IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Traitement du Signal
Vol. 37, No. 5, October, 2020, pp. 873-880
Journal homepage: http://iieta.org/journals/ts

Gesture Feature Extraction and Recognition Based on Image Processing

Yingnan Wang[1*], Yueming Yang[2], Peiye Zhang[1]

[1] Jilin Agricultural University, Changchun 130000, China
[2] Changchun Fawsn Group Co., Ltd., Changchun 130000, China

Corresponding Author Email: yingnanw@jlau.edu.cn

https://doi.org/10.18280/ts.370521

Received: 21 April 2020
Accepted: 3 August 2020

Keywords:
image processing, gesture feature extraction, gesture recognition, convolutional neural network (CNN)

ABSTRACT

Gesture recognition has become increasingly popular, in response to the growing demand for intelligent and personalized human-computer interaction (HCI) and human-to-human interaction. However, gesture recognition raises a high requirement on the background color of the gesture image, and faces difficulty in extracting multiple gesture features. To solve these problems, this paper presents a novel approach for gesture feature extraction and recognition based on image processing. Firstly, the workflow of the proposed gesture recognition method was given, and a series of preprocessing was performed on the original gesture image, prior to formal extraction and recognition. Next, the authors detailed the extraction of features from gesture boundaries and fingertips. Finally, a convolutional neural network (CNN) was constructed for gesture recognition, and a gesture recognition model was developed based on residual network. The proposed approach was proved to be valid through experiments. The research results provide a reference for the application of CNN in the recognition of various postures or shapes.

1. INTRODUCTION

With the proliferation of computer vision and intelligent terminals, there is a growing demand for intelligent and personalized human-computer interaction (HCI) [1-4]. However, the traditional contact HCI devices, with numerous input rules, have a poor interaction efficiency. Compared with HCI media like voice, face, and fingerprint, gesture is the simplest, fastest, and most common interaction medium for human beings. Gesture recognition has become an important research direction in the field of intelligent HCI [5, 6]. Apart from HCI, gesture recognition also supports the human-to-human interaction between deaf-mute and healthy people, enabling the transmission of information in silent situations. This gives practical significance to the research and development (R&D) of gesture recognition.

By the acquisition method, gesture recognition is based on either external sensors or computer vision [7-9]. The external sensor-based technique acquires gesture features by processing the gesture information collected by wearable devices like data gloves, electromyographic sensors, and linear bending sensors. But this technique incurs a high device cost, and lacks diverse sensing functions. As a result, the computer vision-based technique has attracted more attention from foreign researchers [10-13]. Saad et al. [14] segmented the target image of the hand region by background subtraction, and tracked hand gestures using the prior knowledge of the color space of the hand skin. Roma et al. [15] segmented the gesture region based on the entropy information of the hand target image, and used the center-of-gravity contour method to determine the boundary of the hand, thereby increasing the detection rate and recognition rate of gestures to over 85%. Smith et al. [16] extracted the geometric shape descriptors of the skeleton and joints in hand images, and encoded effective descriptors based on multivariate Gaussian function and the generated Fisher vector. Following the principle of image processing, Venkatnarayan and Shahzad [17] extracted and recognized gesture features with the threshold model of hidden Markov model and the support vector machine (SVM) classifier, and correctly recognized more than 91% of 10 dynamic and static gestures.

The research on gesture recognition in China started late, but has been developing rapidly [18-20]. Negin ey al. [21] introduced a multifunctional perceptron for sign language recognition and synthesis; the main functions include real-time recognition of large vocabulary of sign language, a synthesis system for sign language in personal computer (PC) and tablet, and face image monitoring and recognition. Shahzad and Zhang [22] collected gesture information with two pairs of data gloves and three position sensors, and recognized more than 93% of 280 gesture models, using Gaussian mixture model and hidden Markov model. Based on the complete feature set of gestures, Koh et al. [23] fused the color space information of hand skin with the features of hand gestures, and put forward a method for extracting spatiotemporal dynamic gesture features. Yasen and Jusoh [24] adopted the optical flow method to solve the nonlinear transform of static gestures between frames in gesture video, and accurately recognized and classified gestures through orthogonal reconstruction of the original gesture images based on Zernike polynomials.

Despite its popularity, gesture recognition faces multiple difficulties, such as the differences in hand features, the constant changes of gestures, and the complexity of external environments. Moreover, gesture recognition raises a high requirement on the background color of the gesture image, and needs to go through a complex process to extract multiple gesture features. To solve these problems, this paper presents

a novel approach for gesture feature extraction and recognition based on image processing.

The remainder of this paper is organized as follows: Section 2 presents the workflow of the proposed gesture recognition method, and details the preprocessing of the original gesture image, prior to formal extraction and recognition; Section 3 explains the extraction of features from gesture boundaries and fingertips from the gesture image; Section 4 constructs a convolutional neural network (CNN) for gesture recognition, and built up a gesture recognition model was developed based on residual network; Section 5 verifies the effectiveness of our approach through experiments; Section 6 puts forward the conclusions.

2. PREPROCESSING OF GESTURE IMAGE

Figure 1 shows the workflow of the proposed gesture recognition method based on feature extraction from gesture image. For the original noise-containing gesture image, morphological processing like denoising, binarization, expansion, erosion, as well as open and close operations need to be conducted before extracting and recognizing gesture features, aiming to obtain salient image features and achieve good effect of image recognition.

Based on a multi-layer CNN, this paper constructs a gesture recognition model. To prevent the noise accumulation in the iterative training and learning of the model, the Gaussian white noise was filtered out by a Gaussian filter:

$$G(a,b) = e^{-(a^2+b^2)/2\sigma^2} \tag{1}$$

where, $G(a, b)$ is the two-dimensional (2D) Gaussian function value corresponding to the coordinates of the pixel (a, b) in the gesture image; σ is the standard deviation. In addition to the Gaussian filter, a median filter was adopted to eliminate isolated noise points and smooth the edge burrs, thereby reducing the impact of finding the distinguishing line of the wrist joint, and realizing the counting and sorting of the neighboring pixels of the current pixel.

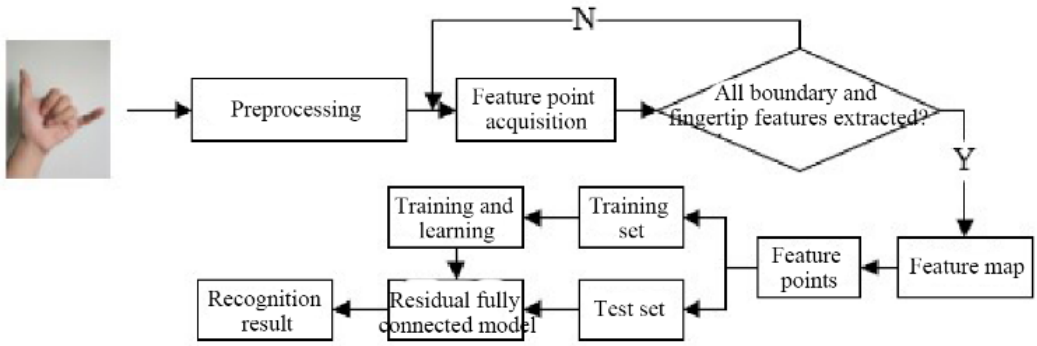


Figure 1. The workflow of gesture recognition based on feature extraction from gesture image

After denoising, binarization was performed to better distinguish the hand area in the gesture image from the image background. Here, the skin color of the hand is segmented in RGB and YCbCr color spaces. In the RGB color space, the global fixed threshold binarization method was adopted:

$$GV_B(a,b) = \begin{cases} 255, GV(a,b) > \varepsilon_1 \\ 0\ , GV(a,b) \le \varepsilon_1 \end{cases} \tag{2}$$

where, $GV(a,b)$ and $GV_B(a,b)$ are the gray value of the original gesture image, and the gray value obtained by comparing the original gesture image pixels with the fixed threshold ε_1, respectively.

In the YCbCr color space, the maximum inter-class variance was adopted for binarization. By this method, the inter-class variance was taken as the scale between the hand foreground and image background. The inter-class variance is positively correlated with the probability of distinguishing the hand from the background.

For a gesture image $GI(a,b)$ of size $W*H$, let λ_l and N_l be the mean gray value and number of all pixels whose gray value is lower than threshold ε_2 between the hand foreground and the image background, respectively; λ_h and N_h be the mean gray value and number of all pixels whose gray value is higher than threshold ε_2, respectively. Then, the proportions of foreground pixels and background pixels to the total number of pixels in the image can be expressed as:

$$\begin{cases} P_l = \dfrac{N_l}{W \times H} \\ P_h = \dfrac{N_h}{M \times N} \end{cases} \tag{3}$$

The mean gray value of the entire gesture image can be calculated by:

$$\lambda = P_l \cdot \lambda_l + P_h \cdot \lambda_h \tag{4}$$

The inter-class variance can be expressed by:

$$IV = (P_l \cdot \lambda_l - \lambda)^2 + (P_h \cdot \lambda_h - \lambda)^2 \tag{5}$$

The segmentation threshold ε_2 falls in [1, 255]. The threshold value should ensure that the inter-class variance is maximized.

The binarized gesture image $GI*$ was subject to morphological removal and filling of isolated interference points, burrs, holes, and gaps. Suppose S is a structural element in the shape of a square, a cross or an ellipse. Then, the morphological elimination can be completed by:

$$GI^* \otimes S = \{a,b \mid (S)_{ab} \subseteq GI^*\} \tag{6}$$

The morphological elimination (6) is an erosion operation, which removes all the boundary points and isolated points from the segmented gesture area. To corrode the gesture image *GI** by the structural element S, the center point of *S* was first translated to the position of pixel (*a*, *b*). If the intersection of *GI** and *S* at (*a*, *b*) is not S, then (*a*, *b*) will be assigned the value 0; otherwise, (*a*, *b*) will be assigned the value 255. The morphological filling can be completed by:

$$GI^* \oplus S = \{a,b \mid (S)_{ab} \cap GI^* \neq \varnothing\} \tag{7}$$

The morphological filling (7) is an expansion operation, which merges the segmented gesture area with the adjacent background points. Similarly, to expand the gesture image *GI** by the structural element S, the center point of S was first translated to the position of pixel (*a*, *b*). If there is no intersection of *GI** and S at (*a*, *b*), then (*a*, *b*) will be assigned the value 0; otherwise, (*a*, *b*) will be assigned the value 255.

Starting from the actual situation of the binarized gesture image *GI**, morphological open and close operations determines the sequence of elimination and filling operations, such that the segmented area from the preprocessed gesture image is as large as that from the original image. Figures 2 displays the effect of graying, binarization, and open and close operations on the original gesture image.

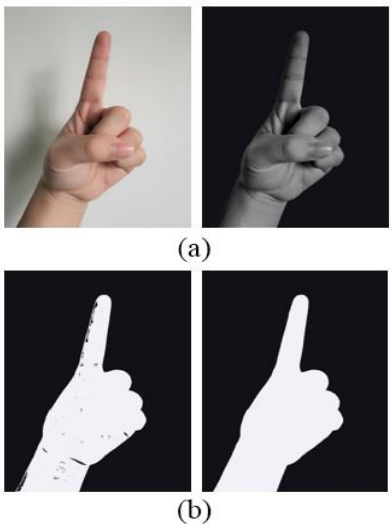


Figure 2. The effect of preprocessing on the original gesture image

3. EXTRACTION OF GESTURE FEATURES

In this paper, the gesture boundary features are extracted by the Fourier descriptor, which is an excellent tool for describing the contour shape of the target. Figure 3 defines the 10 target gestures. For any point (a_0, b_0) on the gesture boundary in the original image, there must exist a point (a_{NBO}, b_{NBO}) that equals (a_0, b_0), where N_{BO} is the number of pixels on gesture boundary. The complex expression of the boundary pixels is as follows:

$$s_i = a_i + jb_i \tag{8}$$

One-dimensional (1D) discrete Fourier transform can be performed on formula (8) to obtain the Fourier descriptor of the gesture boundary:

$$z_d = \frac{1}{N_{BO}} \sum_{i=1}^{N_{BO}-1} s_i e^{-j2\pi di/N_{BO}} \tag{9}$$

The Fourier descriptor z_d can convert 2D image processing problems into 1D problems, which greatly reduces the computing load. The accuracy of the Fourier descriptor during the detection can be obtained through the inverse transform:

$$s_i = a_i + jb_i = \sum_{i=1}^{N_{BO}-1} z_d e^{j2\pi di/N_{BO}} \tag{10}$$

As shown in formulas (9) and (10), the Fourier descriptor is affected by the rotation, translation, or expansion of the gesture boundary. This effect can be reduced or eliminated by calculating the ratio of the modes:

$$\varphi_d = \frac{\|z_d\|}{\|z_1\|} \tag{11}$$

To generalize the boundary shape of the gesture area, it is redundant to use high-frequency Fourier descriptors, which are good at describing the details. Only the first n low-frequency Fourier descriptors, except φ_0 and φ_1, need to be selected. Then, the gesture boundary can be redrawn through inverse transform of the n descriptors by formula (10). Figure 4 presents the effect of extracting the gesture boundaries with different numbers of Fourier descriptors.

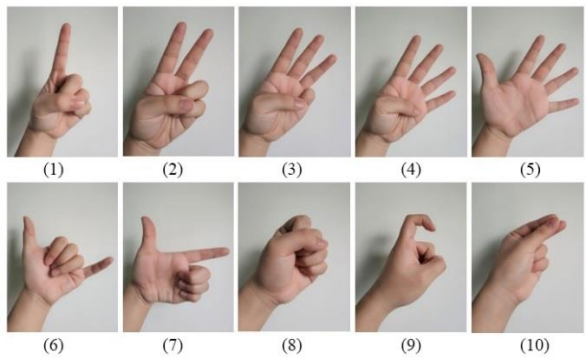


Figure 3. The definitions of the 10 target gestures

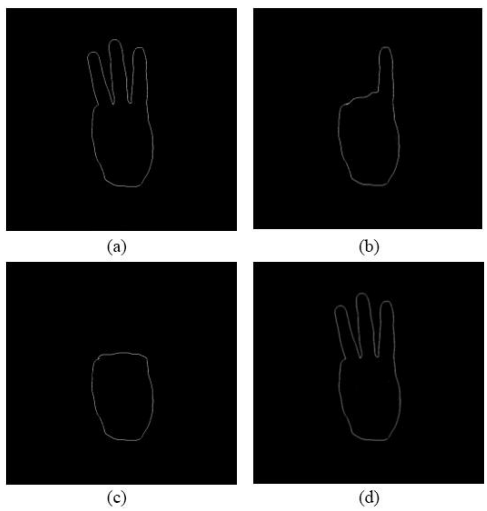


Figure 4. The effects of extracting the gesture boundaries

As shown in Figure 4, obvious bending features can be observed in the fingertips, the depression between fingers, and the joint between hand and wrist in the segmented images. The local bending features can be derived from the curvature of each boundary point of the gesture region. Let p_i be a boundary point on gesture region. Based on the preset traversal step length, the previous and subsequent points of p_i, namely, p_{i-1} and p_{i+1}, can be obtained. Let N_B be the number of all points, and $P=\{p_0, p_1, p_2, \ldots, p_{i-1}, p_i=(a_i,b_i), p_{i+1}, \ldots, p_{NB-1}\}$ be the set of all points. Then, the curvature of point p_i is the cosine value of the angle α between vectors p_ip_{i+1} and $p_{i-1}p_i$:

$$\cos\alpha_i = \frac{\overrightarrow{p_i p_{i+1}} \times \overrightarrow{p_{i-1} p_i}}{\left|p_i p_{i+1}\right| \times \left|p_{i-1} p_i\right|} \tag{12}$$

By observing different gestures, it can be found that fingertip features lay an important basis for gesture recognition. To quickly eliminate other misjudgment points that also have curved features, this paper further filters the circular features of all points with local bending features. Let N_T be the total number of target points. Then, a circle containing the target point $p_i=(a_i,b_i)$ was drawn with $O(x,y)$ as the center and R as the radius:

$$(a-x)^2 + (b-y)^2 = r^2 \tag{13}$$

The target point $p_i=(a_i,b_i)$ can be mapped to the circular feature space (x, y, R) by:

$$\begin{cases} a_i^* = a_i - R\cos\alpha_i \\ b_i^* = b_i - R\sin\alpha_i \end{cases} \tag{14}$$

where, α falls in the range of [0, 90°); R cannot exceed half the width of the finger joint. Then, it was judged whether all the neighboring points of the target point fall on a circle, and the circle with the most neighboring points was chosen.

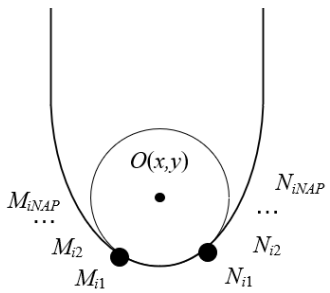


Figure 5. The intersection of circle and gesture boundary

Figure 5 illustrates the intersection of circle and gesture boundary. The two intersection points between the circle containing point p_i and gesture boundary, namely, M_i and N_i, can be expressed by:

$$\begin{cases} M_i = \left(M_{i1}, M_{i2}, \ldots, M_{iN_{AP}}\right) \\ N_i = \left(N_{i1}, N_{i2}, \ldots, N_{iN_{AP}}\right) \end{cases} \tag{15}$$

where, N_{AP} is the total number of pixels on the circle containing point p_i. The mean error of any point K_i on the left and right of M_i and N_i can be calculated by:

$$AE_i = \frac{1}{2N_{AP}} \sum_{q=1}^{2N_{AP}} ED_{iq} \tag{16}$$

The standard error SE_i can be calculated by:

$$SE_i = \sqrt{\frac{1}{2N_{AP}} \sum_{q=1}^{2N_{AP}} \left(ED_{iq} - AE_i\right)^2} \tag{17}$$

The Euclidean distance ED_{iq} between the boundary point of the gesture area and the point on the circle can be obtained by:

$$ED_{iq} = \begin{cases} \sqrt{\left(a_{M_{is}} - a_{K_{iq}}\right)^2 + \left(b_{M_{is}} - b_{K_{iq}}\right)} & 0 \le q \le N_{AP} \\ \sqrt{\left(a_{N_{ir}} - a_{K_{iq}}\right)^2 + \left(b_{N_{ir}} - b_{K_{iq}}\right)} & N_{AP} \le q \le 2N_{AP} \end{cases} \tag{18}$$

where, (a_{Mis},b_{Mis}) and (a_{Nir},b_{Nir}) are the coordinates of the s-th pixel to the left of M_i and r-th pixel to the right of N_i, respectively; (a_{Kiq},b_{Kiq}) is the coordinates of the q-th pixel to the left of M_i on the circle containing p_i. s equals q and r equals q-$2N_{AP}$. The mean error of a pixel can be calculated by:

$$ME_i = \frac{1}{2}\sqrt{AE_i^2 + SE_i^2} \tag{19}$$

The mean error of a circle and its corresponding gesture boundary can be calculated by:

$$ME_{av} = \frac{1}{N_T} \sum_{i=1}^{N_T} ME_i \tag{20}$$

If ME_i is smaller than ME_{av}, the circle is the circle corresponding to the fingertip, i.e. point p_i is the fingertip in the gesture area. Figure 6 shows the feature points recognized in the gesture area of the original image.

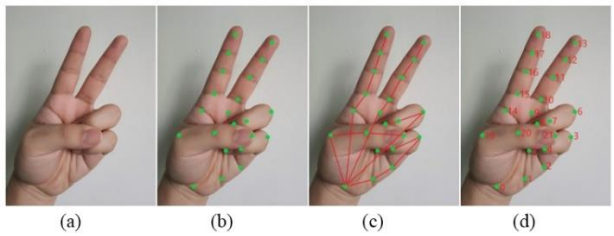


Figure 6. The feature points recognized in the gesture are

4. CONSTRUCTION OF GESTURE RECOGNITION NETWORK

Figure 7 presents the structure of the constructed CNN. The deep neural network faces reduced generalizability and network degradation during the gesture recognition. To solve these problems, the residual idea was incorporated to the constructed CNN. As shown in Figure 8, x is the input of the residual block, and $F(x)$ is the residual, i.e., the output after the linear transform and activation of the block in the first layer. $F(x)$ and x are superimposed and activated before being outputted. The path to superimpose the output function of the block with x before the activation of the output value of the second layer is called a shortcut connection.

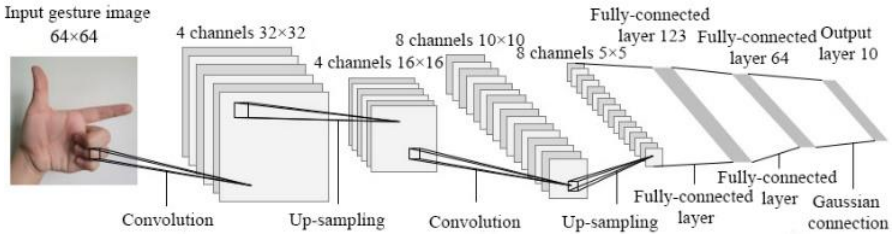


Figure 7. The constructed CNN

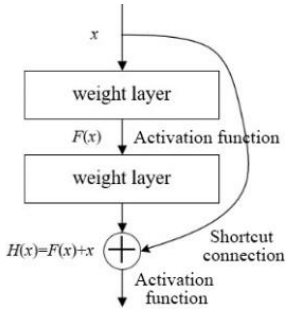


Figure 8. The structure of residual block

To obtain a sufficiently small residual $F(x)$, the loss function of the residual block of the k-th layer of the residual network can be calculated and derived by:

$$y_k = F\left(x_k, \omega_k, \varepsilon_k\right) \tag{21}$$

$$\frac{\partial y_k}{\partial x_k} = \frac{\partial F\left(x_k, \omega_k, \varepsilon_k\right)}{\partial x_k} \tag{22}$$

where, x_k, ω_k, and ε_k are the input, weight, and bias of the k-th residual block. The loss function of the network can be calculated by:

$$\frac{\partial y}{\partial x_k} = \frac{\partial F_m\left(x_{km}, \omega_{km}, \varepsilon_{km}\right)}{\partial x_k} \cdots\cdots \frac{\partial F_2\left(x_{k2}, \omega_{k2}, \varepsilon_{k2}\right)}{\partial x_k} \tag{23}$$

After adding the residual block, the partial derivative can be calculated by:

$$\frac{\partial y_k}{\partial x_k} = \frac{\partial x_k + \partial F_m\left(x_k, \omega_k, \varepsilon_k\right)}{\partial x_k} = 1 + \frac{\partial F_m\left(x_k, \omega_k, \varepsilon_k\right)}{\partial x_k} \tag{24}$$

Gesture images can also be processed following the residual idea. Let x_l be the compressed sampling value of a given gesture image. Then, the image can be preliminarily reconstructed by:

$$y_l^* = F^h\left(x_l, MAP^h\right) \tag{25}$$

where, $F^h(*)$ is the equivalent function of the mapping network; MAP^h is the linear mapping matrix constantly updated during network training. Let $F^g(x^*_l, MAP^g)$ be the residual network composed of residual blocks. The residual can be generated by:

$$r_l^* = F^g\left(x_l^*, MAP^g\right) \tag{26}$$

Taking x_l as the input, the preliminary reconstructed image y^*_l was obtained, and merged with r^*_l into y'_l:

$$y_l' = F^h\left(x_l, MAP^h\right) + F^g\left[F^h\left(x_l, MAP^h\right), MAP^g\right] \tag{27}$$

Figure 9 shows the workflow of the proposed gesture recognition method based on residual network. The convolution kernel and bias of the network were updated by the backpropagation algorithm. Let $Loss_t$ be the loss function value of the forward propagation of the convolution operation.

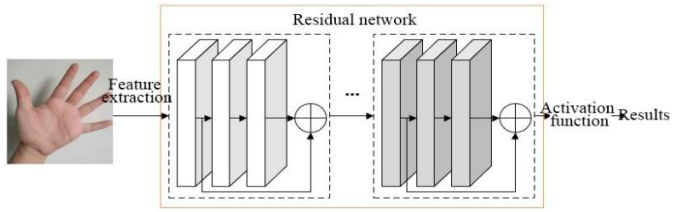


Figure 9. The workflow of gesture recognition based on residual network

Then, the partial derivative of convolution kernel relative to the parameters in column v and row c in the gesture image can be obtained by:

$$\frac{\partial Loss_t}{\partial \xi_{cv}} = \sum_i^W \sum_j^H \left(\frac{\partial Loss_t}{\partial x_{ij}} \frac{\partial x_{ij}}{\partial U_{ij}^w} \frac{\partial U_{ij}}{\partial \xi_{pq}} \right) \tag{28}$$

where, U^w_{ij} satisfies:

$$U_{ij} = \sum_{c=1}^W \sum_{v=1}^H x_{i+c-1, j+v-1} \times \xi_{cv} + \varepsilon \tag{29}$$

Formula (28) can be simplified as:

$$\frac{\partial Loss_t}{\partial \xi_{cv}} = \sum_i^W \sum_j^H \left(error_{ij} \times x_{i+c-1, j+v-1} \right) \tag{30}$$

The error of convolution kernel relative to the parameters in column j and row c in the feature map on the w-1-th layer can be calculated by:

$$error_{ij} = \sum_i^W \sum_j^H \frac{\partial Loss_t}{\partial x_{ij}} \frac{\partial x_{ij}}{\partial U_{ij}} \tag{31}$$

The partial derivative of the loss function relative to the convolution kernel can be obtained the convolution operation:

$$\frac{\partial Loss_t}{\partial \xi} = x * error \tag{32}$$

The error of the convolution kernel can be recursed by:

$$error_{ij} = \sum_{i}^{W} \left(error * rot180(\xi) f(U) \right) \tag{33}$$

where, $rot(*)$ is the fixed degrees of clockwise rotation for the convolution kernel. The above analysis shows that the network error and parameter gradient can be calculated layer by layer. Let η be the learning rate. The parameter update formula can be established as:

$$\xi - \eta(x * error) \to \xi \tag{34}$$

To prevent over-fitting, regularization was applied to attenuate the weight during the backpropagation. The regularization L1, which calculates the sum of the absolute value of all weights, and regularization L2, which calculates the square root of the quadratic sum of all weights, both obey the prior probability. Let γ be the regularization coefficient. Then, L1 must satisfy the Laplace prior probability:

$$L_1 = Loss_0 + \frac{\gamma}{N} \sum_{\varpi} |\varpi| \tag{35}$$

L2 must satisfy Gaussian prior probability:

$$L = Loss_0 + \frac{\gamma}{2N} \sum_{\varpi} \varpi^2 \tag{36}$$

Batch normalization can alleviate the diffusion of gradients that may occur in the model training, and speed up the convergence of the loss function. Let $B=\{x_1,x_2,\ldots,x_N\}$ be the input of batch processing; α and β be the scale factor and translation factor to be learned. First, the mean of the data for batch processing can be calculated:

$$\frac{1}{N} \sum_{i=1}^{N} x_i \to \mu_B \tag{37}$$

Then, the variance can be calculated:

$$\frac{1}{N} \sum_{i=1}^{N} (x_i - \mu_B)^2 \to \sigma_B^2 \tag{38}$$

After that, the normalization can be performed:

$$\frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \phi}} \to \hat{x}_i \tag{39}$$

Finally, scale shift can be implemented:

$$BN_{\alpha,\beta}(x_i) \equiv \alpha \hat{x}_i + \beta \to Y_i \tag{40}$$

5. Experiments and Results Analysis

The trend of the loss curves needs to be judged to adjust the constructed recognition model in time during the training. In our experiments, the total loss of our model on the target images was computed after each iteration. Figure 10 records the loss curves of gesture recognition by our model. It can be seen that over-fitting did not occur after 10,000 iterations; the three loss curves, namely, mini-batch loss, training loss, and test loss, plunged and tended to be stable, with the growing number of iterations. The training loss curve dropped steeper than the other two curves. Despite some fluctuations, the mini-batch loss curve was not much different from the test loss curve. The three loss curves stabilized since the 6,000-th iteration, a sign of the end of model training.

To verify its effectiveness, the proposed fingertip recognition algorithm was compared with popular methods through fingertip detection experiments, such as skin color segmentation method, histograms of oriented gradients (HOG) feature extraction method, and local binary pattern (LBP) feature extraction method. The comparison results are shown in Figure 11, where the x-axis represents the number of classes of recognized gestures, and the y-axis represents accuracy and time consumption, respectively, in the two sub-graphs. It can be seen that our algorithm recognized fingertips more accurately than the other methods, while consuming relatively short time. This verifies the accuracy and efficiency of our algorithm.

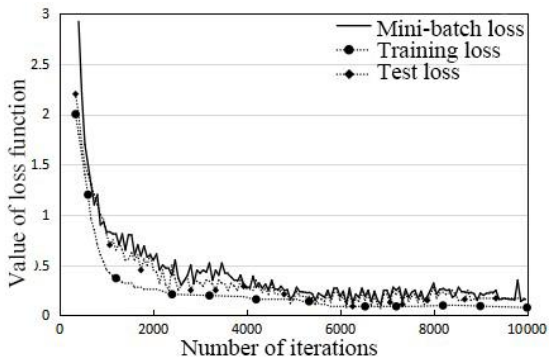
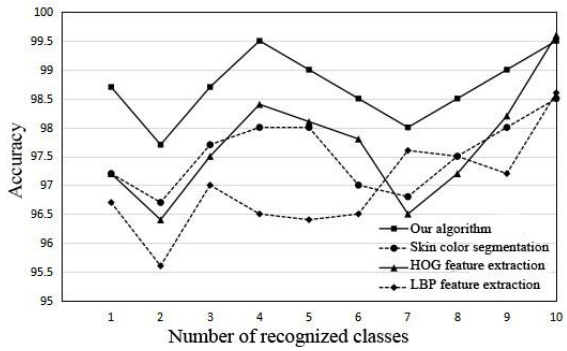
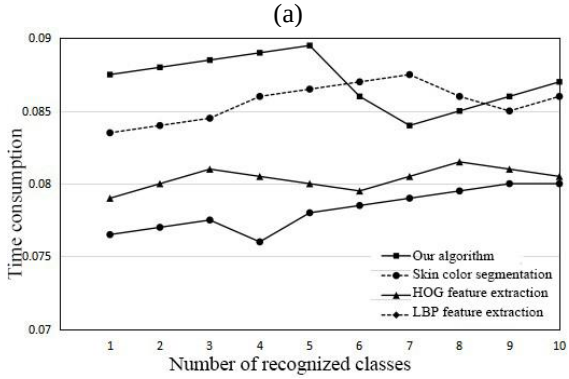


Figure 10. The loss curves of gesture recognition



(a)



(b)

Figure 11. The accuracy and time consumption of different fingertip recognition algorithms

Next, 10 types of gestures were expanded into 4,380 gesture images through rotation, translation, distance adjustment, or focal length adjustment. The accuracy of our algorithm on the 10 types of gestures is displayed in Table 1 and Figure 12. Obviously, our algorithm reduced the recognition error rate to below 2%.

As shown in Figure 12, the training and test accuracies increased rapidly before the 1,600-th iteration, and remained in sync with the falling loss curves in Figure 10. After the 1,600-th iteration, the two curves exhibited different trends. It can be seen that the training accuracy curve fluctuated less significantly than the test accuracy curve, owing to the relatively large size of the training set. Both curves were stabilized at the 5,000-th iteration, similar to the situation of the loss curves.

Table 1. The recognition accuracy of 10 types of gestures

Type	Number of gesture	Number of misjudgment	Rate of recognition error
1	450	3	1.7%
2	400	6	1.5%
3	430	5	1.16%
4	450	5	1.11%
5	450	7	1.56%
6	440	5	1.11%
7	430	6	1.4%
8	430	7	1.6%
9	450	6	1.3%
10	450	6	1.3%

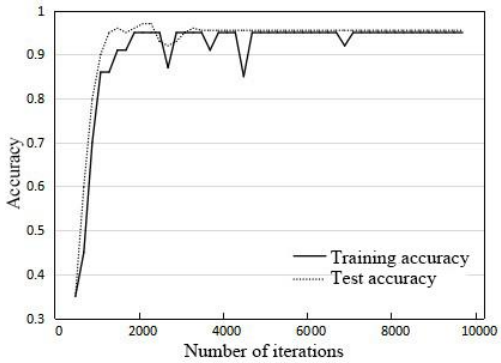


Figure 12. The accuracy curve of gesture recognition

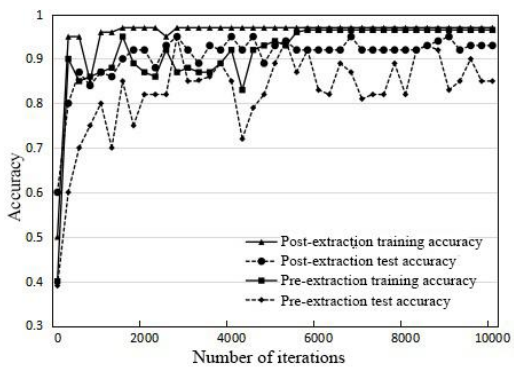


Figure 13. The accuracy with or without fingertip feature recognition

To verify the promoting effect of the fingertip recognition algorithm on the gesture recognition accuracy, the gesture recognition accuracy with or without fingertip feature recognition was summarized. As shown in Figure 13, the training accuracy of the model stabilized at 92% before the fusion of fingertip features, and above 97% after the fusion of such features; the test accuracy curve oscillated relatively significantly, but the value stabilized at around 93% after the fusion of fingertip features. These results fully demonstrate the effectiveness of our algorithm.

6. CONCLUSIONS

This paper proposes a novel method of gesture feature extraction and recognition based on image processing. Firstly, the workflow of the designed gesture recognition method was explained, and the original gesture image was preprocessed by denoising, binarization, expansion, erosion, and open and close operations. Then, the authors introduced the method to extract the features of gesture boundaries and fingertips. Finally, a CNN structure was designed for gesture recognition, and the process of gesture recognition was optimized based on residual network. Experimental results show that our algorithm recognized fingertips more accurately than the other methods, while consuming relatively short time. The fusion of fingertip features improved the training accuracy of the model by 3% to above 97%, and the test accuracy of the model by 8% to around 93%. The results fully demonstrate the effectiveness of the fingertip feature recognition algorithm.

ACKNOWLEDGMENT

This work is supported by Science and Technology Development Project of Jilin Province (Grant No.: 20190304128YY).

REFERENCES

[1] Alavi, S., Arsenault, D., Whitehead, A. (2016). Quaternion-based gesture recognition using wireless wearable motion capture sensors. Sensors, 16(5): 605. https://doi.org/10.3390/s16050605

[2] Halima, N.B. (2016). A robust method for hand gesture recognition using support vector machine. International Journal of Multimedia and Ubiquitous Engineering, 11(11): 365-374. https://doi.org/10.14257/ijmue.2016.11.11.33

[3] Roh, M.C., Fazli, S., Lee, S.W. (2016). Selective temporal filtering and its application to hand gesture recognition. Applied Intelligence, 45(2): 255-264. https://doi.org/10.1007/s10489-015-0757-8

[4] Escalera, S., Athitsos, V., Guyon, I. (2017). Challenges in multi-modal gesture recognition. In Gesture Recognition, The Springer Series on Challenges in Machine Learning, pp. 1-60. https://doi.org/10.1007/978-3-319-57021-1_1

[5] Piana, S., Staglianò, A., Odone, F., Camurri, A. (2016). Adaptive body gesture representation for automatic emotion recognition. ACM Transactions on Interactive Intelligent Systems (TiiS), 6(1): 1-31. https://doi.org/10.1145/2818740

[6] Nyirarugira, C., Choi, H.R., Kim, T. (2016). Hand gesture recognition using particle swarm movement. Mathematical Problems in Engineering, 2016: 1919824. https://doi.org/10.1155/2016/1919824
[7] Sudha, M.R., Sriraghav, K., Jacob, S.G., Manisha, S. (2017). Approaches and applications of virtual reality and gesture recognition: A review. International Journal of Ambient Computing and Intelligence (IJACI), 8(4): 1-18. https://doi.org/10.4018/IJACI.2017100101
[8] Nguyen-Dinh, L.V., Calatroni, A., Tröster, G. (2016). Supporting one-time point annotations for gesture recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(11): 2270-2283. https://doi.org/10.1109/TPAMI.2016.2637350
[9] Singha, J., Laskar, R.H. (2017). Hand gesture recognition using two-level speed normalization, feature selection and classifier fusion. Multimedia Systems, 23(4): 499-514. https://doi.org/10.1007/s00530-016-0510-0
[10] Leite, D.Q., Duarte, J.C., Neves, L.P., De Oliveira, J.C., Giraldi, G.A. (2017). Hand gesture recognition from depth and infrared Kinect data for CAVE applications interaction. Multimedia Tools and Applications, 76(20): 20423-20455. https://doi.org/10.1007/s11042-016-3959-0
[11] Saha, S., Konar, A., Datta, S. (2017). Fuzzy logic and differential evolution-based hybrid system for gesture recognition using Kinect sensor. Expert Systems, 34(3): e12210. https://doi.org/10.1111/exsy.12210
[12] Moschetti, A., Fiorini, L., Esposito, D., Dario, P., Cavallo, F. (2017). Toward an unsupervised approach for daily gesture recognition in assisted living applications. IEEE Sensors Journal, 17(24): 8395-8403. https://doi.org/10.1109/JSEN.2017.2764323
[13] Jayatilaka, A., Ranasinghe, D.C. (2017). Real-time fluid intake gesture recognition based on batteryless UHF RFID technology. Pervasive and Mobile Computing, 34: 146-156. https://doi.org/10.1016/j.pmcj.2016.04.007
[14] Saad, M., Bleakley, C.J., Nigram, V., Kettle, P. (2018). Ultrasonic hand gesture recognition for mobile devices. Journal on Multimodal User Interfaces, 12(1): 31-39. https://doi.org/10.1007/s12193-017-0257-8
[15] Roma, G., Xambó, A., Freeman, J. (2018). User-independent accelerometer gesture recognition for participatory mobile music. Journal of the Audio Engineering Society, 66(6): 430-438. https://doi.org/10.17743/jaes.2018.0026
[16] Smith, K.A., Csech, C., Murdoch, D., Shaker, G. (2018). Gesture recognition using mm-wave sensor for human-car interface. IEEE Sensors Letters, 2(2): 1-4. https://doi.org/10.1109/LSENS.2018.2810093
[17] Venkatnarayan, R.H., Shahzad, M. (2018). Gesture recognition using ambient light. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 2(1): 1-28. https://doi.org/10.1145/3191772
[18] Kim, J.H., Hong, G.S., Kim, B.G., Dogra, D.P. (2018). deepGesture: Deep learning-based gesture recognition scheme using motion sensors. Displays, 55: 38-45. https://doi.org/10.1016/j.displa.2018.08.001
[19] Chaudhary, A., Raheja, J.L. (2018). Light invariant real-time robust hand gesture recognition. Optik, 159: 283-294. https://doi.org/10.1016/j.ijleo.2017.11.158
[20] Gavrilova, M.L., Wang, Y., Ahmed, F., Paul, P.P. (2017). Kinect sensor gesture and activity recognition: New applications for consumer cognitive systems. IEEE Consumer Electronics Magazine, 7(1): 88-94. https://doi.org/10.1109/MCE.2017.2755498
[21] Negin, F., Rodriguez, P., Koperski, M., Kerboua, A., Gonzàlez, J., Bourgeois, J., Chapoulie, E., Robert, P., Bremond, F. (2018). PRAXIS: Towards automatic cognitive assessment using gesture recognition. Expert Systems with Applications, 106: 21-35. https://doi.org/10.1016/j.eswa.2018.03.063
[22] Shahzad, M., Zhang, S. (2018). Augmenting user identification with WiFi based gesture recognition. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 2(3): 1-27. https://doi.org/10.1145/3264944
[23] Koh, J.I., Cherian, J., Taele, P., Hammond, T. (2019). Developing a hand gesture recognition system for mapping symbolic hand gestures to analogous emojis in computer-mediated communication. ACM Transactions on Interactive Intelligent Systems (TiiS), 9(1): 1-35. https://doi.org/10.1145/3297277
[24] Yasen, M., Jusoh, S. (2019). A systematic review on hand gesture recognition techniques, challenges and applications. PeerJ Computer Science, 5: e218. https://doi.org/10.7717/peerj-cs.218