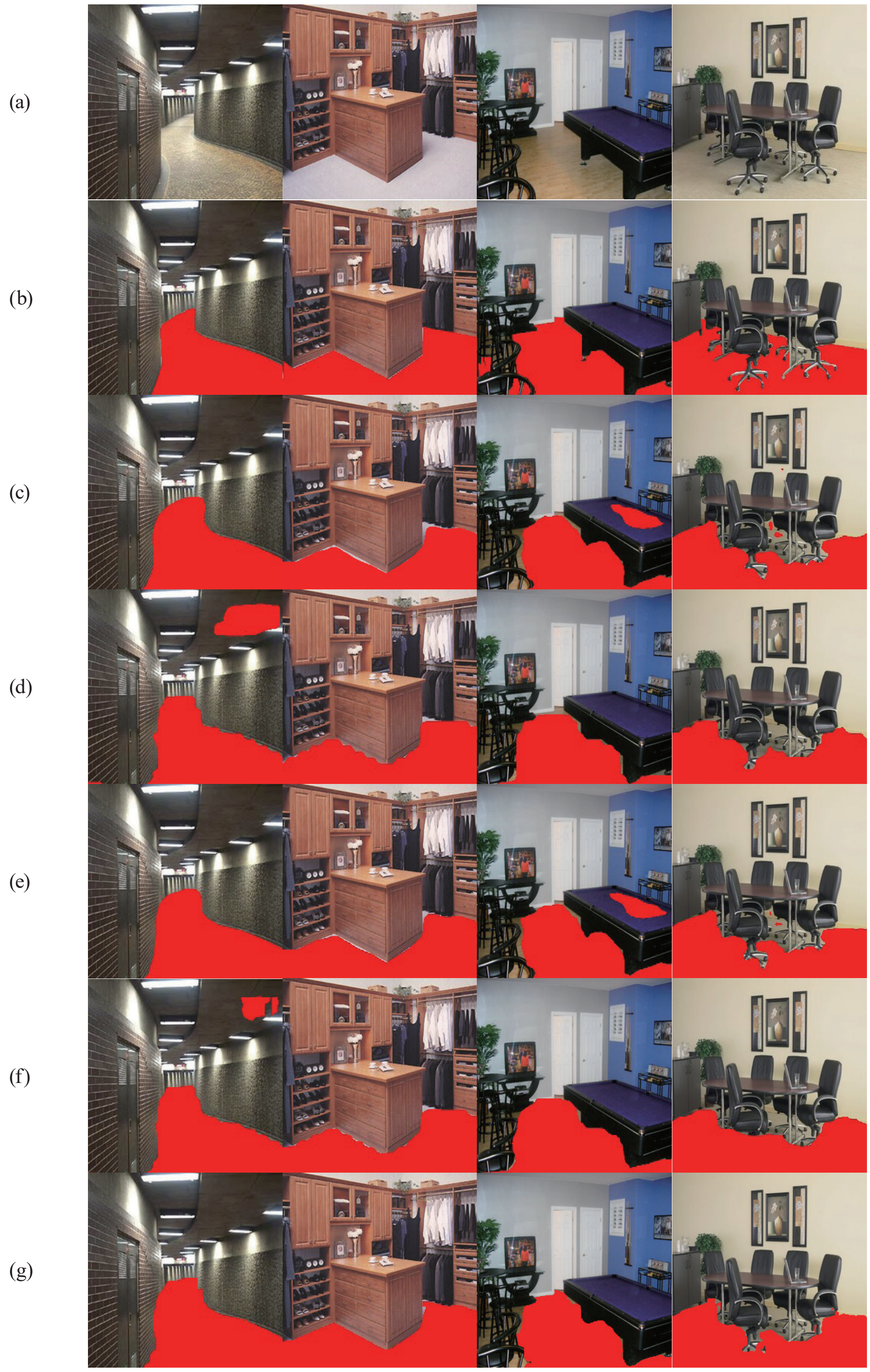

Fig. 13. (Color online) Floor area detection results for different methods: (a) original images, (b) ground-truth images, (c) FCN detection results, (d) DeepLabv2 detection results, (e) FCN (SGD + Adagrad + Adam) – IFI detection results, (f) DeepLabv2 (SGD + Adagrad + Adam) – IFI detection results, and (g) (FCN + DeepLabv2 + ACD) – IFI detection results.

13(d), both the FCN and DeepLabv2 algorithms could detect the floor area. However, they could not successfully detect the finer parts of the floor. For example, smooth objects were misjudged as the floor, and the edge of the floor area was uneven.

Figure 13(g) illustrates the results obtained after the ACD algorithm was used to smooth the edge of the detected floor area. The inclusion of the IFI in the proposed method resulted in more accurate floor area detection results, as depicted in Figs. 13(e)–13(g).

5. Conclusions

We proposed a multiple-classifier method based on the IFI for detecting floor areas. The IFI in the proposed method is different from the traditional FI in that its fuzzy density value is optimized using the ABC algorithm. The proposed method takes the outputs of several classifiers and the ACD algorithm as the input. The IFI is used to evaluate each classifier and extract its advantages. The FCN–DeepLabv2–ACD–IFI combination in the proposed method achieved an accuracy, G-Mean, precision, recall, and F1-score of 97.83%, 96.98%, 95.86%, 95.65%, and 95.75%, respectively. The experimental results indicate that the combination in the proposed method can achieve an average floor area detection accuracy of 97.8%, thus improving the overall recognition rate.

In the future, we will explore how to automatically screen out classifiers with smaller or similar contributions during the evolution process. Thus, the best classifier combination can be automatically determined.

References

1 C. Chun, D. Park, W. Kim, and C. Kim: Proc. IEEE Int. Conf. Image Processing (IEEE, 2013) 3358−3362. https://doi.org//10.1109/ICIP.2013.6738692

2 S. Kumar, M. S. Karthik, and K. M. Krishna: IEEE Int. Conf. Robotics and Automation (IEEE, 2014) 494−500. https://doi.org//10.1109/ICRA.2014.6906901

3 S. Aggarwal, A. M. Namboodiri, and C. V. Jawahar: Int. Conf. Pattern Recognition (2014) 4275−4280. https://doi.org//10.1109/ICPR.2014.733

4 Y. Boykov and V. Kolmogorov: IEEE Trans. Pattern Anal. Mach. Intell. **26** (2004) 1124. https://doi.org//10.1109/TPAMI.2004.60

5 J. Long, E. Shelhamer, and T. Darrell: 2015 Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (IEEE, 2015) 3431−3440. https://doi.org//10.1109/CVPR.2015.7298965

6 V. Badrinarayanan, A. Kendall, and R. Cipolla: IEEE Trans. Pattern Anal. Mach. Intell. **39** (2017) 2481. https://doi.org//10.1109/TPAMI.2016.2644615

7 L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille: IEEE Trans. Pattern Anal. Mach. Intell. **40** (2018) 834. https://doi.org//10.1109/TPAMI.2017.2699184

8 G. Lin, A. Milan, C. Shen, and I. Reid: 2017 Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (IEEE, 2017) 5168−5177. https://doi.org//10.1109/CVPR.2017.549

9 C. S. Asness, T. J. Moskowitz, and L. H. Pedersen: J. Finance **68** (2013) 929. https://doi.org/10.1111/jofi.12021

10 T. Dozat: Int. Conf. Learning Representations (2016) 2013−2016. https://openreview.net/pdf/OM0jvwB8jIp57ZJjtNEZ.pdf

11 J. Duchi, E. Hazan, and Y. Singer: J. Mach. Learn. Res. **12** (2011) 2121. https://www.jmlr.org/papers/volume12/duchi11a/duchi11a.pdf

12 M. D. Zeiler: arXiv:1212.5701 (2012). https://doi.org/10.48550/arXiv.1212.5701

13 Lecture 6a- overview of mini-batch gradient descent: https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf (accessed May 2022).

14 D. P. Kingma and J. Ba: arXiv:1412.6980 (2014) 1. https://doi.org/10.48550/arXiv.1412.6980

15 L. J. Yang: Master thesis, National Formosa University (2022) 1−40. https://hdl.handle.net/11296/6ffggx
16 J. Canny: IEEE Trans. Pattern Anal. Mach. Intell. PAMI-**8** (1986) 679. https://doi.org//10.1109/TPAMI.1986.4767851
17 R. G. von Gioi, J. Jakubowicz, J. M. Morel, and G. Randall: IEEE Trans. Pattern Anal. Mach. Intell. **32** (2010) 722. https://doi.org//10.1109/TPAMI.2008.300
18 R. G. von Gioi, J. Jakubowicz, J. M. Morel, and G. Randall: Image Process. Line **2** (2012) 33. https://doi.org/10.5201/ipol.2012.gjmr-lsd
19 Y. Li and S. T. Birchfield: IEEE/RSJ Int. Conf. Intelligent Robots and Systems (2010) 837−843. https://doi.org//10.1109/IROS.2010.5652818
20 A. AbuBaker, R. Qahwaji, S. Ipson, and M. Saleh: IEEE Int. Conf. Signal Processing and Communications (IEEE, 2007) 1283−1286. https://doi.org//10.1109/ICSPC.2007.4728561
21 T. Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie: 2017 Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR) (IEEE, 2017) 936−944. https://doi.org//10.1109/CVPR.2017.106
22 P. Chiranjeevi, V. Goresultingpalakrishnan, and P. Moogi: IEEE Trans. Image Process. **24** (2015) 2701. https://doi.org//10.1109/TIP.2015.2421437
23 N. Bouadjenek, H. Nemmour, and Y. Chibani: IET Biom. **6** (2017) 429. https://doi.org//10.1049/iet-bmt.2016.0140
24 B. Tong and Y. Liu: 2016 Proc. IEEE Conf. Robotics and Biomimetics (IEEE, 2016) 1337−1342. https://doi.org//10.1109/ROBIO.2016.7866512
25 C. J. Lin, S. S. Kuo, and C. C. Peng: Int. J. Fuzzy Syst. **14** (2012) 380.
26 D. Karaboga and B. Basturk: J. Global Optim. **39** (2007) 459. https://doi.org//10.1007/s10898-007-9149-x

About the Authors

Chi-Huang Shih received his Ph.D. degree in electrical engineering from National Cheng Kung University, Taiwan, R.O.C., in 2008. Currently, he is an associate professor in the Computer Science and Information Engineering Department, National Chin-Yi University of Technology, Taichung, Taiwan, R.O.C. His current research interests are machine learning, multimedia network communication, biomedical worn devices, embedded systems, and IoT applications.

Cheng-Jian Lin received his B.S. degree in electrical engineering from Ta Tung Institute of Technology, Taipei, Taiwan, R.O.C., in 1986 and his M.S. and Ph.D. degrees in electrical and control engineering from National Chiao Tung University, Taiwan, R.O.C., in 1991 and 1996, respectively. Currently, he is a chair professor of the Computer Science and Information Engineering Department, National Chin-Yi University of Technology, Taichung, Taiwan, R.O.C., and dean of Intelligence College, National Taichung University of Science and Technology, Taichung, Taiwan, R.O.C. His current research interests are machine learning, pattern recognition, intelligent control, image processing, intelligent manufacturing, and evolutionary robots.
(cjlin@ncut.edu.tw)

Jyun-Yu Jhang received his B.S. and M.S. degrees from the Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung, Taiwan, in 2013 and 2015 and his Ph.D. degree from Institute of Electrical and Control Engineering in 2021. Currently, he is an assistant professor in the College of Intelligence, National Taichung University of Science and Technology, Taichung, Taiwan. His current research interests include fuzzy logic theory, type-2 neural fuzzy systems, evolutionary computation, machine learning, computer vision, and applications.
(jyjhang@nutc.edu.tw)