Implementation of A Pyramidal Manipulator for Handwritten Character Recognition and Writing Using CNN

Shih-Ting Wang, *I-Hsum Li, and Wei-Yen Wang

Abstract—**This paper integrates information, mechanical, and electrical technologies to design a pyramidal-shaped pyramidal manipulator for handwritten-characters recognition and writing. In terms of mechanism design, we use the iGus pyramidal-shaped manipulator, which is a modular structure that can be assembled into a pyramidal-shaped manipulator by yourself. The motion of the iGus pyramidal-shaped manipulator is controlled by an Advantech motion control module, in which we use three AC servo motors to drive three sliders and enable the end-effector moving in three-dimensional space. In terms of character recognition, we adopt a convolutional neural network (CNN) for recognizing handwritten English letters and numbers, and then return the recognition results to the robot arm via MQTT. After that, the manipulator writes the character. Finally, through experiments, we verify that our developed image recognition cone-shaped pyramidal manipulator can accurately recognize handwritten fonts and return them to the robot arm to complete the font writing.**

Index Terms*—*Convolutional neural network, Deep learning, Image recognition, Robot arm.

I. Introduction

In recent years, robots plays an increasingly vital role in our lives and have being applied to a wider range of industrial fields. As a result, in the future, robots not only perform repetitive tasks but must also can learn autonomously, identify solutions to problems on their own, and accumulate experiences. These can be achieved through deep learning techniques [1-3].

Convolution Neural Network (CNN) is a deep learning model built with reference to the visual organization of the human brain. It is often used for capturing image features, and has been applied to many fields, such as handwritten recognition, face recognition, and object detection. CNN can extract visual features by itself instead of describing features manually before training. The extracted features are further fed into multi convolutional layers and pooling layers for further feature extraction, thus enhancing the learning efficiency of the neural network [4-7].

This work was supported by the National Science and technology Council, Taiwan, under Grants MOST 110-2221-E-032-036, MOST 111-2221-E-003-025, and MOST 110-2634-F-A49-004. (Corresponding author: I-Hsum Li)

Shih-Ting Wang is with the Department of Electrical Engineering, National Taiwan Normal University, Taipei 106, Taiwan (e-mail: melody870322@gmail.com).

I-Hsum Li is with the Department of Mechanical and Electro-Mechanical Engineering, Tamkang University, New Taipei City, Taiwan (e-mail: ihsumlee@mail.tku.edu.tw).

Wei-Yen Wang is with the Department of Electrical Engineering, National Taiwan Normal University, Taipei 106, Taiwan (e-mail: wywang@ntnu.edu.tw).

The main contribtuions of this paper are:

1. We develop an intelligent pramidal manipulator capable of recognizing handwritting characters, communciating with a remote server and writing down the recognized characters.

2. The intelligent pramidal manipulator exhibits real-time response from capturing an image, sending the image to a server and recognizing a character, to write down the character.

II. The Application of Deep Learning in the Design of Pyramidal Manipulator

Figure 1 depicts the overall system of the intelligent pyramidal manipulator for the issue of handwritten character recognition and writing it down. The system is developed by using the OpenCV computer vision library for image processing and the Keras deep learning library to train and test a CNN deeper learning model for the handwritten characters reorganization. While trained well, the CNN model is implemented in the pyramidal manipulator. The detailed steps are as follows:

Step 1: The pyramidal manipulator processes the handwritten character image captured by the pramidal manipulator (IP: 192.168.1.182).

Step 2: The pyramidal manipulator sends the processed image to the server (IP: 192.168.1.190) via MQTT and require the trained CNN to recognize the handwritten character.

Step 3: The server answers the recognized result and then the pyramidal manipulator writes down the character.

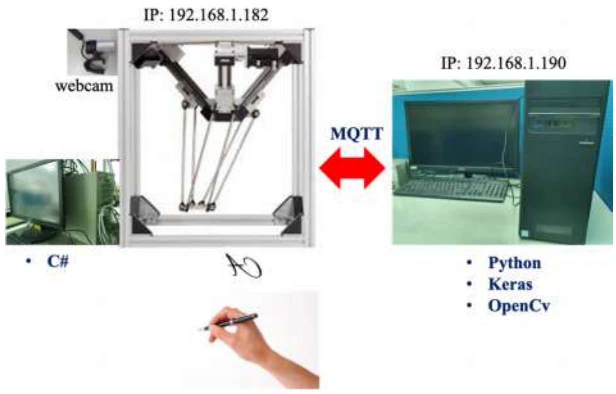

Fig. 1. Diagram of the overall architecture for handwritten character recognition

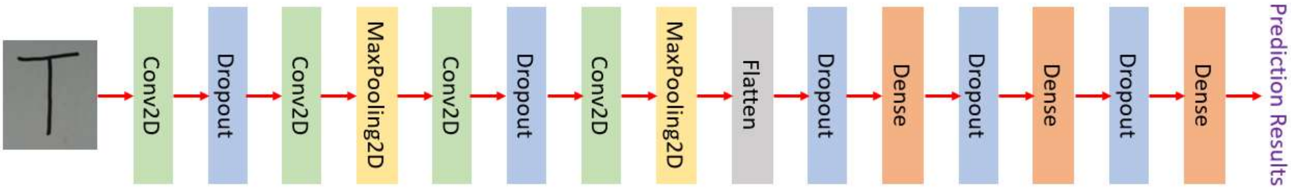

Fig. 2. Overall CNN architecture for handwritten character recognition.

A. Handwritten Network

Figure 2 shows the CNN architecture for the handwritten recognition network. We use four convolutional layers to extract local features and reduce the dimensionality of the input image and use two max-pooling layers to further reduce the dimensionality of the CNN model, avoiding the issue of computational burden and the risk of overfitting. Additionally, four Dropout layers and three Dense Layers are applied for preventing overfitting. Subsequently, we utilize the flatten layer to flatten the captured features into the fully connected layer for handwritten character classification.

B. MQTT

MQTT is a light communication protocol over TCP/IP and is helpful to transmit information between the pyramidal manipulator and server. In this paper, the server acts as the Publisher and the industrial personal computer (IPC) next to the Delta robot acts as the Subscriber while sending a recognized handwritten character back to the industrial computer from the server. On the other hand, the IPC publishes a topic to require for recognizing handwritten characters. Here, the Mosquitto MQTT Server [8] is used as the Broker to receive messages from the Publisher and then forward them to the Subscriber.

C. The establishment of the Delta robot model

Figure 3 shows the single-axis vector diagram of a pyramidal manipulator. We analyze the workspace and motion of terminal point by using forward and inverse kinematics for the pyramidal manipulator, respectively. After that, we can calculate the required motion of the three axes using inverse kinematics if a targeted coordinate in the three-dimensional space is given.

Inverse kinematics analysis: The three-axis structure of our developed pyramidal manipulator is identical, so the inverse kinematics result of one axis is sufficient to determine the kinematics of the entire pyramidal manipulator. The single-axis vector of the pyramidal manipulator is shown in Figure 3, and its closed-loop equation is as follows:

$$\overrightarrow{A_\iota C_\iota} + \overrightarrow{C_\iota B_\iota} = \overrightarrow{OP} + \overrightarrow{PB_\iota} - \overrightarrow{OA_\iota}, \tag{1}$$

$$l_i \boldsymbol{I}_{0i} = \boldsymbol{L}_i - d_i \boldsymbol{d}_{0i}, \tag{2}$$

and

$$\boldsymbol{L}_i = \boldsymbol{P} + \boldsymbol{b}_i - \boldsymbol{a}_i, \tag{3}$$

where $\boldsymbol{I}_{0i}$ is the unit vector of $\overrightarrow{C_\iota B_\iota}$, d_i is the linear displacement for the ith axis, which stands for the movement of the slider, an*d* $\boldsymbol{d}_{0i}$ is represented as the unit vector of $\overrightarrow{A\iota C\iota}$, which can be expressed as follows:

$$d_{0i} = [-\cos\alpha_i \cos\beta_i \; -\cos\alpha_i \sin\beta_i \; -\sin\alpha_i \;]^T, \tag{4}$$

where the angles α_i and β_i are depicts in Fig. 3. Taking square on both sides of equation (2) and rearranging, we get:

$$d_i{}^2 - 2d_i \boldsymbol{d}_{0i}^T \boldsymbol{L}_i + \boldsymbol{L}_i{}^T \boldsymbol{L}_i - l^2 = 0. \tag{5}$$

Finally, by solving equation (5), we can obtain the value of d_i:

$$d_i = \boldsymbol{d}_{0i}^T \boldsymbol{L}_i \pm \sqrt{(d_{0i}^T \boldsymbol{L}_i)^2 - \boldsymbol{L}_i^T \boldsymbol{L}_i + l^2}. \tag{6}$$

There are two solutions of (6) and the solution $d_i = \boldsymbol{d}_{0i}^T \boldsymbol{L}_i + \sqrt{(\boldsymbol{d}_{0i}^T \boldsymbol{L}_i)^2 - \boldsymbol{L}_i^T \boldsymbol{L}_i + l^2}$ is not satisfied with the condition of the construction of the pyramidal manipulator. So, the unique solution that satisfies the structure of the pyramidal manipulator with three sliding axes inclined inward from top to bottom is:

$$d_i = \boldsymbol{d}_{0i}^T \boldsymbol{L}_i - \sqrt{(\boldsymbol{d}_{0i}^T \boldsymbol{L}_i)^2 - \boldsymbol{L}_i^T \boldsymbol{L}_i + l^2} \tag{7}$$

Referring to the datasheet of iGus Robot [6], we can find the clear conditions for Eq. (7).

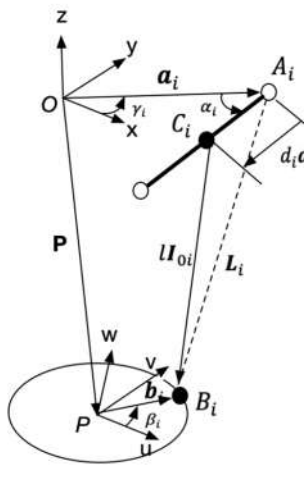

Fig. 3. Diagram of single-axis vector for pyramidal manipulator.

D. Designing Motion Trajectories

The motion of the pyramidal manipulator is regulated by an Advantech motion control card (PCI-1203). So, the motions for the learnt characters can be pre-planned by using Advantech Common Motion Utility, as shown in Fig. 4. Before planning the motions, it is necessary to convert the three displacements

d_1, d_2 and d_3 into the PPU (Pulse Per Unit) values required to regulate the motors. To achieve this, the displacements d_1, d_2 and d_3 have to be multiplied by 514.2857 and divided by 10. These two values can be calculated according to the transmission setup of the pyramidal manipulator, i.e. the size of gear and the length of the gear belt. Fig. 4 shows the "Edit Path" function in the Common Motion Utility. We can use it to build a complete motion according to the trajectory we simulate in Matlab in advance, as shown in Fig. 5. Then, the motion is saved as a bin file to provide the pyramidal manipulator moving along the planned trajectory.

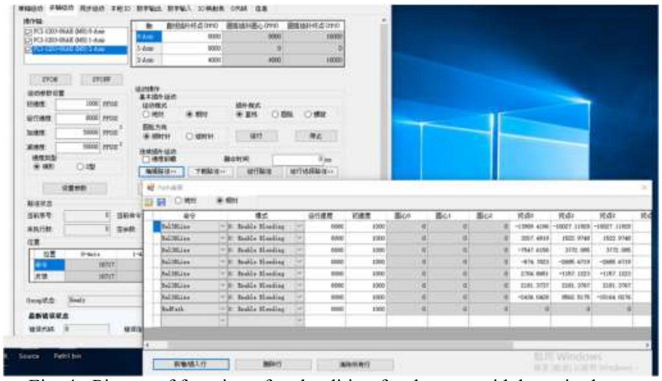

Fig. 4. Picture of function of path editing for the pyramidal manipulator.

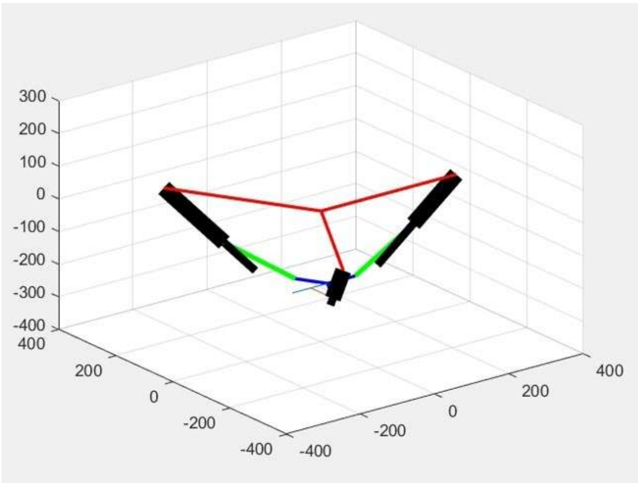

Fig.5 Simulated Trajectory of the characters in Matlab

III. Experimental Results

A. Hardware

In this experiment, we showed that a user wrote down a character with a laptop and the pyramidal manipulator received the requirement, recognized the character, and wrote it down in a paper.

During the training phase, the CNN model for the handwritten recognition network, which is shown in Figure 2, is trained by a server with a RTX 2080TI. During the testing phase, the raw images can be captured by a Microsoft 1425 Webcam, providing 1080p resolution images. Figure 6 shows our complete hardware architecture, which includes a pyramidal robotic arm, an industrial computer, and a distribution box. The pyramidal manipulator offers the following advantages:1) it has a lightweight construction, 2) it provides a new calibration pin for zero positioning, and 3) it is easy to install by yourself [9].

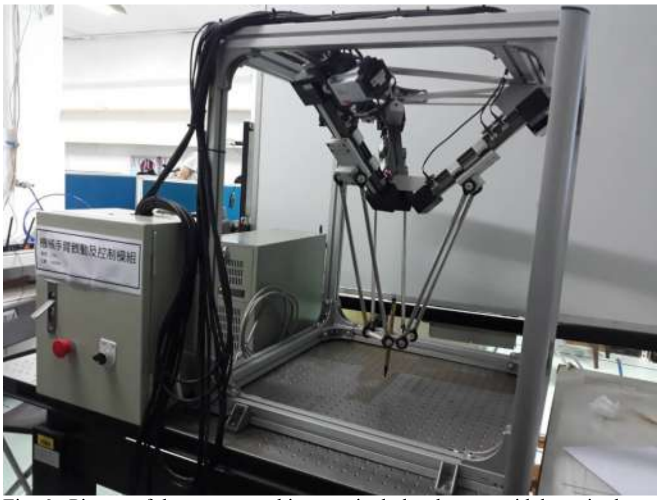

Fig. 6. Picture of the system architecture includes the pyramidal manipulator, industrial computer, and distribution box.

B. Dataset

The dataset for training and testing is from the Chars74K dataset [10]. The Chars74K dataset is a well-known character recognition dataset that provides over 74K images of 62 classes from 55 volunteers. It includes 55 character classes (A-Z and a-z) and 10 digit classes (0-9).

C. Results for the Training Phase

The handwritten recognition model consists of two blocks. The former block begins with one convolutional layer with 32 filters of kernel size 5×5, followed by a same configuration convolutional layer, followed by a max pooling layer with pool size 2 × 2. The latter block starts with one convolutional layer with 64 filters of kernel size 5 × 5, followed by a same configuration convolutional layer and a max pooling layer with pool size 2 × 2. The batch size is 100. Adam optimization and Categorical cross-entropy are used to make the training process more efficient. After training, the training accuracy, training loss, validation accuracy and validation loss for 5 epochs were respectively 0.961, 0.1432, 0.9734 and 0.0942, as shown in Figure 7.

```
Train on 331837 samples, validate on 110613 samples
Epoch 1/5
 - 1021s - loss: 0.4061 - acc: 0.8833 - val_loss: 0.1466 - val_acc: 0.9590
Epoch 2/5
 - 1043s - loss: 0.1925 - acc: 0.9463 - val_loss: 0.1124 - val_acc: 0.9703
Epoch 3/5
 - 1203s - loss: 0.1668 - acc: 0.9543 - val_loss: 0.1012 - val_acc: 0.9724
Epoch 4/5
 - 896s - loss: 0.1520 - acc: 0.9584 - val_loss: 0.0909 - val_acc: 0.9757
Epoch 5/5
 - 790s - loss: 0.1432 - acc: 0.9610 - val_loss: 0.0942 - val_acc: 0.9734
SAVED
CNN Score: 0.9733937240649834
```

Fig. 7. Training and validation results for the handwritten recognition network.

D. Results for the Testing Phase

The recognition results for the handwritten T, K, and U are shown in Figures 7, 8, and 9, respectively. The window in the top left corner displays the image captured by the camera. The user can take a photo for a character by pressing X, and the recognition result appears in the Console. At the same time, the result was transmitted to the pyramidal manipulator through MQTT. After receiving the MQTT message, the manipulator moved to initial position automatically and waited for 5 seconds before starting to write the handwritten characters specified in the message. Figure 10 shows the results of writing the characters T, K, and U. The experimental video is provided in our website: http://ihsumlee.bounceme.net/Lee/web/node/2.

IV. CONCLUSION

This paper utilized Keras and Python to build a convolutional neural network for training and testing over 500,000 handwritten English and numeric characters, and the trained network model was applied to recognize the handwritten characters in real-time. In terms of controlling the pyramidal manipulator, a mathematical model of the pyramidal manipulator was derived and simulated by Matlab to verify the forward and inverse kinematics and plan the motion trajectory of each handwritten characters. In practical experiments, the captured image of the handwritten character was captured by a webcam, and the learned CNN handwritten recognition network was used for prediction. The predicted result was then transmitted to the pyramidal manipulator through the MQTT communication protocol over the network. When the manipulator received the message, it wrote down the character according to the planned trajectory.

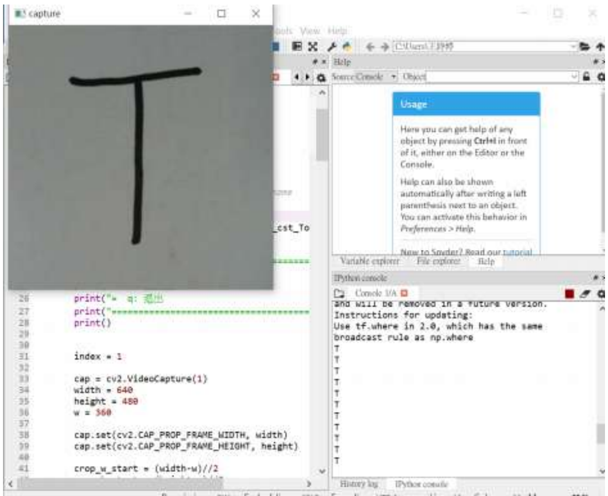

Fig. 7. Results of recognizing a character 'T' for the handwritten recognition network.

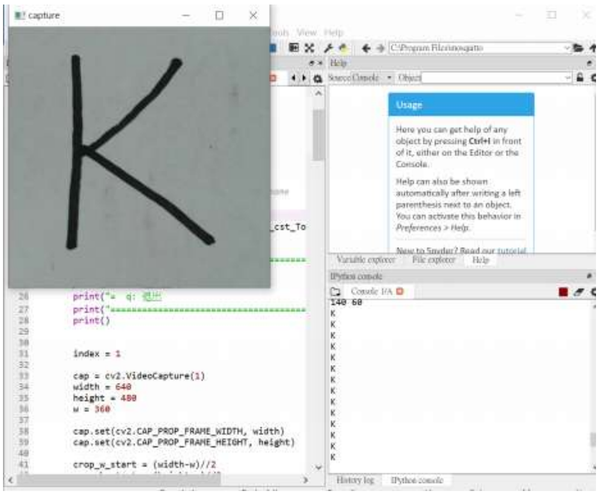

Fig. 8. Results of recognizing a character 'K' for the handwritten recognition network.

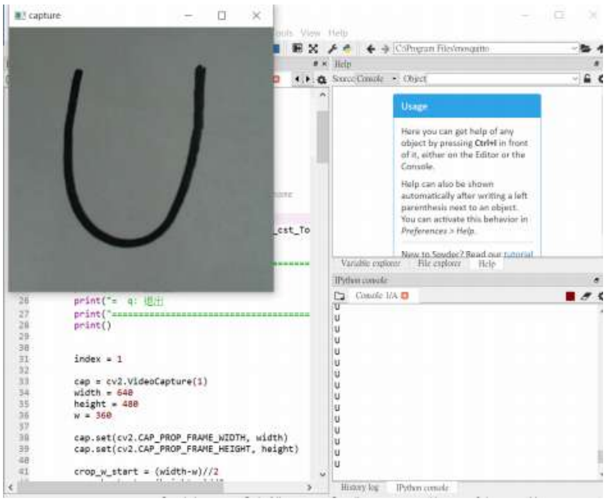

Fig. 9. Results of recognizing a character 'U' for the handwritten recognition network.

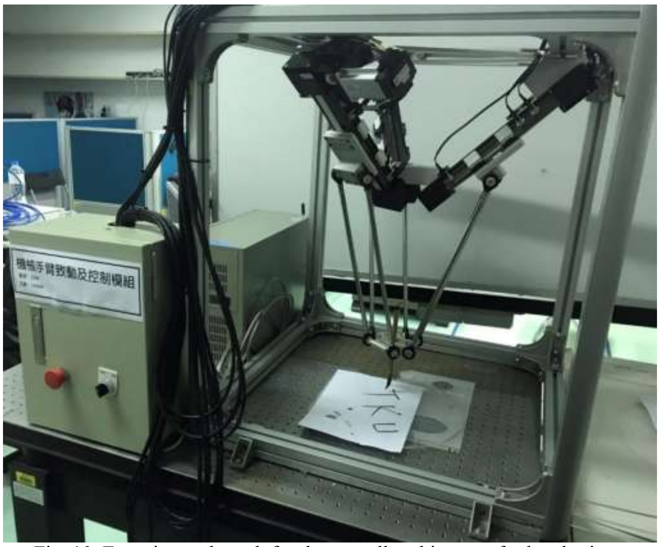

Fig. 10. Experimental result for the overall architecture for handwritten character recognition (http://ihsumlee.bounceme.net/Lee/web/node/2)

REFERENCES

[1] Ian Lenz, "Deep learning for robotics," Cornell University, January 2016.

[2] Ruo-Syuan Mei and Jen-Yuan (James) Chang, "Human Expert Skill Transferring and Imitation for Teleoperated Dual-Arm Manipulation," Special issue on new generation of robot-centric manufacturing, vol. 4, no. 3, 2021.

[3] Sy Nguyen-Van and Minh-Quang Tran, "Steady-State and Time-History Analysis for A Spatial Cable-Driven Parallel Robot," iRobotics Journal, vol. 4, no. 2, 2021.

[4] Huang, G., Liu, Z., Van Der Maaten, L. and Weinberger, K. Q., "Densely connected convolutional networks," In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4700-4708, 2017.

[5] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. and Chen, L. C., "Mobilenetv2: Inverted residuals and linear bottlenecks," In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4510-4520, 2018.

[6] K. Muhammad, S. Khan, M. Elhoseny, S. Hassan Ahmed and S. Wook Baik, "Efficient Fire Detection for Uncertain Surveillance Environment," in IEEE Transactions on Industrial Informatics, vol. 15, no. 5, pp. 3113-3122, May 2019.

[7] M. Jain, G. Kaur, M. P. Quamar and H. Gupta, "Handwritten Digit Recognition Using CNN," 2021 International Conference on Innovative Practices in Technology and Management (ICIPTM), Noida, India, 2021, pp. 211-215.

[8] Eclipse Mosquitto, https://github.com/eclipse/mosquitto.

[9] Igus delta robot, https://www.igus.com.tw/info/robotics-delta-robot.

[10] De Campos, Teófilo Emídio, Bodla Rakesh Babu, and Manik Varma, "Character recognition in natural images," 2009.