Sensors and Materials, Vol. 31, No. 11 (2019) 3451–3463
MYU Tokyo

S & M 2023

Dual-input Control Interface for Deep Neural Network Based on Image/Speech Recognition

Neng-Sheng Pai,* Yi-Hsun Chen, Chin-Pao Hung,
Pi-Yun Chen, Ying-Che Kuo, and Jun-Yu Chen

Department of Electrical Engineering, National Chin-Yi University of Technology,
Taichung 41170, Taiwan (ROC)

(Received May 24, 2019; accepted October 8, 2019)

Keywords: Adaboost, convolutional neural network, Mel-frequency cepstral parameters, long short-term memory neural network

The objective of this study was to design a control interface for dual-input video/audio recognition consisting of two input interface systems, hand posture and speech recognition, with the use of specific hand postures or voice commands for control without the need for wearable devices. Original video camera images were preprocessed for hand posture recognition, and the face in the image was used as the reference point and identified using the Adaboost classifier. An image of a specific size was selected as the recognition input image to increase the recognition speed. A neural network comprising convolutional, activation, max pooling, and fully connected layers was used to classify and recognize hand posture images as well as speech. Long short-term memory (LSTM) in a recurrent neural network (RNN) was used to achieve speech recognition. Speech features were extracted by preprocessing, and Mel-frequency cepstral coefficients (MFCCs) and a fast Fourier transform (FFT) were then used to convert the signals from the time domain to the frequency domain. The frequency domain signals subsequently underwent a discrete cosine transform through triangular bandpass filters to derive MFCCs as the speech eigenvalue input. The speech feature parameters were then input to the LSTM neural network to make predictions and achieve speech recognition. Experimental results showed the image/speech dual-input control interface had good sound recognition capability, supporting the findings of this study.

1. Introduction

The convolutional neural network (CNN)(1) was first proposed by LeCun in 1989. A CNN is a type of deep learning model with layered learning features. However, owing to the limited performance of computer hardware at the time, the CNN concept could not be effectively realized. However, modern graphics processing units (GPUs) are very powerful and their substantial computing power has been effectively applied in deep-learning computing. As a result, CNN development has flourished and CNNs are widely used in many fields, such as object detection(2) and face recognition. Current applications are focusing on the development

*Corresponding author: e-mail: pai@ncut.edu.tw
https://doi.org/10.18494/SAM.2019.2481

of artificial intelligence, and natural language processing and speech recognition have been established.(3) The successful application of LeNet-5(4) in the field of handwritten character recognition has also drawn the attention of the academic community towards CNNs. The features learned through CNNs display a stronger discriminative and learning ability than artificial design features. Long short-term memory (LSTM)(5) was proposed by Hochreiter and Schmidhuber in 1997 and the advantage of "recurrence" has been widely applied in speech recognition and emotion analysis. A CNN and LSTM were used as the main structures in this study in view of the excellent image/speech input feature recognition capability of CNNs.

2. Introduction of the Hardware and System Environment

The aim of this study was to enrich the control interface of a quadrotor system with control signals of gestures and voices, where the system structural diagram is shown in Fig. 1. It consists of two parts: the ground computing terminal and the quadrotor flight controller. Wireless signals were transmitted between them using Wi-Fi.

3. Hand Posture Recognition

The images captured using a video camera were processed and then classified using a CNN, which consisted of convolutional, activation, maxpooling, and fully connected layers, to establish hand posture recognition from images.

3.1 Image preprocessing

The preprocessing of the images included conversion from color to grayscale. The Adaboost classifier(6,7) was used to identify the face in the image as a reference point, and an image of a

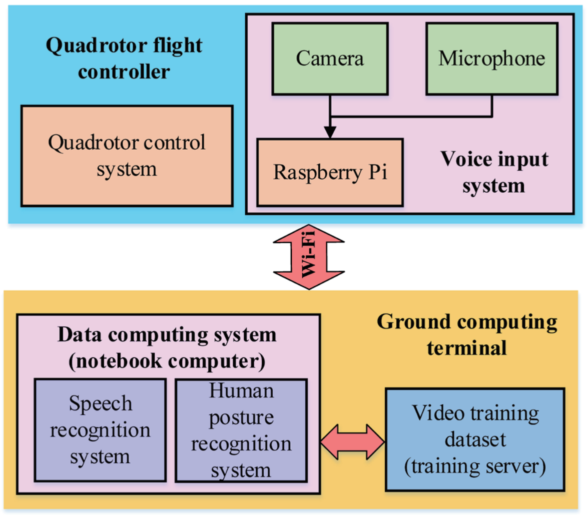

Quadrotor flight controller
Camera
Microphone
Quadrotor control system
Raspberry Pi
Voice input system
Wi-Fi
Data computing system (notebook computer)
Ground computing terminal
Speech recognition system
Human posture recognition system
Video training dataset (training server)

Fig. 1. (Color online) Structure of system.

specific size was selected as the recognition input. The 45° Haar rectangle feature[8] proposed by Lienhart and Maydt was used to extract face sample features, which were placed in the classifier for use in training. After classifier training, the robust linear tandem classifier[9] formed by multiple weak classifiers was obtained and is shown in Fig. 2.

3.2 CNN network structure

The CNN network architecture used in this paper is shown in Table 1.

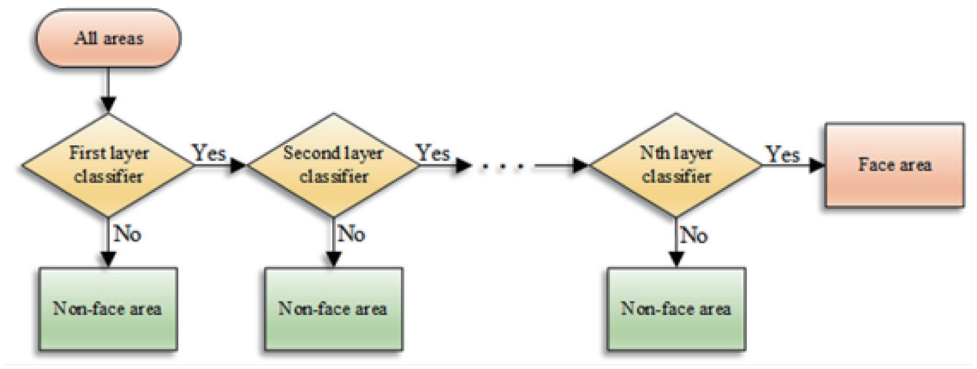


Fig. 2. (Color online) Structure of cascade classifier.

Table 1
CNN network structure.

Layer	Setting	Parameters
Conv 1	Input shape: 350 × 420 Filters: 16 Kernel size: 5 × 5 Activation: relu Stride: 1 × 1 Padding: same	416
MaxPooling 1	Pool size: 2 × 2 Stride:2 × 2 Padding: valid	
Dropout 1	0.25	
Conv 2	32, 5 × 5, relu	12832
MaxPooling 2	Pool size: 2 × 2	
Dropout 2	0.25	
Conv 3	64, 5 × 5, relu	51264
MaxPooling 3	Pool size: 2 × 2	
Dropout 3	0.25	
Conv 4	128, 5 × 5, relu	204928
MaxPooling 4	Pool size: 2 × 2	
Dropout 4	0.25	
Conv 5	128, 5 × 5, relu	409728
MaxPooling 5	Pool size: 2 × 2	
Dropout 5	0.25	
FC	Number of nodes: 256	16640
Dense	Number of nodes: 18 Activation: softmax	774

4. Speech Recognition

Speech commands were input to a microphone and converted into digital potential signals by an analog-to-digital converter (ADC). After the features had been obtained, the LSTM(10) in a recurrent neural network (RNN)(11) was used to perform speech recognition. Speech recognition was accomplished using voice signal preprocessing, speech eigenvalue extraction, and the RNN.

4.1 Voice signal preprocessing

Voice signals were preprocessed to extract the required eigenvectors. Preprocessing included digital sampling, preemphasis, frame, and window function steps, as shown in Fig. 3.

4.2 Speech feature extraction

The signals were converted to the frequency domain using a discrete Fourier transform and observed using the energy distribution, and Mel-frequency cepstral coefficient (MFCCs)(12–14) were used to extract features. Log energy and the delta cepstrum were added to increase the diversity of the eigenvalues. The MFCC process is shown in Fig. 4.(15)

4.3 LSTM

LSTM was employed to avoid the problem of gradient disappearance. The concept of the memory cell and gate was incorporated into the RNN. Figure 5 shows a single memory block of LSTM,(16) which includes three gates: Input, Output, and Forget. The gates are all nonlinear

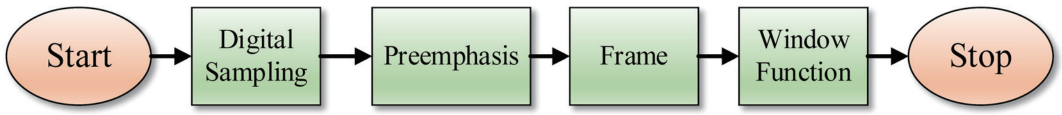


Fig. 3. (Color online) Flowchart of voice signal preprocessing.

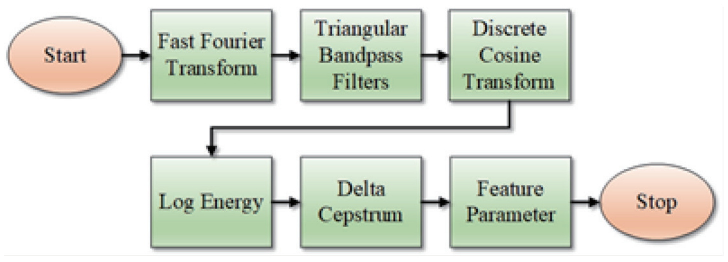


Fig. 4. (Color online) Flowchart of Speech feature extraction.

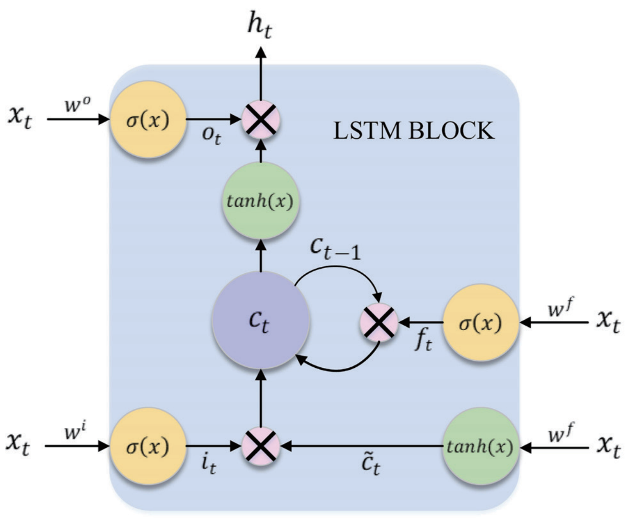


Fig. 5. (Color online) Schematic diagram of LSTM memory block.

summing units. The main function of a gate is to determine whether to accept an input signal, to forget the past status, or to output a predicted result, as well as other actions. The status update process can be represented by the following equations:

Input gate
$$i_t = \sigma\left(w^i x_t\right), \tag{1}$$

Output gate
$$o_t = \sigma\left(w^o x_t\right), \tag{2}$$

Forget gate
$$f_t = \sigma\left(w^f x_t\right), \tag{3}$$

New memory cell
$$\tilde{c}_t = \tanh\left(w^c x_t\right), \tag{4}$$

Update memory cell
$$c_t = f_t c_{t-1} + i_t \tilde{c}_t\,, \tag{5}$$

Update hidden cell
$$h_t = o_t \tanh c_t. \tag{6}$$

Here, t represents the time step, x_t is the input vector of the LSTM block, $\sigma(.)$ is the logistic sigmoid function, tanh(x) represents the hyperbolic tangent function, w^i, w^o, w^f, and w^c represent the weights of the matrices, $\tilde{c}_t$ represents the candidate value, c_t represents the cell status output, and c_{t-1} represents the status output of the previous unit.

After speech was input, eigenvectors were extracted through speech preprocessing, and speech features were extracted using MFCCs. Finally, the processed speech features were placed in the LSTM to carry out recognition. Recognition results were then converted into appropriate control signals.

5. Experimental Results

This paper is divided into two parts. The first part is concerned with hand posture recognition involving input images for gray scaling and the use of the Adaboost face recognition classifier to identify a face in the image as a reference point; the second part involves speech recognition voice signals, which underwent preprocessing to derive speech eigenvectors. MFCCs were then used to extract the speech features and the LSTM was used to perform recognition.

5.1 Experimental results of hand posture recognition

The images used for the input were 640 × 480 pixels in size. About 50000 images were used for training and 5000 images were used for testing. Six different instructional postures were used as shown in Figs. 6–8.

Since the preset device in this paper was a quadrotor drone, the video camera was situated 2 m above ground level and 3.5 m from the user, as shown in Fig. 9.

After the input image had been grayscaled, the Adaboost face classifier was used to identify the face in the image as the reference point, and the face was selected as the input image for turning angle recognition, with a size of 40 × 40, as shown in Fig. 10.

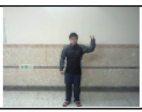
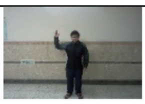
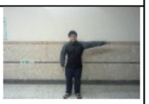
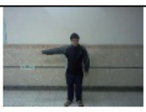
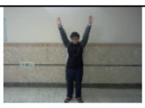
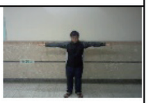
Instruction	Front	Behind	Left
Posture			
Instruction	Right	Up	Down
Posture			

Fig. 6. (Color online) Instructions for hand posture recognition.

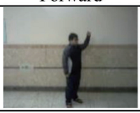
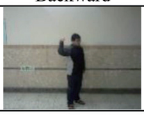
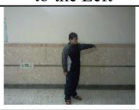
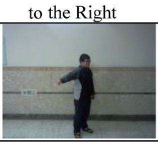
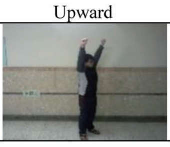
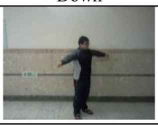
Left Oblique 45° Forward	Left Oblique 45° Backward	Left Oblique 45° to the Left
		
Left Oblique 45° to the Right	Left Oblique 45° Upward	Left Oblique 45° Down
		

Fig. 7. (Color online) Instructions for left oblique angles at 45°.

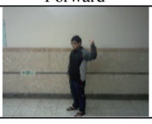
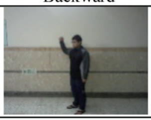
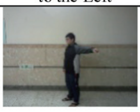
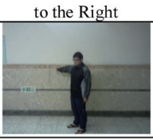
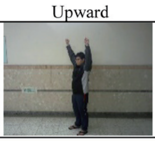
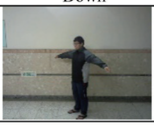
Right Oblique 45° Forward	Right Oblique 45° Backward	Right Oblique 45° to the Left
		
Right Oblique 45° to the Right	Right Oblique 45° Upward	Right Oblique 45° Down
		

Fig. 8. (Color online) Instructions for right oblique angles at 45°.

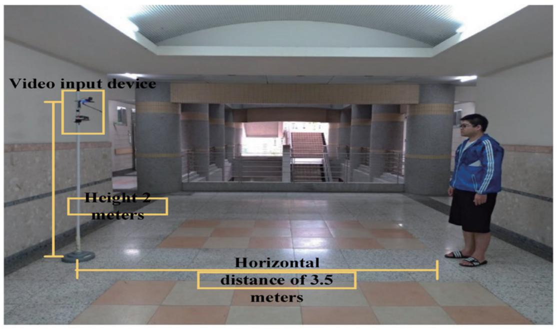


Fig. 9. (Color online) Test distance diagram.

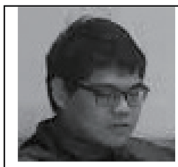
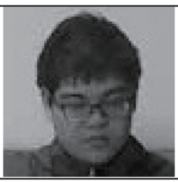
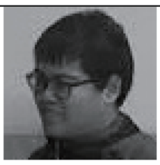
		
Left oblique 45 degrees	positive	Right oblique 45 degrees

Fig. 10. Input image for turning angle recognition.

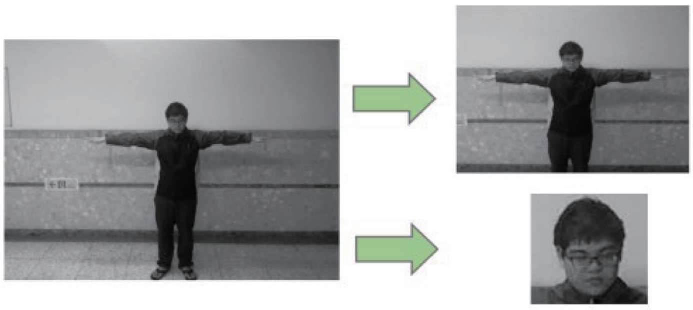


Fig. 11. (Color online) Image input for posture recognition.

After the input image had been converted to grayscale, the Adaboost face recognition classifier was used to identify the face in the image as the reference point and select it as the input image for turning angle recognition. The computing time was also reduced in addition to the improved recognition rate. As shown in Fig. 11, the samples used for these datasets were from 10 people, although the recognition rate may be reduced if more people need to be recognized; the image on the left is the uncut original image. The face and upper body were selected using Adaboost.

The posture recognition interface used in this study had two CNN structures, the batch size was 50, and each epoch had 1000 iterations. The first CNN determines the angular view of the body presented to the camera based on the direction of the face, as shown in Fig. 12 and Table 2. The second CNN predicts the hand posture output as shown in Fig. 13 and Table 3.

Figures 14–16 show actual recognition images. In recognition interface windows, the pictures show the postures of front, left oblique angle of 45°, and right oblique angle of 45° respectively, and the corresponding results of posture, appeared at the upper right window after recognition was completed.

5.2 Experimental results of speech recognition

The training datasets of the speech recognition system contained 1187 data entries. Among these, six were voice instructions: left, right, front, behind, up, and down. Ten students recorded each instruction 20 times to make the training datasets and the recording time was 2 s.

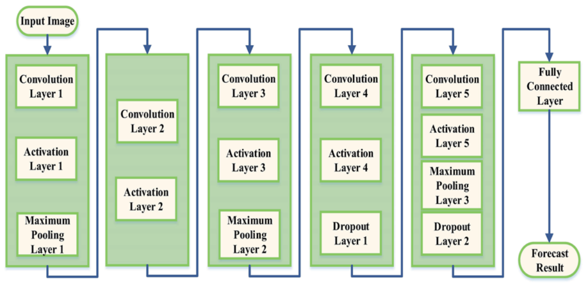


Fig. 12. (Color online) Structure diagram of CNN angle recognition.

Table 2
Angle recognition accuracy results.

	Epoch	Accuracy	Loss	Time (s)
	1	0.8134	0.0763	636
	2	0.9134	0.0301	594
CNN	3	0.9548	0.0213	570
	4	0.9866	0.0174	567
	5	**0.9962**	**0.0138**	**585**
SVM	**1**	**0.9600**		75600

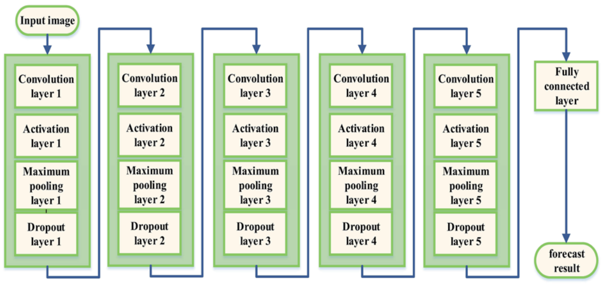


Fig. 13. (Color online) Structure diagram of CNN posture recognition.

Table 3
Posture recognition accuracy results.

	Epoch	Accuracy	Loss	Time (s)
	1	0.7287	0.0847	1756
	2	0.8531	0.0386	1703
CNN	3	0.9766	0.0289	1689
	4	0.9823	0.0204	1697
	5	**0.9951**	**0.0138**	**1709**
SVM	**1**	**0.9400**		**162000**

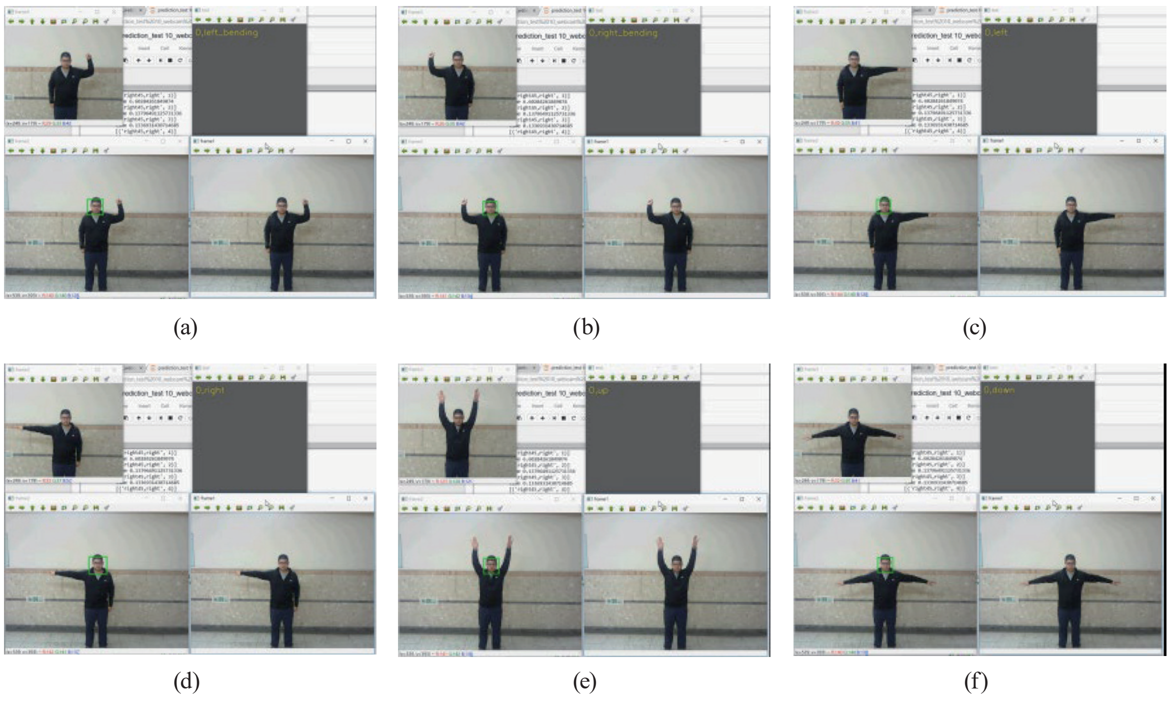
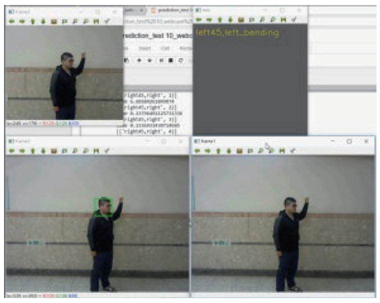
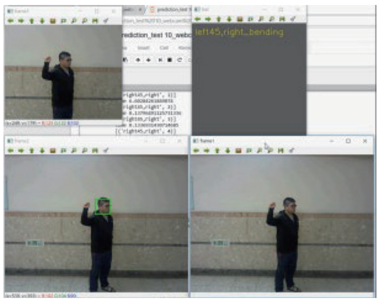


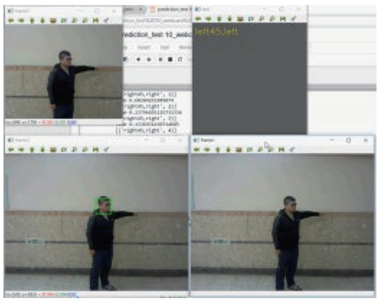
Fig. 14. (Color online) Actual test results of front posture recognition.


left45_left_bending

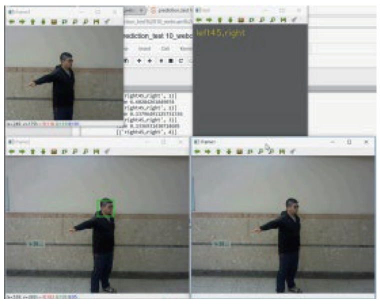
(a)


left45_right_bending

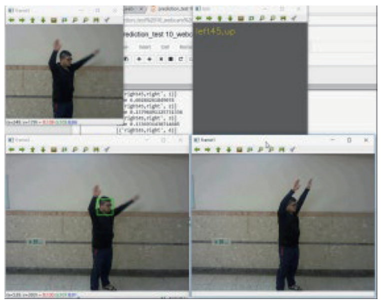
(b)


left45_left

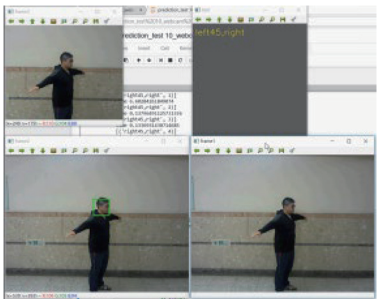
(c)


left45_right

(d)

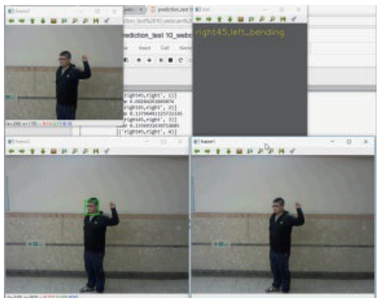

left45_up

(e)

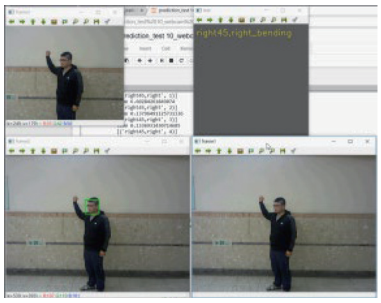

left45_right

(f)

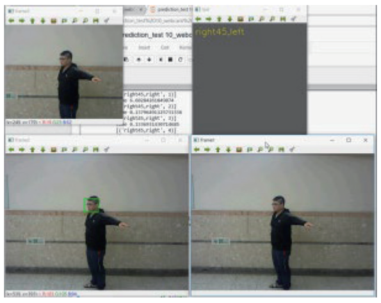
Fig. 15. (Color online) Actual test results of posture recognition at a left oblique angle of 45°.


right45_left_bending

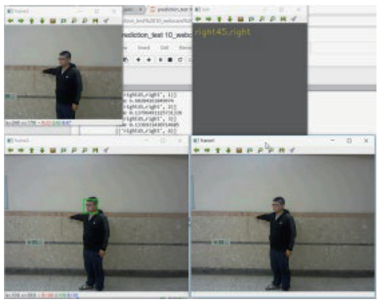
(a)


right45_right_bending

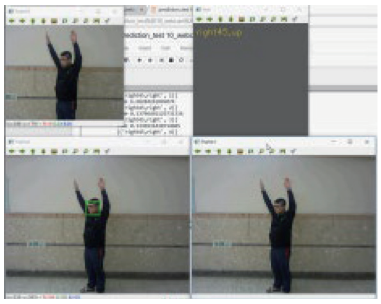
(b)


right45_left

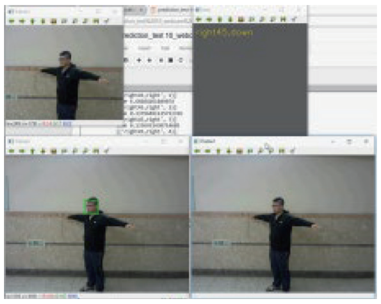
(c)


right45_right

(d)


right45_up

(e)


right45_down

(f)

Fig. 16. (Color online) Actual test results of posture recognition at a right oblique angle of 45°.

To increase the rate of recognition, the samples included loud and soft commands as well as fast and slow, and low- and high-pitched voices. The original datasets contained 1200 samples, but after the defective samples had been removed, 1187 samples remained, which were used as training data. Figure 17 shows the signals of the original voice instructions.

The speech feature extraction involves extracting features by preprocessing the speech signals and MFCCs. Figure 18 shows the spectrograms of the voice instructions converted to the frequency domain.

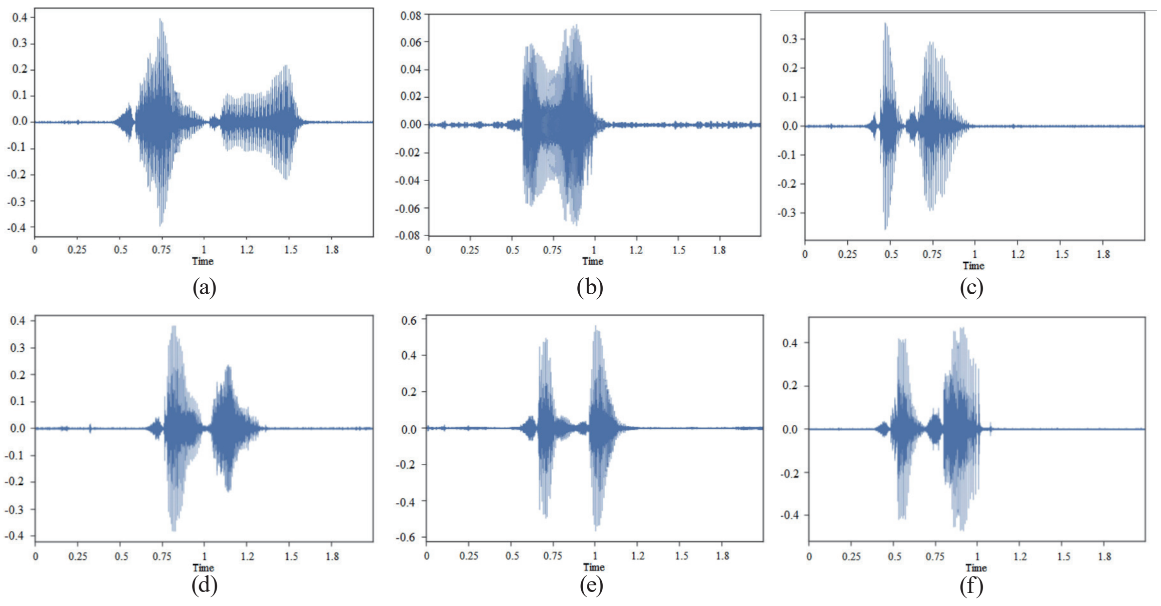


Fig. 17. (Color online) Waveforms signals of original voice input instructions: (a) left, (b) right, (c) front, (d) behind, (e) up, and (f) down.

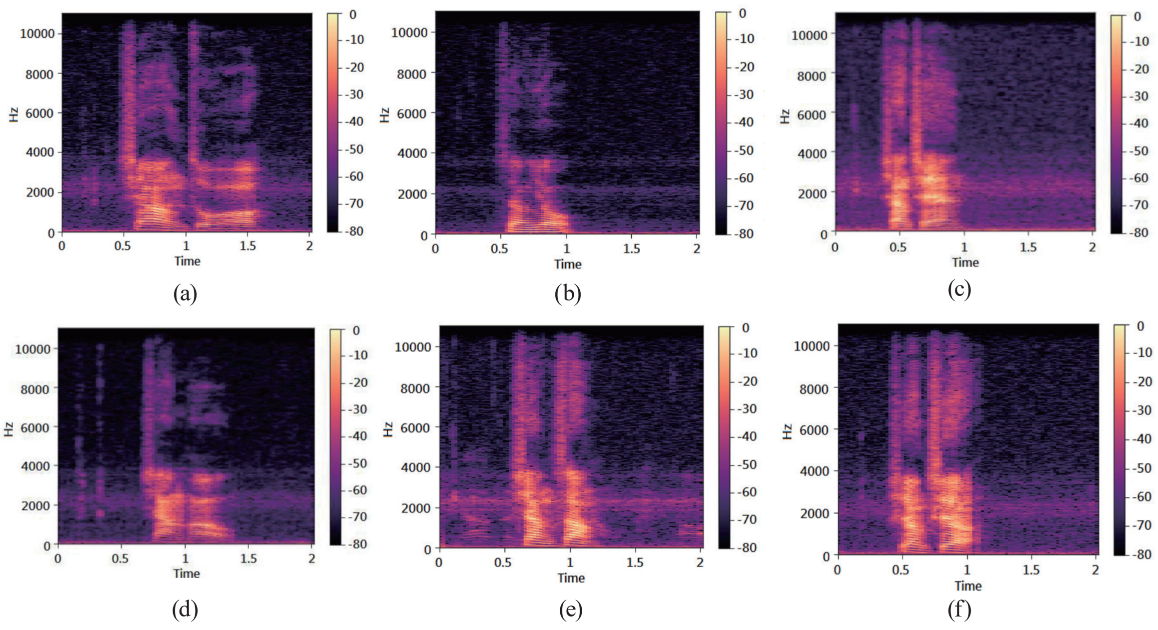


Fig. 18. (Color online) Frequency domain spectrograms of voice instructions: (a) left, (b) right, (c) front, (d) behind, (e) up, and (f) down.

In this study, LSTM improved by the RNN was used for speech recognition prediction. Figures 19–21 show flowcharts of the LSTM, RNN, and CNN, respectively. The LSTM was used as the speech recognition system in this paper.

Table 4 shows the approximate accuracy rates of the LSTM, RNN, and CNN in terms of speech recognition. Because the input voice signals used in this study had no correlation with the preceding or succeeding text and were simply single words input as sounds into the LSTM, the RNN memory function was not effectively implemented. Figure 22 shows the image patterns obtained from actual speech recognition.

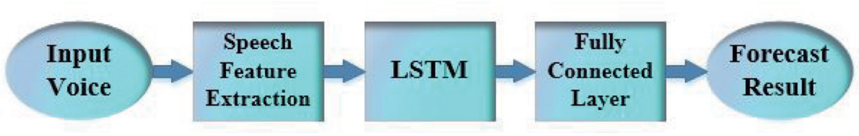


Fig. 19. (Color online) Flowchart of LSTM.

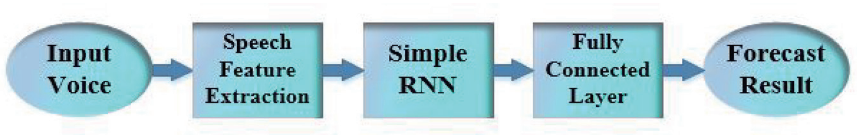


Fig. 20. (Color online) Flowchart of RNN.

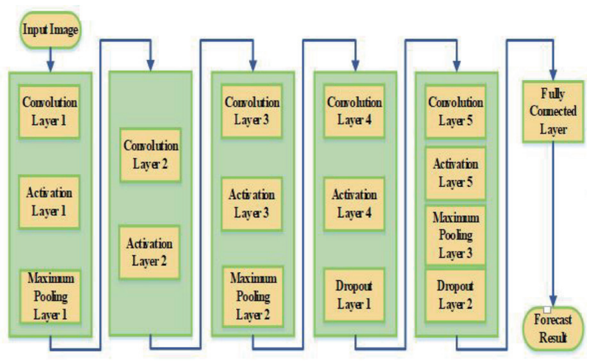
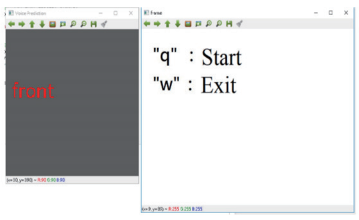


Fig. 21. (Color online) Flowchart of CNN.

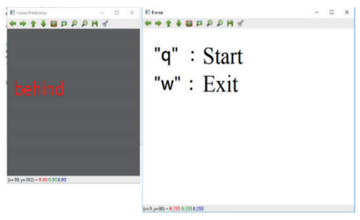
Table 4
Speech recognition accuracy results.

	Epoch	Accuracy	Time (s)
LSTM	1	0.9836	325
RNN	1	0.9802	308
CNN	1	0.7923	239
	2	0.8681	215
	3	0.9482	211
	4	0.9761	206
	5	0.9811	216


front

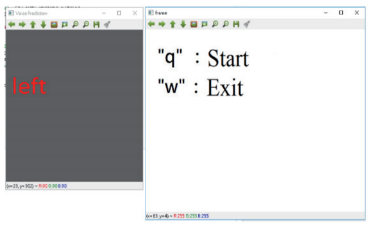
"q" : Start
"w" : Exit

(a)


behind

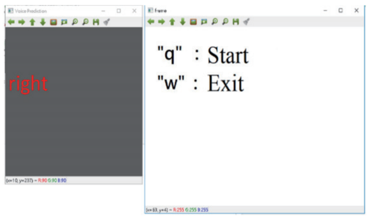
"q" : Start
"w" : Exit

(b)


left

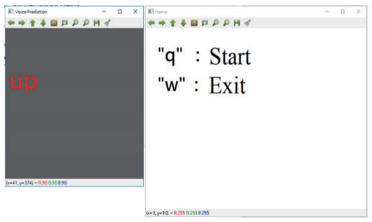
"q" : Start
"w" : Exit

(c)


right

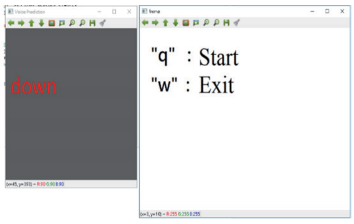
"q" : Start
"w" : Exit

(d)


up

"q" : Start
"w" : Exit

(e)


down

"q" : Start
"w" : Exit

(f)

Fig. 22. (Color online) Actual test results of speech recognition.

6. Conclusion

In this study, a dual-input control interface realized for a deep neural network was implemented. A CNN and LSTM were used to achieve hand posture and voice recognition. This made control by a specific posture or voice command possible without the need for a wearable device. In the hand posture recognition, the images were used for the CNN parameter training of the data server. A Raspberry Pi3 Wi-Fi transmission module was used to transmit images to the video processing device where hand posture recognition was performed. In the speech recognition, speech data were also used to train LSTM parameters through the training server. Voice signals were preprocessed and MFCCs were used to obtain speech feature parameters and LSTM was used to make predictions.

Acknowledgments

This work was supported by the Ministry of Science and Technology, Taiwan, under contract number MOST 107-2221-E-167-024 (duration: August 1, 2018–July 31, 2019).

References

1 Y. LeCun: Department of Computer Science University of Toronto, Technical Report CRG-TR-89-4 (1989).
2 M. Menikdiwela, C. Nguyen, H. Li, and M. Shaw: IEEE 2017 Int. Conf. Image and Vision Computing New Zealand (IVCNZ, 2017). https://doi.org/10.1109/IVCNZ.2017.8402455.
3 H. Geoffrey, D. Li, Y. Dong, G. E. Dahl, A.R. Mohamed, J. Navdeep, S. Andrew, V. Vincent, N. Patrick, T. N. Sainath, and B. Kingsbury: IEEE Signal Process. Mag. **29** (2012) 82. https://doi.org/10.1109/MSP.2012.2205597
4 Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner: Proc. IEEE **86** (1998) 2278. https://doi.org/10.1109/5.726791
5 S. Hochreiter and J. Schmidhuber: Neural Computation **9** (1997) 1735. https://doi.org/10.1162/neco.1997.9.8.1735

6 P. Viola and M. Jones: Proc. 2001 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR, 2001) 511. https://doi.org/10.1109/CVPR.2001.990517
7 P. I. Rani and K. Muneeswaran, 2014 Int. Conf. Communication and Network Technologies (2014) 276. https://doi.org/10.1109/CNT.2014.7062769
8 R. Lienhart and J. Maydt: Proc. Int. Conf. Image Processing (2002) 900. https://doi.org/10.1109/ICIP.2002.1038171
9 R. Lienhart, L. Liang, and A. Kuranov: Int. Conf. Multimedia and Expo. ICME '03 (2003) 277. https://doi.org/10.1109/ICME.2003.1221607
10 S. Hochreiter and J. Schmidhuber: Workshop, Spatiotemporal Models in Biological and Artificial Systems (1996) 65.
11 W. Kahuttanaseth, A. Dressler, and C. Netramai, 2018 5th Int. Conf. Business and Industrial Research (ICBIR, 2018) 161. https://doi.org/10.1109/ICBIR.2018.8391185
12 F. Zheng, G. Zhang, and Z. Song: J. Comput. Sci. Technol. **16** (2001) 582. https://doi.org/10.1007/BF02943243
13 T. Ganchev, N. Fakotakis, and G. Kokkinakis, Proc. SPECOM-**2005** (2005) 191. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.75.8303
14 V. Tiwari: Int. J. Emerging Technol. **1** (2010) 19.
15 A. K. H. Al-Ali, D. Dean, B. Senadji, V. Chandran, and G. R. Naik, IEEE Access **5** (2017) 15400. https://doi.org/10.1109/ACCESS.2017.2728801
16 R. Zazo, P. S. Nidadavolu, N. Chen, J. Gonzalez-Rodriguez, and N. Dehak, IEEE Access **6** (2018) 22524. https://doi.org/10.1109/ACCESS.2018.2816163