Hand Gesture Recognition in Complex Background Based on Convolutional Pose Machine and Fuzzy Gaussian Mixture Models

Tong Zhang[1] · Huifeng Lin[2] · Zhaojie Ju[3] · Chenguang Yang[4]

Abstract Hand gesture is one of the most intuitive and natural ways for human to communicate with computers, and it has been widely adopted in many human–computer interaction applications. However, it is still a challenging problem when confronted with complex background, illumination variation and occlusion in real-world scenarios. In this paper, a two-stage hand gesture recognition method is proposed to tackle these problems. At the first stage, hand pose estimation is developed to locate the hand keypoints using the convolutional pose machine, which can effectively localize hand keypoints even in a complex background. At the second stage, the Fuzzy Gaussian mixture models (FGMMs) are tailored to reject the nongesture patterns and classify the gestures based on the estimated hand keypoints. Extensive experiments are conducted to evaluate the performance of the proposed method, and the result demonstrates that the proposed algorithm is effective, robust, and satisfactory in real-time scenarios.

Tong Zhang and Huifeng Lin are joint first authors and they contributed equally to this work.

✉ Chenguang Yang
cyang@ieee.org

1   School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China

2   Key Laboratory of Autonomous Systems and Networked Control, College of Automation Science and Engineering, South China University of Technology, Guangzhou 510640, China

3   School of Computing, University of Portsmouth, Portsmouth, UK

4   Bristol Robotics Laboratory, University of the West of England, Bristol BS16 1QY, UK

1 Introduction

Recently, with the development of computer vision and machine learning, human–computer interaction has been playing an important role in people's daily life. Compared to the traditional two-dimensional graphical user interface, the ultimate goal of human–computer interaction is to realize the natural communication between the human and the computer and provide the operator with a more intuitive and comfortable interactive experience. Kinds of research on interactive techniques about face, gait, gestures, and posture have been carried out. Among these interaction methods, hand gesture is the most intuitive and natural one which has aroused great attention of researchers.

Gestures are used to convey information and includes static gestures and dynamic gestures. Hand detection and tracking are the main difficulties in gesture recognition. Early researches used data glove or mark-based methods to deal with this problem [1], but they both require additional equipment, making the recognition system uncomfortable and inconvenient for users. Compared to the wearable device-based gesture recognition, vision-based gesture recognition system enables users to communicate with computers more naturally with a low-cost camera.

Hand segmentation plays an important role in most of the vision-based gesture recognition system, which aims to segment the hand from the backgrounds. Some of the research analyzed and tested their algorithms in simple background like a white wall [2–4], which facilitated the data preprocessing by simply thresholding the original

images. However, the background is usually complex in real-world scenarios. Hand segmentation methods cannot handle the complex background well, which makes the overall gesture recognition system sensitive to complex background, illumination variation, and occlusion.

Several researchers bring up solutions to the complex background problem in gesture recognition [5–8]. Yin and Xie [8] employed a restricted coulomb energy neural network to segment the hand from complex backgrounds. However, the performance of the segmentation is not very satisfactory because it is solely based on skin colors. Pisharady et al. [9] employed a Bayesian model of visual attention combining low-level and high-level image features to produce a saliency map, which helps in hand segmentation. This method can work well with backgrounds including skin-colored complex backgrounds. However, the processing speed of this algorithm is not satisfactory to real-time requirements and it needs 2.65 s to process every single image. Dominio et al. [10] utilized depth information to address the problems of illumination changes and complex backgrounds. However, the RGB-D camera is required to obtain the depth images, which limits the usage of this method.

In this paper, we present a static gesture recognition approach which consists of two stages, a hand pose estimator and a hand pose classifier. The former is used to estimate the hand keypoints locations, while the latter classifies these predicted locations into different categories. We first train a hand pose estimator based on a network architecture named convolutional pose machine [11] using data collected in different backgrounds. Due to its special network structure, it can handle the problem of complex background and occlusion well. The convolutional pose machine takes an RGB image of a human hand as input, the output of which is heatmaps for each hand keypoint. We could obtain the location of each hand keypoint according to these heatmaps. Then, these location features are fed to a classifier to predict the category of the corresponding gesture. Considering that the ability to reject unknown categories is necessary for a gesture recognition system, we modify the Fuzzy Gaussian Mixture Models (FGMM) [12] to act as the classifier. FGMM is a kind of generative model, which can properly filter out the nontarget gesture and meanwhile the time it takes to classify the data is very little and could be neglected. Therefore, the overall gesture recognition system can recognize gestures in complex background in real time.

In summary, the main contribution of this paper is given as follows:

– We propose a two-stage gesture recognition method in this paper, which is based on robust hand pose estimation to tackle the problem of complex background.
– A hand pose classifier based on Fuzzy Gaussian Mixture Models is proposed to classify the gesture which performs well in rejecting the nongestures with limited numbers of nongesture training samples.
– Extensive experiments have been conducted to test the performance of the proposed method and the result demonstrates that our algorithm is effective, robust to complex backgrounds, and satisfactory to real-time requirements.

The remainder of this paper is structured as follows. Related works are reviewed and discussed in Sect. 2. Detailed description of our proposed algorithm is given in Sects. 3, 4, and 5. The experimental results implemented with the proposed method and related analysis are shown in Sect. 6. The conclusion is summarized in the last section.

2 Related Works

In this section, we present a brief review of the related works on hand segmentation and gesture recognition.

2.1 Hand Segmentation

Hand segmentation and detection is the foundation of a gesture recognition system, which has a large influence on the performance of the overall gesture recognition algorithm. The main purpose of hand detection is to localize the human hand for a given image and hand segmentation aims to separate the human hand from the background.

Among numerous works on hand segmentation, skin color segmentation is the most commonly used method. Researchers tried to segment the human hands based on skin color on different color spaces such as RGB color space, YUV color space, and YCbCr color space [13–15]. The approach was proposed by Jones and Rehg [16], which applied Bayesian classifier for skin color segmentation. These segmentation methods are robust to the hand shape variation, but when the light condition changes a lot or the background color is similar to the color of skin, the performance of hand segmentation is not guaranteed.

The movement of the hand is utilized for hand segmentation to deal with the problems above [17]. However, this method also has its limitation that it only works well with moving hands in fixed backgrounds. Another solution is to harness depth information in hand segmentation like the methods proposed in Refs. [18–21]. However, the RGB-D camera is required to obtain the depth images and is limited to be used indoors.

2.2 Gesture Recognition

After extracting features of the hand region segmented from the original image, gesture recognition aims to classify these features to a specific category of gestures. Extensive gesture recognition methods have been proposed in recent years.

Different kinds of features are designed and utilized in these methods. Priyal and Bora [22] used edge feature to match the test patterns and the saved patterns, while [23] utilizes Haar-like features to identify specific gestures. Pisharady et al. [9] employed a Bayesian model of visual attention combining low-level and high-level image features to produce a saliency map, which helps to hand segmentation. And these features of hand region are combined properly and then fed to an SVM classifier to predict the hand gesture. Dardas and Georganas [24] extracted the features of the image using scale invariance feature transform (SIFT) and maps them into a bag-of-words vector, which will be fed to a multiclass SVM to make final classification decisions.

Instead of designing features manually, researchers turn to deep learning-based approaches which are able to learn features from training data automatically [25]. Stacked denoising autoencoder and convolutional neural network are applied to the task of static gesture recognition by Oyedotun and Khashman [2]. In the study by Liang et al. [26], the convolutional neural network is treated as a feature extractor and the extracted features are then fed to an SVM classifier.

Most of the researches only discuss the problem of gesture recognition in simple background like a white wall. The performance is not guaranteed when they are confronted with complex background and illumination variation. In this paper, we propose a two-stage gesture recognition method to tackle the problem of complex background, which is inevitable in real-world scenarios. The experimental results show that our algorithm has good performance and also meets the real-time requirements.

3 Hand Pose Estimation

The main purpose of hand pose estimation is to localize hand keypoints, which can facilitate the subsequent procedure of gesture recognition. In order to obtain a hand pose estimator which is robust to the complex background, we tailor the method proposed by Wei et al. [11] called convolutional pose machine (CPM), which is originally used for human pose estimation. In this paper, the CPM takes an RGB image of a human hand as input and the output are heatmaps for each hand keypoint. We consider 21 hand keypoints in this paper which are denoted as the

blue points in Fig. 1, and consequently the CPM generates 22 heatmaps in total including one for the background.

3.1 Network Architecture

CPM is a combination of the convolutional architectures and the pose machine architecture [27]. Therefore, it is not only able to learn feature representations automatically from the training dataset, but also able to learn and infer the long-range relationships between keypoints, which is very suitable for hand keypoints localization.

A CPM consists of several stages, which forms a sequential architecture. Each stage of the CPM takes the heatmaps generated by its previous stage and the image features extracted by a CNN architecture as input and outputs refined heatmaps, except for the first stage which only takes the image features as input. This can be formulated as

$$P_{t+1} = g_{t+1}(P_t, f(X)), t \in \{1, \ldots, T-1\} \tag{1}$$

where P_t denotes the output of stage t, $f(X)$ denotes the features extracted from image X, and T represents the number of the stages.

This sequential architecture enables the overall network to infer the relationships between keypoints. It can leverage the spatial context information of previous heatmaps to infer the difficult-to-detect keypoints from the easier-to-detect keypoints or infer the occluded parts and the undistinguished parts from the detected parts. This ability ensures the performance of the hand pose estimator, which can work well even under challenging situation such as occlusion, complex background, and light changes.

The network architecture of CPM used in this paper is depicted in Fig. 2, which is actually a kind of fully convolutional networks (FCN) [28] composed of only convolutional layers and pooling layers. The feature extractor is modified from the VGGNet [29], which consists of several convolutional layers and pooling layers. The CONV2 and
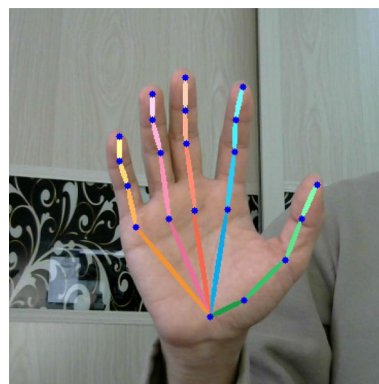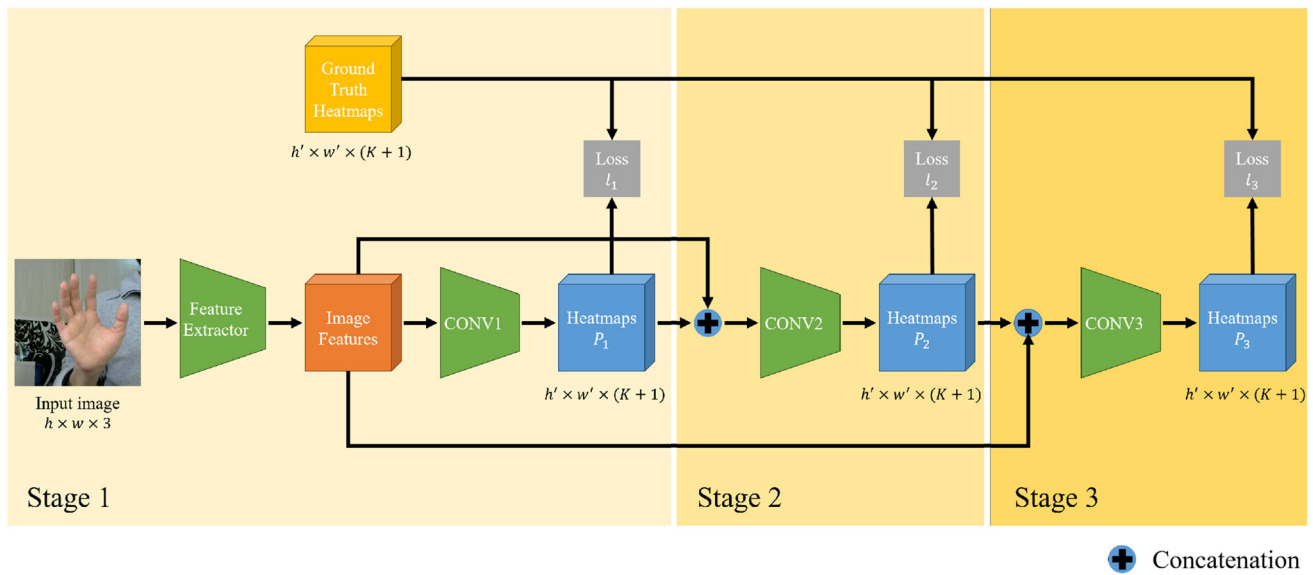

Fig. 1 The hand keypoints

Fig. 2 The network architecture

CONV3 in this figure have the same architecture, which is a stack of several convolutional layers. However, they do not share the same parameters.

As shown in Fig. 2, there are three stages in the CPM, each of which produces its own heatmaps to predict the locations of the keypoints. These stages predict the locations from coarse to fine. The former stages can only make rough predictions because the corresponding effective receptive fields are small, in other words, they can only see a small patch of the input image. On the contrary, the latter stages have larger effective receptive fields covering a large patch of the input image, which helps them better leverage the spatial context information provided by previous stage and image texture features, thus they can make accurate predictions. Although the outputs of the former stages are noisy, they are necessary and informative and they can provide strong cues for the latter stages. This sequential architecture allows the hand pose estimator to infer step by step, instead of forcing it to predict the accurate localization just in one step.

In this work, in order to reduce the number of the parameters of the network, these three stages share the same feature extractor to provide the same image texture features. Besides, considering that hand pose estimation does not need very large receptive fields as human pose estimation does, all the convolutional filters in this network use small kernel size such as 1×1 or 3×3 while [11] uses large kernel size such as 9×9 or 11×11. These changes can also improve the computational efficiency.

3.2 Network Training

Convolutional neural networks with too many layers like CPM are prone to encounter the problem of vanishing gradients in the training phase [30–32]. It means that the magnitude of the gradients of the layers close to the input layer is likely to vanish during training and the parameters of these layers will not be updated. This problem prevents deep neural networks from being well trained. In order to tackle the problem of vanishing gradients, [11] introduces intermediate supervision into CPM, which is easy to implement in this sequential prediction framework.

Although the output of each stage relies on the contextual information provided by its previous stage, all of these stages are expected to make prediction for the localization of the hand keypoints as possible as they can. To encourage each stage to achieve the same goal, the same loss function is defined for each stage, which aims to minimize the l_2 distance between the output of each stage and the ground truth heatmaps. Therefore, the cost function of stage t can be formulated as

$$l_t = \sum_{k=1}^{K+1} \left\| P_t^k - G^k \right\|_2^2 \tag{2}$$

where K denotes the number of hand keypoints, P_t^k denotes the output of stage t corresponding to the k-th keypoint, and G^k denotes the ground truth heatmap of the k-th keypoint. Note that all P_t and G are tensors and the shape of them are the same, which is $h' \times w' \times c'$. h' and w' are the height and width of the output of each stage, respectively, and c' is the number of channels of the output, which is equal to $K + 1$ in this network. And the overall loss function of the whole

network is the sum of the loss of each stage, which is given by

$$\mathcal{L} = \sum_{t=1}^{T} l_t \tag{3}$$

where T is the number of stages. Since this neural network is fully differential, all the T stages can be jointly trained using backpropagation [33].

There are $K + 1$ ground truth heatmaps for each input image including K hand keypoints and one background. The ground truth heatmap corresponding to k-th keypoint is generated according to a 2D Gaussian function centered at the actual location of this keypoint, which can be given by

$$G^k(x, y) = e^{-\frac{(x-x_k)^2 + (y-y_k)^2}{2\sigma^2}}, k \in \{1, \ldots, K\} \tag{4}$$

where $G^k(x, y)$ is the intensity of the ground truth heatmap at coordinate (x, y), (x_k, y_k) denotes the actual location of the k-th keypoint and σ is the standard deviation which is predefined. The background heatmap $G^{K+1}(x, y)$ is obtained by

$$G^{K+1}(x, y) = 1 - \max_{k \in \{1, \ldots, K\}} (G^k(x, y)) \tag{5}$$

By now, the shape of these generated ground truth heatmaps is consistent to the shape of the input image. Since the CPM architecture contains several pooling layers, the shape of the outputs of each stage is scaled down to $h' \times w'$. In order to maintain the consistency of the shape, the ground truth heatmaps are also resized to $h' \times w'$ by a downsampling operation.

3.3 Network Prediction

In the prediction phase, an RGB image is fed into the trained network and each stage of the network outputs $K + 1$ heatmaps. The output of the last stage is the most predictive among these predictions because it can acquire enough spatial context information and image texture information. Therefore, it is chosen to make the final prediction. The intensity of the pixels in a heatmap can be viewed as the probability that specific keypoint is located at this position. The predicted location of the k-th keypoint is calculated as

$$(\bar{x}_k, \bar{y}_k) = \underset{(x,y)}{\arg\max} P_T^k(x, y), k \in \{1, \ldots, K\} \tag{6}$$

where $(\bar{x}_k, \bar{y}_k)$ denotes the predicted location of the k-th keypoint and P_T^k is the output of the last stage corresponding to that keypoint. If the sum of the intensity of all predicted keypoints is lower than a predefined threshold, it can be considered that this image contains no hands.

The predicted locations of these hand keypoints are considered as the feature of the input RGB image, which is independent of the background of the image. And then these features are fed into the hand pose classifier for further gesture recognition, which will be discussed in Sect. 5.

4 Fuzzy Gaussian Mixture Models

In applications such as machine learning, pattern recognition, or computer vision, the data often have irregular probability distribution patterns. Mixture model is a kind of probability model used to establish these irregular probability distribution patterns. It is the combination of several probability density functions called mixture components which have the same form. In general, a mixture model can be given by

$$p(\mathbf{x}|\boldsymbol{\Theta}) = \sum_{k=1}^{m} \alpha_k p(\mathbf{x}|\boldsymbol{\theta}_k) \tag{7}$$

where $\boldsymbol{\Theta} = \{\alpha_k, \boldsymbol{\theta}_k\}_{k=1}^{m}$ denotes the parameters set of the mixture model, m denotes the number of components, $\boldsymbol{\theta}_k$ is the parameter of the mixture component $p(\mathbf{x}|\boldsymbol{\theta}_k)$, and α_k is the weight of the mixture component. These mixture weights should be nonnegative and satisfy $\sum_{k=1}^{m} \alpha_k = 1$, which ensure that the integral of the overall probability density model is equal to 1.

$$\int p(\mathbf{x}|\boldsymbol{\Theta})d\mathbf{x} = 1 \tag{8}$$

Gaussian mixture model is the most commonly used mixture model [34–37]. The mixture component of this model is the Gaussian distribution, which is given by

$$\begin{aligned} p(\mathbf{x}|\boldsymbol{\theta}_k) &= \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \Sigma_k) \\ &= (2\pi)^{-d/2}|\Sigma_k|^{-1/2} \exp\left\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_k)^T \Sigma_k^{-1}(\mathbf{x} - \boldsymbol{\mu}_k)\right\} \end{aligned} \tag{9}$$

where $\boldsymbol{\mu}_k$ and Σ_k are the mean and covariance of the Gaussian distribution, respectively. Given a set of samples $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^{n}$, the likelihood function of the mixture model can be obtained by

$$\mathcal{L}(\boldsymbol{\Theta}|\mathcal{X}) = p(\mathcal{X}|\boldsymbol{\Theta}) = \prod_{i=1}^{n} \sum_{k=1}^{m} \alpha_k p(\mathbf{x}_i|\boldsymbol{\theta}_k) \tag{10}$$

For the convenience of analysis, the log-likelihood function is often used instead of the likelihood function, which is given by

$$\log \mathcal{L}(\boldsymbol{\Theta}|\mathcal{X}) = \sum_{i=1}^{n} \log \left\{ \sum_{k=1}^{m} \alpha_k p(\mathbf{x}_i|\boldsymbol{\theta}_k) \right\} \tag{11}$$

Maximum likelihood estimation (MLE) is a commonly used method to estimate the unknown parameters of a probability density function (PDF). The objective of MLE is to find $\boldsymbol{\theta}^*$ that maximizes $\log \mathcal{L}(\boldsymbol{\Theta}|\mathcal{X})$:

$$\boldsymbol{\theta}^* = \arg \max_{\boldsymbol{\theta}} \log \mathcal{L}(\boldsymbol{\theta}|\mathcal{X}) \tag{12}$$

For certain distributions, it is very easy to estimate the parameters by directly maximizing the log-likelihood function through taking the partial derivative with respect to the parameters. However, this direct method is not practical for the GMM. We can see in Eq. 10 that the log-likelihood $\log \mathcal{L}(\boldsymbol{\Theta}|\mathcal{X})$ contains the logarithm of the addition, which makes the solution of Eq. 12 difficult.

Expectation–maximization (EM) algorithm is an effective method for the MLE of mixture models [38]. EM algorithm estimates the parameters in an iterative way by introducing an auxiliary function Q given by

$$Q = \sum_{i=1}^{n} \sum_{k=1}^{m} \omega_{ik} \log\{\alpha_k p(\mathbf{x}_i|\boldsymbol{\theta}_k)\} \tag{13}$$

where ω_{ik} denotes the posteriori probability for the k-th component and it can obtained by

$$\begin{aligned} \omega_{ik} &= p(\boldsymbol{\theta}_k|\mathbf{x}_i) \\ &= \frac{\alpha_k \, p(\mathbf{x}_i|\boldsymbol{\theta}_k)}{\sum_{l=1}^{m} \alpha_l \, p(\mathbf{x}_i|\boldsymbol{\theta}_l)}. \end{aligned} \tag{14}$$

Instead of maximizing $\log \mathcal{L}(\boldsymbol{\Theta}|\mathcal{X})$ directly, EM algorithm maximizes the function Q iteratively, which is much easier just by taking the partial derivative with respect to α_k and $\boldsymbol{\theta}_k$. It guarantees that the log-likelihood increases monotonically until it reaches the local maximum. Given a set of samples $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^{n}$, the procedure of EM algorithm to estimate the parameters of GMM is presented as follows:

- E-step: calculate the posteriori probability of every data point for each component using Eq. 15.
- M-step: update all the parameters of GMM according to the current parameters using Eqs. 16–19.

$$\omega_{ik}^{t} = \frac{\alpha_k^t \mathcal{N}(\mathbf{x}_i|\boldsymbol{\mu}_k^t, \Sigma_k^t)}{\sum_{l=1}^{m} \alpha_l^t \mathcal{N}(\mathbf{x}_i|\boldsymbol{\mu}_l^t, \Sigma_l^t)} \tag{15}$$

$$n_k^t = \sum_{i=1}^{n} \omega_{ik}^t \tag{16}$$

$$\alpha_k^{t+1} = \frac{n_k^t}{n} \tag{17}$$

$$\boldsymbol{\mu}_k^{t+1} = \frac{1}{n_k^t} \sum_{i=1}^{n} \omega_{ik}^t \mathbf{x}_i \tag{18}$$

$$\Sigma_k^{t+1} = \frac{1}{n_k^t} \sum_{i=1}^{n} \omega_{ik}^t (\mathbf{x}_i - \boldsymbol{\mu}_k^{t+1})(\mathbf{x}_i - \boldsymbol{\mu}_k^{t+1})^T \tag{19}$$

This procedure is repeated several times until the log-likelihood value converges. EM algorithm is sensitive to the initial values and K-means [39] is often utilized for a good initialization.

Since it may take many times of iteration for the vanilla EM algorithm to converge, to improve the computational efficiency, fuzzy membership is incorporated into the EM algorithm to become Fuzzy Gaussian Mixture Models (FGMM) [40], inspired by the implement of Fuzzy C-means algorithm [41, 42]. The only difference between vanilla EM algorithm and fuzzy EM algorithm lies in the parameters updating formulas, which is derived as follows.

In FGMM, a dissimilarity function is introduced to better describe the distance between the data and clustering centers, which is given by

$$d_{ik}^2 = \frac{1}{\alpha_k p(\mathbf{x}_i|\boldsymbol{\theta}_k)} \tag{20}$$

And the degree of membership u_{ik} is computed according to Eq. 21, which is originally from Fuzzy C-means algorithm [41].

$$u_{ik} = \left[\sum_{j=1}^{m} \left(\frac{d_{ik}}{d_{ij}} \right)^{\frac{2}{m-1}} \right]^{-1} \tag{21}$$

By combining Eqs. 20 and 21, a key formula can be obtained:

$$u_{ik}^z = \frac{[\alpha_k p(\mathbf{x}_i|\boldsymbol{\theta}_k)]^{\frac{z}{z-1}}}{\left[\sum_{j=1}^{m} (\alpha_j p(\mathbf{x}_i|\boldsymbol{\theta}_j))^{\frac{1}{z-1}} \right]^z} \tag{22}$$

where $p(\mathbf{x}_i|\boldsymbol{\theta}_k)$ can be obtained by Eq. 9 and z represents the degree of fuzziness. The procedure of FGMM to estimate the parameters is similar to the EM algorithm, except that the parameters updating equations are changed as follows:

$$\alpha_k^{t+1} = \frac{\sum_{i=1}^{n} u_{ik}^z}{\sum_{k=1}^{m} \sum_{i=1}^{n} u_{ik}^z} \tag{23}$$

$$\boldsymbol{\mu}_k^{t+1} = \frac{\sum_{i=1}^{n} u_{ik}^z x_i}{\sum_{i=1}^{n} u_{ik}^z} \tag{24}$$

$$\Sigma_k^{t+1} = \frac{\sum_{i=1}^{n} u_{ik}^z (x_i - \boldsymbol{\mu}_k^{t+1})(x_i - \boldsymbol{\mu}_k^{t+1})^T}{\sum_{i=1}^{n} u_{ik}^z} \tag{25}$$

The equations above are used to update the parameters of FGMM, which will be further discussed in Sect. 5.2. The experimental results in [40, 43] demonstrate that when the

number of components m is greater than 1, the incorporation of fuzziness into EM algorithm accelerates the convergence with a fewer number of iterations when compared to the conventional EM algorithm.

5 Hand Pose Classifier Based on FGMM

In this section, we discuss the gesture recognition based on the hand pose obtained from Sect. 3. Not only does the algorithm need to classify the gesture accurately, but it also has to reject unknown classes. The ability to reject unknown category is very necessary for an automatic gesture recognition system.

Since the number of nongestures without specific patterns can be almost infinite, it is not practical to obtain the set of nontarget gesture training samples. To handle this problem, we modify the Fuzzy Gaussian Mixture Models to act as the gesture classifier in this second stage. Actually, FGMM is a kind of generative model, and it is suitable to filter out the nontarget gesture categories. The process of the classification can be summarized as follows: FGMM is first employed to estimate the probability distribution of the known categories using training samples. When given a testing sample, the corresponding likelihood is calculated based on this FGMM. If the likelihood is lower than a predefined threshold, it is considered as an unknown gesture. Otherwise, it is considered as a target gesture and needs further classification.

5.1 Feature Preprocessing

After hand pose estimation, we obtain 21 two-dimensional coordinates, each of which is corresponding to one of the hand keypoints, and we have to preprocess these data and design effective features to facilitate the following classification.

We denote the location of the k-th hand keypoint as $Z_k = (x_k, y_k)$ and denote the location of the hand wrist as $Z_1 = (x_1, y_1)$ for convenience. To ensure that features are invariant to shifting, all coordinates of these keypoints are subtracted by the coordinate of the hand wrist. It means that we use the relative position instead of the absolute position. This transformation can be given by

$$Z_k = Z_k - Z_1, k \in \{2, \ldots, 21\} \tag{26}$$

Note that Z_k is a 2D vector, therefore the subtraction operation here means the subtraction of vectors. After this transformation, the coordinate of the hand wrist is aligned to the origin of coordinates, which can be ignored now.

Invariance to scaling is also important for a gesture recognition system because the distance between the human hand and the camera is not fixed. To ensure that

features are invariant to scaling, all coordinates of these keypoints are scaled by the maximum norm value according to the following formulas:

$$I = \arg\max_{i \in \{2, \ldots, 21\}} \|Z_i\|_2 \tag{27}$$

$$Z_k = \frac{Z_k}{\|Z_I\|_2}, k \in \{2, \ldots, 21\} \tag{28}$$

where $\|Z_i\|_2$ denotes the l_2 distance between Z_i and the origin. After scaling, the norms of these coordinates are all smaller than or equal to 1, which helps to normalize the features. Then, these coordinates are concatenated one by one to become a single feature vector and the length of this vector is $20 \times 2 = 40$ ignoring the coordinate of the hand wrist. The processed feature vectors are then fed to the classifier for training or testing.

5.2 Classifier Training

Given a set of training samples with labels, we assume that they cluster around several centers well in the feature space. We first employ FGMM to estimate the probability distribution of these data, which is very useful to reject unknown category in the prediction phase. Since the labels of training samples have been given, we know the number of gesture categories actually. Therefore, we can set the number of mixture components m equal to the number of gesture categories and estimate the parameters of these components according to Eqs. 22–25.

After the convergence of the modified EM algorithm, we obtain m sets of parameters $\{\alpha_k, \boldsymbol{\mu}_k, \Sigma_k\}_{k=1}^m$. However, since the training of FGMM is in an unsupervised way, the mapping relationships between the mixture components and the actual gesture categories are unknown. We need to obtain these mapping relationships according to the labels of the training samples. First, each training sample is assigned to the mixture component with maximum posterior probability, which is given by

$$\mathcal{X}_k = \{\mathbf{x}_i | p(\boldsymbol{\theta}_k | \mathbf{x}_i) = \max_{l=1,\ldots,m} p(\boldsymbol{\theta}_l | \mathbf{x}_i), i = 1, \ldots n\} \tag{29}$$

where $\mathcal{X}_k$ is the set of samples belonging to k-th component and $p(\boldsymbol{\theta}_k | \mathbf{x}_i)$ denotes the posteriori probability that can be obtained by Eq. 14. Note that $\mathcal{X}_k$ contains samples with different labels and we assign the label that has the maximum number to component k. We denote the gesture categories as $Q = \{q_1, \ldots, q_m\}$ and the mapping is given by

$$k \leftarrow \arg\max_{q \in Q} |\{\mathbf{x}_i | \mathbf{y}_i = q, \mathbf{x}_i \in \mathcal{X}_k\}| \tag{30}$$

where $\mathbf{y}_i$ is the label of $\mathbf{x}_i$ which satisfies $\mathbf{y}_i \in Q$ and $|\cdot|$ denotes the cardinality of the set.

Consequently, each mixture component is associated with a certain gesture category and the mapping relationships between the mixture components and the actual gesture categories are established.

5.3 Classifier Prediction

After the training phase, all parameters of the FGMM have been obtained, which is denoted as $\boldsymbol{\Theta}$. Given a testing sample $\mathbf{x}$, we can compute the likelihood $p(\mathbf{x}|\boldsymbol{\Theta})$ by Eq. 7. And the equation below determines the approval or rejection of the testing data as a target gesture.

$$p(\mathbf{x}|\boldsymbol{\Theta}) > \tau \tag{31}$$

If the likelihood is lower than the predefined threshold value τ, this sample is considered as a nongeseture pattern. Otherwise, it is considered as a target gesture and needs further classification. We can obtain the posteriori probability for each component $p(\boldsymbol{\theta}_k|\mathbf{x})$ by Eq. 14 and assign this sample to the component with the maximum posteriori value. According to the mapping relationships given by Eq. 30, we can finally classify the sample to the corresponding gesture category.

6 Experiments

6.1 Analysis of CPM

The network architecture presented in Fig. 2 is implemented using Tensorflow [44] deep learning framework. Considering that the hand pose estimator should be robust to the complex background, we adopt the Rendered Hand Pose (RHD) dataset [45] for training. This dataset is composed of a large number of synthetic images and the background of each of these images is randomly sampled from a pool of 1231 images taken in different cities and landscapes. The creation method of this dataset ensures that it has enough variance of the background.

We crop the training images to make them center around the region of hands and resize the cropped images to 368×368. Data augmentation techniques such as rotation, shifting, and scaling are used in the training phase. We train on this dataset for 150 epochs which are enough for the convergence and visualize the performance of this network on some images taken in real-world scenarios. The evaluation of the overall gesture recognition system based on this CPM will be given in the following section.

The output of each stage is presented in Fig. 3. The input image shown in Fig. 3a is fed to the trained network and we can obtain 22 heatmaps at each stage. For the convenience of visualization, we combine 21 heatmaps
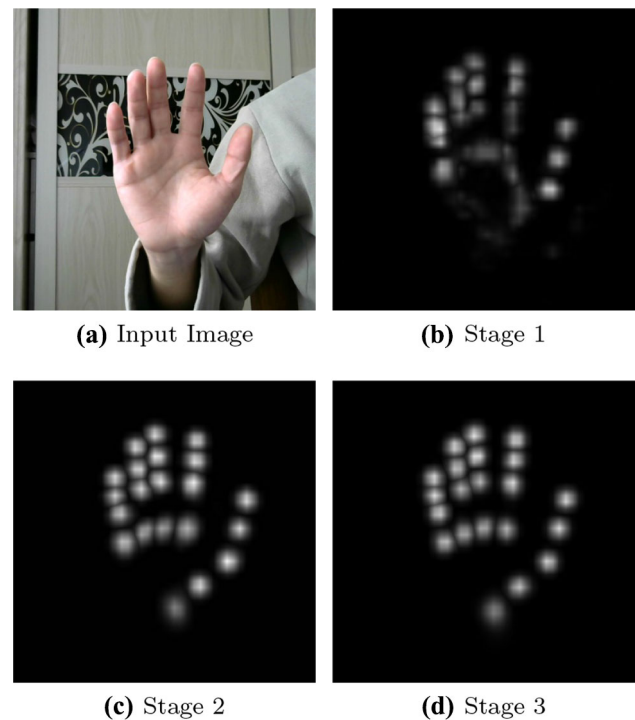

(a) Input Image **(b)** Stage 1

(c) Stage 2 **(d)** Stage 3

Fig. 3 The output of each stage

(ignoring the background heatmap) of each stage into one heatmap using the following equation:

$$H = \max_{k \in \{1,\ldots,K\}} (G^k(x,y)) \tag{32}$$

where $G^k(x,y)$ denotes the heatmap corresponding to the k-th keypoint.

The combination heatmap of each stage is shown in Fig. 3. We can see that the heatmap produced at the first stage is a little noisy and the activation values are weak. It is because the effective receptive field at this stage is small and the long-range relationships between keypoints cannot be learned well with small receptive fields. When it comes to the second stage, the receptive field becomes larger and therefore the combination heatmap is much more clear as shown in Fig. 3c. At the third stage, the receptive field is the largest and it is able to handle the long-range relationships between parts. The heatmap of Stage 3 is more clear, the response value is stronger, and the location of keypoints is more accurate compared to the heatmap of Stage 2.

We also test the performance of the trained network in challenging situations and some examples are shown in Fig. 4. The hand pose estimator works well even when the hand is in a strange pose shown in Fig. 4a. In Fig. 4b, when some joints of the hand are occluded, it can also infer the location of these keypoints accurately. When the hand is put on the arm, though they have similar skin color, the algorithm can still distinguish the hand from the arm. In
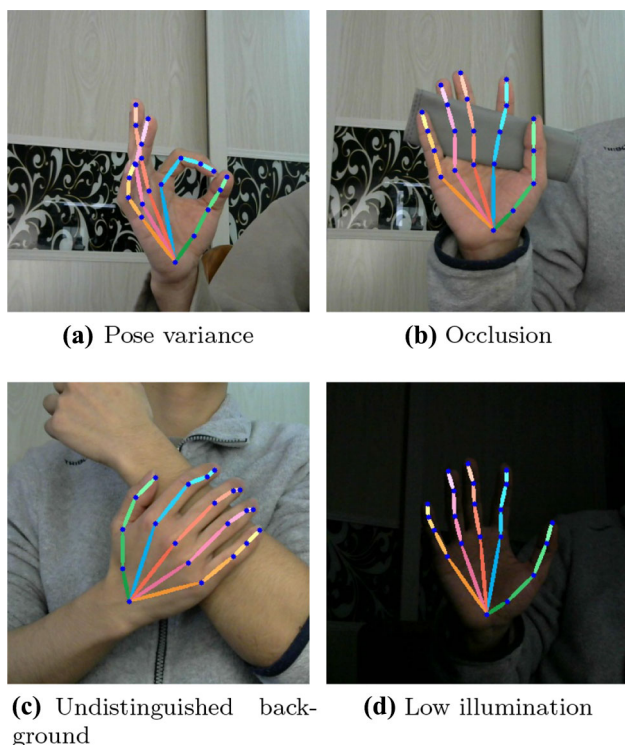
(a) Pose variance

(b) Occlusion

(c) Undistinguished background

(d) Low illumination

Fig. 4 Examples of some challenging situations

low illumination conditions such as Fig. 4d, it is still able to locate the keypoints. All the examples above demonstrate that the hand pose estimator is robust to the complex background and challenging situations.

6.2 Evaluation on the Gesture Recognition System

In order to evaluate the performance of the overall gesture recognition system in the real-world scenarios, we collect a gesture dataset consisting of 7 gesture categories and 1 nongesture category. These 7 gestures are shown in Fig. 5. We collect these images of different gestures in different indoor scenarios under varying lighting conditions. In this dataset, each gesture category contains 200 samples and the nongesture category contains 100 samples. In each gesture category, four-fifths of the samples are used as training samples and the remainder are left for evaluation. In order to evaluate the ability of the recognition system to handle

the nongesture patterns, only one-fifth of the samples are used as training samples and all the remaining samples are used for evaluation. Therefore, this dataset consists of 1140 training images and 360 testing images totally.

After the previous experiment, the hand pose estimator is obtained. To construct a complete gesture recognition system, we train an FGMM classifier described in Sect. 5 on top of the pose estimator. Note that only the gesture patterns in the training set are used for the training of FGMM, the nongesture patterns are used to determine a proper threshold value in Eq. 31. Therefore, the number of components m is set to 7 during training.

For comparison, we also train an support vector machine (SVM) classifier and a multi-layer perceptron (MLP) as the gesture classifier. We can assume that the hand pose data are not linearly separable. However, SVM with Gaussian radial basis kernel function is good at handling these nonlinear classification problems. Therefore, it is chosen as the kernel function of the SVM classifier. The MLP classifier we use is actually a two-layer fully connected neural network with ReLU activation functions. The number of neurons of the output layer is 8 including one for the nongestures. All the training samples are used to train the SVM classifier and the MLP classifier including the nongesture patterns. After training, we test these two classifiers on each category subset, respectively, and on the whole testing set. The comparison result is given in Table 1.

From Table 1, we can see that for the gesture categories the performance of FGMM is comparable to that of the SVM and MLP. However, for the nongesture patterns, the accuracy of SVM is only 65%, and MLP is only 47.5% while FGMM can achieve 95% which leads to the better performance on the whole testing set. Even when the number of nongesture training data is small, the tailored FGMM classifier can still have satisfactory performance for the nongesture patterns, which demonstrates the ability of FGMM to reject unknown categories. On the overall testing set which is collected in different indoor scenarios, the proposed gesture recognition system achieves an accuracy of 98.06%, which demonstrates the effectiveness and the robustness to complex background.
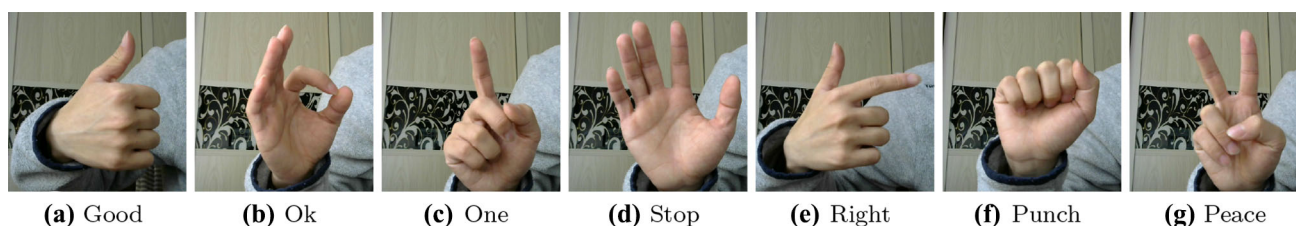

(a) Good **(b)** Ok **(c)** One **(d)** Stop **(e)** Right **(f)** Punch **(g)** Peace

Fig. 5 Gestures in our dataset

Table 1 Results on the dataset

Gesture	Accuracy (%)		
	SVM	MLP	FGMM
Good	97.5	100	100
Ok	100	100	100
One	100	100	100
Stop	100	100	95
Right	95	100	97.5
Punch	100	100	100
Peace	97.5	100	100
Nongesture	65	47.5	**95**
Total	91.11	88.33	**98.06**

Bold value indicates the better performance of the proposed method

We also conduct an experiment to evaluate the performance of these three classifiers to reject the nongestures with limited nongesture training samples. We randomly choose 50 samples from the 100 nongestures in the dataset for testing and choose different numbers of nongestures from the remaining 50 nongestures for training. The result is presented in Fig. 6. Note that only the accuracy of nongestures classification is considered in this experiment. Though there is only one nongesture training sample, the FGMM classifier can reject 66% of the nongesture patterns. However, the SVM and MLP classifiers are not able to reject the unknown categories in this limited situation. When the number of nongestures in training set is smaller than 30, FGMM has the best performance to reject the unknown gestures, while the performance of the other two classifiers is not satisfactory. This experiment reveals the ability of FGMM to reject the nongestures with limited numbers of nongesture training samples.
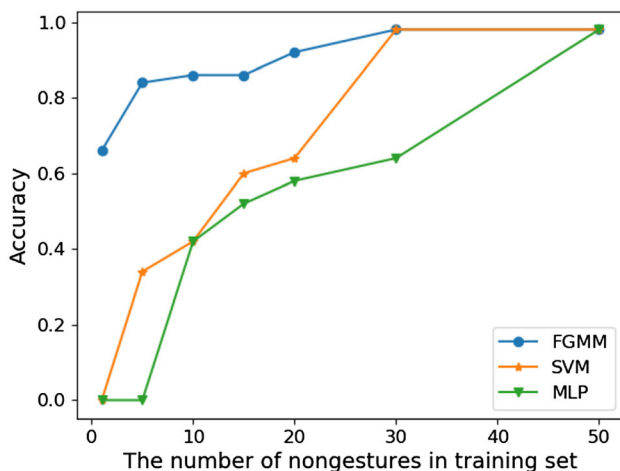

Fig. 6 Comparison results for different classifiers on training set with limited numbers of nongestures

We evaluate the computational efficiency of the overall gesture recognition system on an NVIDIA 2080 GPU, which achieves more than 30 fps. It means that it only takes 33 ms for the system to process every single frame, which meets the real-time requirements.

7 Conclusions

In this paper, a two-stage gesture recognition system is proposed to tackle the problem of complex background. Convolutional pose machine is first applied to estimate the pose of the hand, which can effectively localize hand keypoints even in complex background. After being pre-processed, these hand keypoints are then fed to a tailored FGMM classifier for gesture recognition. After modification, the FGMM classifier is able to reject the nongesture patterns and classify the gesture patterns well. Experimental results demonstrate that our algorithm is not only robust to the complex background but also satisfactory to real-time requirements.

References

1. Gao, W., Fang, G., Zhao, D., Chen, Y.: A chinese sign language recognition system based on sofm/srn/hmm. Pattern Recognit. **37**(12), 2389–2402 (2004)
2. Oyedotun, O.K., Khashman, A.: Deep learning in vision-based static hand gesture recognition. Neural Comput. Appl. **28**(12), 3941–3951 (2017)
3. Han, J., Awad, G., Sutherland, A.: Automatic skin segmentation and tracking in sign language recognition. Iet Comput. Vis. **3**(1), 24–35 (2009)
4. Chang, C.-C., Liu, C.-Y., Tai, W.-K.: Feature alignment approach for hand posture recognition based on curvature scale space. Neurocomputing **71**, 1947–1953 (2008)

5. Lai, G., Liu, Z., Zhang, Y., Chen, C.P., Xie, S., Liu, Y.: Fuzzy adaptive inverse compensation method to tracking control of uncertain nonlinear systems with generalized actuator dead zone. IEEE Trans. Fuzzy Syst. **25**(1), 191–204 (2017)

6. Liu, L., Chen, C.P., Zhou, Y., You, X.: A new weighted mean filter with a two-phase detector for removing impulse noise. Inf. Sci. **315**, 1–16 (2015)

7. Liu, Z., Wang, F., Zhang, Y., Chen, X., Chen, C.P.: Adaptive tracking control for a class of nonlinear systems with a fuzzy dead-zone input. IEEE Trans. Fuzzy Syst. **23**(1), 193–204 (2015)

8. Yin, X., Xie, M.: Estimation of the fundamental matrix from uncalibrated stereo hand images for 3d hand gesture recognition. Pattern Recognit. **36**(3), 567–584 (2003)

9. Pisharady, P.K., Vadakkepat, P., Loh, A.P.: Attention based detection and recognition of hand postures against complex backgrounds. Int. J. Comput. Vis. **101**(3), 403–419 (2013)

10. Dominio, F., Donadeo, M., Zanuttigh, P.: Combining multiple depth-based descriptors for hand gesture recognition. Pattern Recognit. Lett. **50**, 101–111 (2014)

11. Wei, S.-E., Ramakrishna, V., Kanade, T., Sheikh, Y.: Convolutional pose machines. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4724–4732 (2016)

12. Ju, Z., Liu, H.: Fuzzy gaussian mixture models. Pattern Recognit. **45**(3), 1146–1158 (2012)

13. Kovac, J., Peer, P., Solina, F.: Human skin color clustering for face detection. In: EUROCON 2003. Computer as a tool. The IEEE region 8, vol. 2, IEEE, pp. 144–148 (2003)

14. Qian, C., Sun, X., Wei, Y., Tang, X., Sun, J.: Realtime and robust hand tracking from depth. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1106–1113 (2014)

15. Van den Bergh, M., Van Gool, L.: Combining rgb and tof cameras for real-time 3d hand gesture interaction. In: Proceedings of the 2011 IEEE workshop on applications of computer vision (WACV), IEEE, pp. 66–72 (2011)

16. Jones, M.J., Rehg, J.M.: Statistical color models with application to skin detection. Int. J. Comput. Vis. **46**(1), 81–96 (2002)

17. Peng, X., Wang, L., Cai, Z., Qiao, Y.: Action and gesture temporal spotting with super vector representation. In: Workshop at the European conference on computer vision, Springer, pp. 518–527 (2014)

18. Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., Blake, A.: Real-time human pose recognition in parts from single depth images. In: proceedings of the 2011 IEEE conference on computer vision and pattern recognition (CVPR), IEEE, pp. 1297–1304 (2011)

19. Kang, B., Tan, K.-H., Jiang, N., Tai, H.-S., Treffer, D., Nguyen, T.: Hand segmentation for hand-object interaction from depth map. In: Proceedings of the 2017 IEEE global conference on signal and information processing (GlobalSIP), IEEE, pp. 259–263 (2017)

20. Chen, C.P., Xie, S.: Freehand drawing system using a fuzzy logic concept. Comput. Aided Des. **28**(2), 77–89 (1996)

21. Zhou, J., Chen, L., Chen, C.P., Zhang, Y., Li, H.-X.: Fuzzy clustering with the entropy of attribute weights. Neurocomputing **198**, 125–134 (2016)

22. Priyal, S.P., Bora, P.K.: A robust static hand gesture recognition system using geometry based normalizations and krawtchouk moments. Pattern Recognit. **46**(8), 2202–2219 (2013)

23. Lienhart, R., Maydt, J.: An extended set of haar-like features for rapid object detection. In: Proceedings of the 2002 international conference on image processing. 2002. vol. 1, IEEE, pp. I–I (2002)

24. Dardas, N.H., Georganas, N.D.: Real-time hand gesture detection and recognition using bag-of-features and support vector machine

techniques. IEEE Trans. Instrum. Measur. **60**(11), 3592–3607 (2011)

25. Krizhevsky, A., Sutskever, I., Hinton, G. E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp. 1097–1105 (2012)

26. Liang, C., Song, Y., Zhang, Y.: Hand gesture recognition using view projection from point cloud. In: Proceedings of the 2016 IEEE international conference on image processing (ICIP), IEEE, pp. 4413–4417 (2016)

27. Ramakrishna, V., Munoz, D., Hebert, M., Bagnell, J. A., Sheikh, Y.: Pose machines: articulated pose estimation via inference machines. In: European conference on computer vision, Springer, pp. 33–47 (2014)

28. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3431–3440 (2015)

29. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409. 1556

30. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics, pp. 249–256 (2010)

31. Ballard, D.H.: Modular learning in neural networks, pp. 279–284. AAAI, Menlo Park (1987)

32. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Trans. Neural Netw. **5**(2), 157–166 (1994)

33. Rumelhart, D.E., Hinton, G.E., Williams, R.J., et al.: Learning representations by back-propagating errors. Cogn. Model. **5**(3), 1 (1988)

34. Zivkovic, Z.: Improved adaptive gaussian mixture model for background subtraction. In: Proceedings of the 17th international conference on pattern recognition, 2004. ICPR 2004. vol. 2, IEEE, pp. 28–31 (2004)

35. Reynolds, D.A., Quatieri, T.F., Dunn, R.B.: Speaker verification using adapted gaussian mixture models. Digit. Signal Process. **10**(1–3), 19–41 (2000)

36. Reynolds, D.A., Rose, R.C., et al.: Robust text-independent speaker identification using gaussian mixture speaker models. IEEE Trans. Speech Audio Process. **3**(1), 72–83 (1995)

37. Gao, Y., Wang, D., Pan, J., Wang, Z., Chen, B.: A novel fuzzy c-means clustering algorithm using adaptive norm. Int. J. Fuzzy Syst. **21**(8), 2632–2649 (2019)

38. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J R Stat Soc Ser B (Methodological) **39**, 1–38 (1977)

39. Krishna, K., Murty, M.N.: Genetic k-means algorithm. IEEE Trans Syst Man Cybern Part B (Cybernetics) **29**(3), 433–439 (1999)

40. Ju, Z., Liu, H.: Applying fuzzy em algorithm with a fast convergence to GMMS. In: Proceedings of the 2010 IEEE international conference on fuzzy systems (FUZZ), IEEE, pp. 1–6 (2010)

41. Bezdek, J.C., Ehrlich, R., Full, W.: FCM: the fuzzy c-means clustering algorithm. Comput. Geosci. **10**(2–3), 191–203 (1984)

42. Zhao, X., Li, Y., Zhao, Q.: A fuzzy clustering approach for complex color image segmentation based on gaussian model with interactions between color planes and mixture gaussian model. Int. J. Fuzzy Syst. **20**(1), 309–317 (2018)

43. Lin, H., Zhang, T., Chen, Z., Song, H., Yang, C.: Adaptive fuzzy Gaussian mixture models for shape approximation in robot grasping. Int. J. Fuzzy Syst. **21**(4), 1026–1037 (2019)

44. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M.,Ghemawat, S., Irving, G., Isard, M., et al.: Tensorflow:

a system for large-scale machine learning. In: 12th {USENIX} symposium on operating systems design and implementation ({ OSDI } 16), pp. 265–283 (2016)

45. Zimmermann, C., Brox, T.: Learning to estimate 3d hand pose from single RGB images. In: Proceedings of the IEEE international conference on computer vision, pp. 4903–4911 (2017)

Tong Zhang received the BS degree in software engineering from Sun Yat-sen University, Guangzhou, China, in 2009, the MS degree in applied mathematics from the University of Macau, Macau, China, in 2011, and the PhD degree in software engineering from the University of Macau, Macau, China, in 2016. He currently is an associate professor with the School of Computer Science and Engineering, South China University of Technology, China. His research interests include affective computing, evolutionary computation, neural network, and other machine learning techniques and their applications. He has been working in publication matters for many IEEE conferences. He is a member of the IEEE.

Huifeng Lin received the B. Eng. degree in automation from the South China University of Technology, Guangzhou, China, in 2018, and is currently pursuing the M.S. degree in the South China University of Technology, Guangzhou, China. His research interests include human-robot interaction, machine learning and image processing.

Zhaojie Ju received the B.S. in automatic control and the M.S. in intelligent robotics both from Huazhong University of Science and Technology, China, in 2005 and 2007 respectively, and the Ph.D. degree in intelligent robotics at the University of Portsmouth, UK, in 2010. He is currently a Senior Lecturer in the School of Computing, University of Portsmouth. He previously held research appointments at University College London and University of Portsmouth, UK. His research interests include machine intelligence, pattern recognition, and their applications on human motion analysis, human-robot interaction and collaboration, and robot skill learning.

Chenguang Yang received the Ph.D. degree in control engineering from the National University of Singapore, Singapore, in 2010 and performed postdoctoral research in human robotics at Imperial College London, London, UK from 2009 to 2010. He has been awarded EU Marie Curie International Incoming Fellowship, UK EPSRC UKRI Innovation Fellowship, and the Best Paper Award of the IEEE Transactions on Robotics as well as over ten conference Best Paper Awards. His research interest lies in human robot interaction and intelligent system design.