sustainability

MDPI

Article

An LSTM Based Generative Adversarial Architecture for Robotic Calligraphy Learning System

Fei Chao [1,2], Gan Lin [1], Ling Zheng [1,*], Xiang Chang [2], Chih-Min Lin [3], Longzhi Yang [4] and Changjing Shang [2]

[1] School of Informatics, Xiamen University, Xiamen 361005, China; fchao@xmu.edu.cn (F.C.); 13796637199@163.com (G.L.)
[2] Department of Computer Science, Aberystwyth University, Aberystwyth, Ceredigion SY23 3DB, UK; xic9@aber.ac.uk (X.C.); cns@aber.ac.uk (C.S.)
[3] Department of Electrical Engineering, Yuan Ze University, Taoyuan 32003, Taiwan; cml@saturn.yzu.edu.tw
[4] Department of Computer and Information Sciences, Northumbria University, Newcastle upon Tyne NE1 8ST, UK; longzhi.yang@northumbria.ac.uk
* Correspondence: liz5@xmu.edu.cn; Tel.: +86-592-2580168

Received: 22 September 2020; Accepted: 16 October 2020; Published: 31 October 2020

Abstract: Robotic calligraphy is a very challenging task for the robotic manipulators, which can sustain industrial manufacturing. The active mechanism of writing robots require a large sized training set including sequence information of the writing trajectory. However, manual labelling work on those training data may cause the time wasting for researchers. This paper proposes a machine calligraphy learning system using a Long Short-Term Memory (LSTM) network and a generative adversarial network (GAN), which enables the robots to learn and generate the sequences of Chinese character stroke (i.e., writing trajectory). In order to reduce the size of the training set, a generative adversarial architecture combining an LSTM network and a discrimination network is established for a robotic manipulator to learn the Chinese calligraphy regarding its strokes. In particular, this learning system converts Chinese character stroke image into the trajectory sequences in the absence of the stroke trajectory writing sequence information. Due to its powerful learning ability in handling motion sequences, the LSTM network is used to explore the trajectory point writing sequences. Each generation process of the generative adversarial architecture contains a number of loops of LSTM. In each loop, the robot continues to write by following a new trajectory point, which is generated by LSTM according to the previously written strokes. The written stroke in an image format is taken as input to the next loop of the LSTM network until the complete stroke is finally written. Then, the final output of the LSTM network is evaluated by the discriminative network. In addition, a policy gradient algorithm based on reinforcement learning is employed to aid the robot to find the best policy. The experimental results show that the proposed learning system can effectively produce a variety of high-quality Chinese stroke writing.

Keywords: robotic calligraphy system; robotic learning; motion planning; Long Short-Term Memory network; adversarial learning

1. Introduction

Robot is becoming an important role in improving efficiency of recycling and sustaining industrial manufacturing [1]. Furthermore, robotic manipulators show their advantages in garbage sorting, waste removal, component disassembling, and so on [2–4]. The writing of Chinese characters is a very task for robotic manipulators since a single Chinese character is formed by orderly organizing a set of stokes in a certain structure [5]. This structural complexity of Chinese character makes robotic calligraphy often to be used as a test bed for control method evaluation.

Calligraphic robots are built to learn the ways of calligraphers' writing and then perform its own calligraphy. These kind of robots can be used to help people learn the fundamental skill of Chinese calligraphy. Furthermore, they can engage in the repair of calligraphic collections in order to protect the cultural heritages [6,7]. Traditional calligraphic robots only mimic the writing of calligraphers regarding the shape of Chinese characters [8]. This leads to the lack of aesthetic preferences in robotic calligraphy. Therefore, these traditional calligraphic robots is not capable of developing new writing styles [9]. The situation is worsened due to the limited training data set. A new framework of robotic calligraphy which allows writing robots to learn aesthetic preferences with the small size of human calligrapher samples is very meaningful.

Many learning-based approaches to robotic calligraphy have attempted to build automatic calligraphic robots. However, these methods cannot generate the correct writing sequences for Chinese strokes. There have been two classes of solutions in literature. One is to manually pre-define the robot's end joint angles for each writing action to write Chinese characters or letters [10,11]. However, such methods may require a lot of work from human engineers. The other is to use the learning from demonstration (LfD) approach [12] and imitation learning method [5]. This type of methods do not need an understanding of the control or programming model of robots. However, it requires a lot of labour costs and possesses the poor generalization ability.

Furthermore, many scientists have tried to use generative adversarial nets (GANs) combining with other main machine learning techniques to find writing sequence information. For example, Chao et al. [13] used a GAN-based method to produce stroke trajectories. Although this method can realize the writing of various strokes, the writing sequence of strokes was generated in accordance with the rules predefined by humans. We noticed that the Long Short-Term Memory (LSTM) networks [14] is effective for solving time series problems. Two groups of researchers: Gregor et al. [15] and Im et al. [16] attempted to achieve the sequential painting by using the LSTM network. In the field of robotics, Rahmatizadeh et al. [12] tried to use GAN to transform an input image into a low-dimensional space and use LSTM to predict their robot's each joint value. However, all of these methods must require the massive training data to obtain action sequence information.

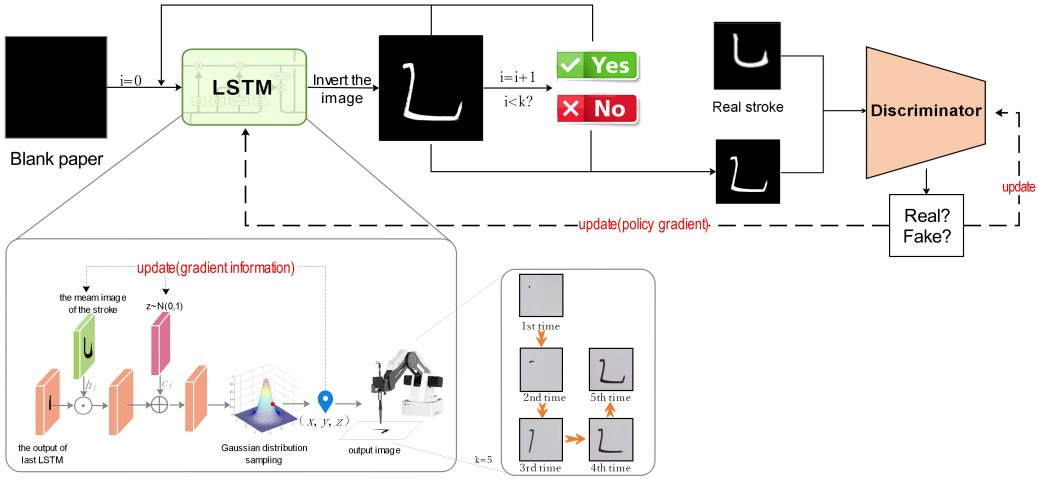
To beat the above challenges, we introduce an LSTM network into a GAN-based robotic calligraphy system [13], so as to implement an LSTM-based generative adversarial architecture. In this work, the generator network inside a GAN is replaced by an LSTM. Thus, within a single generation process, the LSTM network contains multiple loops, each of which generates a new trajectory point. A calligraphic robot then uses the point to write a segment of a stroke. The written stroke in an image format is taken as input to the next loop of the LSTM network, until the whole stroke is finally written. Additionally, a reinforcement learning algorithm is adopted by using the output of a discriminator network as a reward for training the LSTM network. The main contribution of this work is that in the absence of the robot motion trajectory dataset, the generative adversarial architecture can convert the pixel stroke image to the vector trajectory of the controllable robot, so that the robot can write high-quality Chinese character strokes and finally the pixel image information can be used to control the robot. The rest of this article is organized as follows. Section 2 details the calligraphy robot's learning system. Section 3 specifies the experimental setup and discusses the experimental results. Section 4 concludes the paper and gives perspectives of future work.

2. Proposed Framework

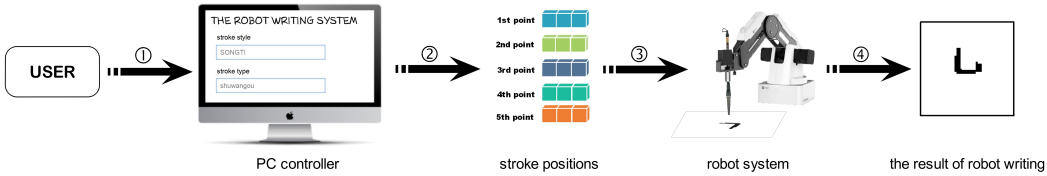
2.1. Framework Architecture

Figure 1a shows the training procedures of the proposed architecture for the robotic calligraphy system. The architecture consists of an LSTM-based stroke generation module and a convolutional neural network (CNN)-based discriminator module. The generation module produces the probability distribution of the stroke points of the strokes in sequence. The discriminator determines whether an input image is real (training data) or fake (written by the robot). Then, the generative adversarial

training scenario is used to train the entire architecture. However, since a robot system participates in the training process, the error back-propagation method of the traditional GAN cannot be applied for the architecture. To solve this problem, with reference to our previous work [13], the policy gradient method of reinforcement learning is employed to train the system.



(**a**) The training procedures of robotic calligraphy systems.



(**b**) The real operations of robotic calligraphy systems for users.

Figure 1. The training procedures and the real operations of robotic calligraphy systems.

Policy gradient are normally used to solve reinforcement learning problems. The methods based on policy gradient target at modelling and optimizing the policy directly while the goal of reinforcement learning is to encourage the agent to obtain optimal rewards. The policy is often modelled with a parameterized function with respect to θ and its mathematical expression is $\pi_\theta(a|s)$ where a are actions while s are observations.

In the stroke generation module, the input of the LSTM is a blank image. Then, the robot obtains information of the stroke position from the output of the LSTM by using Gaussian sampling. Afterwards, the robot uses the inverse kinematics calculation to convert the stroke position information into the manipulator's joint values. The robot uses this mechanical arm joint value to continue writing the stroke by linking the last point of the previous loop to the new point of the current loop. The robot captures an image of the current stroke with a camera and transmits the image to the next loop of the LSTM network. This process is repeated until all the points are generated.

Figure 1b illustrates the usage of the trained robotic calligraphy systems. Only the LSTM-based generation module is used in the system operation. A user first inputs a stroke type and the stroke style into the generation module. The module then generates all the robot joint values of a stroke and the robot writes out the whole stroke in turn. A detailed description of the implementation of the three modules in the framework is given below.

2.2. Stroke Generation Module

The stroke generation module is implemented using a LSTM network for a robot. An example of using five-epochs LSTM networks in dealing with the stroke generation is shown in Figure 2. The LSTM

network generates a probability distribution at each loop. The robot obtains a three-dimensional coordinate value, M_i, by the sampling on this distribution. The robot subsequently uses inverse kinematics to convert the stroke position to its robot joint value. The robot needs to connect the previous trajectory point to the current trajectory point on the drawing board until all the trajectory values of the vector $M = [M_0, M_1, ..., M_{k-1}]$ are obtained. The number of loops, k, of the LSTM network is preset according to the complexity of the strokes. For example, for simple strokes, the LSTM network only undergoes two loops. In other words, the LSTM outputs two coordinate values, and then, the robot connects the two coordinate values in a sequence. For complex strokes, the number of epochs of the LSTM network is set to a larger value. A complex stroke requires the LSTM network to undergo five loops. In this case, the robot obtains a trajectory vector $M = [M_0, M_1, ..., M_4]$, which means that the robot needs to write five times in succession to complete a stroke.

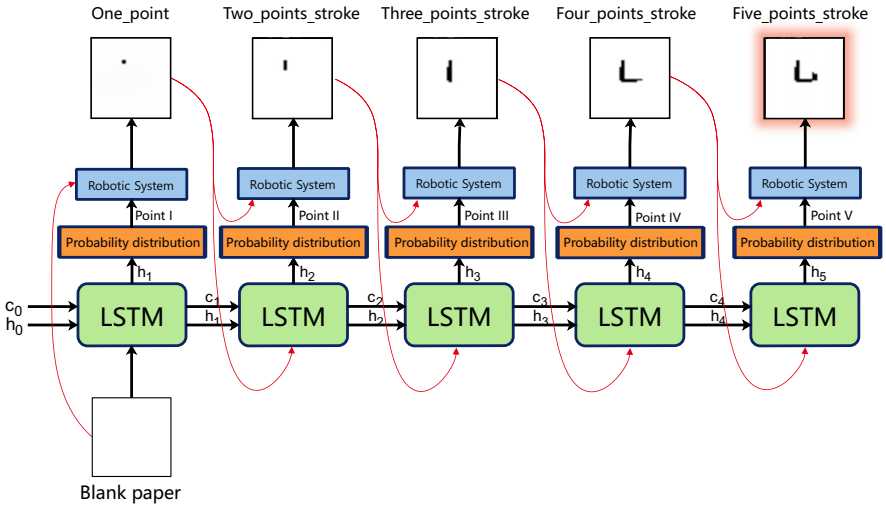


Figure 2. An example of the stroke generation module using five epochs of the LSTM networks.

The LSTM network used in this work is a 60-dimensional single hidden layer cyclic neural network for all strokes. In each loop of the LSTM, the input of the LSTM is labeled as p_{i-1}. In the first loop of the LSTM, the input is a 28×28 pixel blank vector, p_0. The image sample is averaged as the global feature of the network and set to an initial value, h_0, of the LSTM hidden layer. C_0 is a random vector of 28×28 dimensions also as the global feature of the network. The input of the ith loop of the LSTM network is a set of 28×28 pixel vectors p_{i-1}, h_{i-1} and c_{i-1} while the outputs are h_i and c_i, which are formulated as follows:

$$h_i, c_i = LSTM(p_{i-1}, h_{i-1}, c_{i-1}), i \in (0, k] \tag{1}$$

where h_i is used to predict the mean, μ_i, of the Gaussian distribution of the three-dimensional coordinates through a fully connected layer. μ_i is defined as follows:

$$\mu_i = sigmoid(f(h_i)), i \in (0, k] \tag{2}$$

where $f(\cdot)$ represents the two-layer full connection layer of the neural network. The sigmoid function is used to map variables between 0 and 1.

The variance of Gaussian distribution is fixed on the identity matrix, E, with a diagonal of 1. The sampling on Gaussian distribution, $N(x|\mu, E)$, is used by the robotic arm to generate 3-dimensional coordinates $M_i = (x_i, y_i, z_i)$ that need to be written. Represented by:

$$N(x|\mu, E) = \frac{1}{(2\pi)^{\frac{D}{2}}} \frac{1}{|E|^{\frac{1}{2}}} exp\{-\frac{1}{2}(x-\mu)^T E^{-1}(x-\mu)\} \tag{3}$$

$$M_i \sim t \times N(x|\mu, E) \tag{4}$$

where t is the maximum value of $[28, 28, 4]$. N is the Gaussian distribution.

Figure 3 shows the experimental system and the figuration of the robot. The robotic system used in this experiment includes a three-degree-of-freedom robot arm, a camera, and a writing board. The tip of the soft pen is mounted on the arm and operated in the working range of the arm. l denotes a mechanical linking rod, (x, y, z) denotes the coordinate axis of the robot, and J denotes the steering gear of the robotic arm. The robot converts the three-dimensional coordinate point t_i into three joint values $\theta_i = (\theta_1, \theta_2, \theta_3)$ of the robot by inverse kinematics. The camera is used to capture the completed characters written on the board and the captured images are sent back to the neural network afterwards. The specific calculation method is as follows:

$$\theta_1 = \arctan \frac{y_i}{x_i} \tag{5}$$

$$\theta_2 = \pi - \arccos\left(\frac{(l_2 - l_3 \cos\theta_3) \cdot (z_i - l_1 + l_4) - d_i l_3 \sin\theta_3}{(l_2 - l_3 \cos\theta_3)^2 - (l_3 \sin\theta_3)^2}\right) \tag{6}$$

$$\theta_3 = \arccos\left(\frac{l_2^2 + l_3^2 - (z_i - l_1 + l_4) - d_i^2}{2 l_2 l_3}\right) \tag{7}$$

$$\theta_i = T(M_i) \tag{8}$$

where $d_i^2 = x_i^2 + y_i^2$ and $T(\cdot)$ represents the transformation process of inverse kinematics.

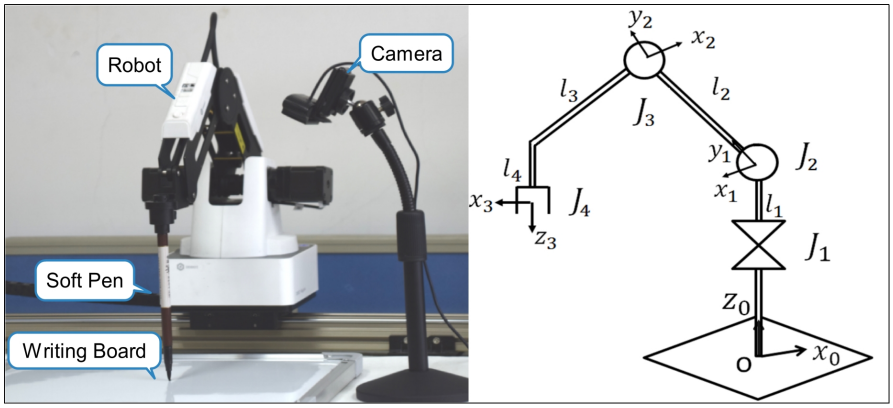


Figure 3. The hardware of the proposed framework and the structure of the robot.

The robot continues to write the stroke by following the previous trajectory point generated in the last cycle, i.e., connecting the coordinate point M_{i-1} of the last loop to the coordinate point M_i of this cycle. If it is the first loop of LSTM, only the coordinate point is generated. In addition, the camera next to the robot captures, binarizes, and trims the written result to an image with 28×28 pixels. This process is expressed as $W(\cdot)$. The image is used as the input p_i of the LSTM for the next loop. The writing result of the robot system is expressed as:

$$p_i = \begin{cases} W(T(M_i)), i = 0 \\ W(T(M_{i-1}), T(M_i)), i > 0 \end{cases} \tag{9}$$

Finally, the output of the generation module is as follows:

$$p_k = G(p_0, h_0, c_0) \tag{10}$$

2.3. Stroke Discrimination Module

The stroke discrimination module is built on a CNN network. The input of the stroke discrimination module is divided into two categories. The first type is the image X_{fake}, the writing result of a robot taken by a camera, which is binarized and trimmed. The second type is the real stroke image X_{real}. The size of the input image layer is set to 28×28 while the size of the network's output is 1. The output predicts the probability of the data distribution of X from the real image, X_{real}, or the image generated by the robot, X_{fake}. The hidden layer of the CNN network consists of two convolutional layers and two fully connected layers. The network's structure is shown in Figure 4. The image is up-sampled at the convolutional layer to 320 dimensions and passed through the fully connected layer to produce the one-dimensional output.

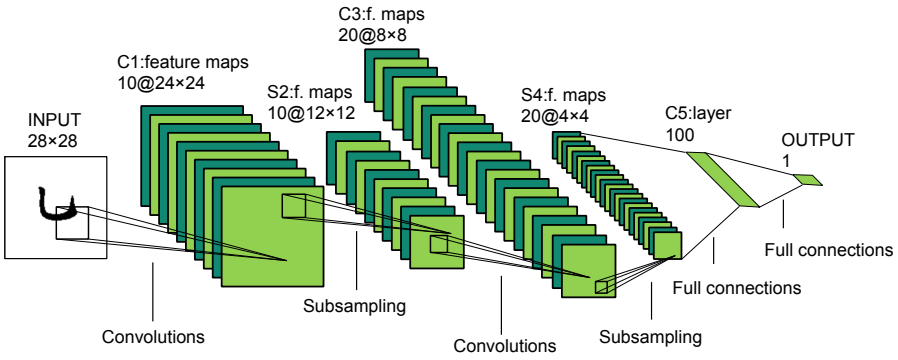


Figure 4. D network structure diagram.

2.4. Training Algorithm

The objective function of this architecture is expressed as:

$$\begin{aligned} \min_G \max_D V(D,G) &= E_x[sigmoid(D(x))] \\ &+ E_{h_0,c_0}[sigmoid(1 - D(G(p_0,h_0,c_0)))] \end{aligned} \tag{11}$$

where $D(\cdot)$ represents the output of a CNN network, $G(\cdot)$ represents the output of a LSTM network and $E[\cdot]$ represents the expected value of the LSTM network. The target of the CNN network is expressed as the following loss function:

$$\begin{aligned} D_{loss} = &- E_x[sigmoid(D(x))] \\ &- E_{h_0,c_0}[sigmoid(1 - D(G(p_0,h_0,c_0)))] \end{aligned} \tag{12}$$

$D(x)$ represents the score of the CNN network for the real stroke sample, and $D(G(p_0,h_0,c_0))$ represents the score of the CNN network for the stroke sample generated by the LSTM network, ranging from 0 to 1.

In order to make the LSTM network obtain higher rewards, the LSTM network must guarantee the quality of each trajectory point. Therefore, the goal of the LSTM network is to increase the occurrence probability of the trajectory with a high score in the CNN network. The loss function of the LSTM network is as follows:

$$G_{loss} = E_{h_0,c_0}[(\prod_{i=0}^{k} \log_{prob}(LSTM(p_i,h_i,c_i))) \cdot D(G(p_0,h_0,c_0))] \tag{13}$$

where $\log_{prob}(LSTM(p_i, h_i, c_i))$ represents the probability of the output trajectory points of the LSTM for the ith loop; $\prod_{i=0}^{k} \log_{prob}(LSTM(p_i, h_i, c_i))$ represents the occurrence probability of the stroke calculated by multiplying the likelihood probabilities of all the trajectory points in the stroke. $D(G(p_0, h_0, c_0))$ represents the output from the CNN network, of which values are ranged from 0 to 1.

The gradient of the objective function $J(\theta)$ and the LSTM network parameter, θ, are derived by:

$$\begin{aligned} &\nabla_\theta J(\theta) = E_{h_0,c_0}[\\ &\nabla_\theta(\prod_{i=0}^{k} \log_{prob}(LSTM(p_i, h_i, c_i))) \cdot D_\theta(G_\theta(p_0, h_0, c_0))] \end{aligned} \tag{14}$$

Since the expectation $E[\cdot]$ can be approximated by sampling, the parameters of the LSTM network are updated in the following ways:

$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta) \tag{15}$$

where α is the learning rate. The training procedures are presented in pseudo code listed at Algorithm 1.

Algorithm 1 Training Procedure Pseudocode

Require: Real stroke images database $\boldsymbol{X}_{real}$, mean of real stroke images $\boldsymbol{h}_0$, random number $\boldsymbol{c}_0$, blank vector $\boldsymbol{p}_0$.

```
Initialize LSTM and CNN network with random weights;
repeat
    for g-step do
        Input p_0 , h_0 , c_0 into LSTM;
        for i in 0 : k do
            Use Equation (4) to sample a trajectory point M_i;
            Robot writes the trajectory, which is captured as a input image for the next cycle;
        end for
        Update LSTM parameters via Equation (15);
    end for
    for d-step do
        Combine the new stroke images X_fake with real stroke images X_real;
        Train CNN by Equation (12);
    end for
until GAN Converges
```

3. Experimentation

3.1. Training Data

The architecture proposed above was applied to the task of robotic writing on Chinese character strokes, which is also used for system verification and evaluation. The images of the stroke training data were extracted from the Chinese character images, normalized and classified. Then, the training processes of the CNN network and LSTM network and the robot writing action were carried out, and the learning performance of the policy gradient was obtained.

First, we adopt the method proposed in [17] to automatically extract the strokes of a character. Next, the stroke images are converted into binary forms. An else CNN network is used to classify the binary-valued strokes into 31 categories, which will be stored in the database. In addition, we also calculated the mean value of all the types of stroke $S = [S_1, S_2, ..., S_m]$ as the h_0 in the LSTM to facilitate the network to learn features of the stroke style. m is the number of images in this category. h_0 is given as:

$$h_0 = \frac{1}{m} \sum_{i=0}^{m} S_i. \tag{16}$$

In this experiment, we selected six different Chinese character strokes to train and test our proposed architecture. Each class of strokes have 500 sample images. Figure 5 shows the training samples used in the experiment of the six types of stroke, each row shows one type of stroke with various variants. From Figure 5, we can see that the strokes in the same type are not exactly identical to each other. The types of strokes from top to bottom are: "horizontal stroke", "short left-falling stroke", "right-falling stroke", "vertical, turn-right and hook stroke" and "horizontal hook stroke".

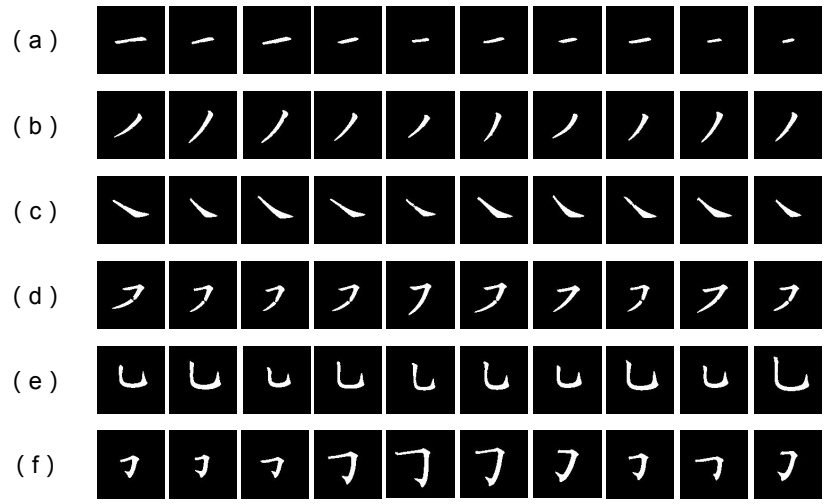


Figure 5. Illustrative training samples used in the experiment: each sub-image indicates a stroke while every single row of rows (**a**–**f**) signifies 10 variants of a specific stroke.

3.2. Training Process and Writing Results

Figure 6 shows the writing process and results of the robot for the "vertical, turn-right and hook stroke". The figure shows the three stages of the training process: (1) early stage, (2) medium stage, and (3) final stage. In every row of each stage, the images from left-to-right show the robot's writing results after each loop of the LSTM. The images at far right are the image to be passed to the discriminating module after color inversion. Results from the other five strokes show a similar training process. Before passing to the LSTM of the next loop, the image was binarized and reversed to ensure the consistency of the input image of each LSTM.

In the early stage of training, the writing results of strokes were chaotic; the shapes were not close to the target stroke. In the medium stage of training, the written results got closed to the target stroke. However, a large detailed difference between the results and target image still existed. In contract, the final stage of writing shows a high level of quality, and the shapes produced at this stage are very similar to the target strokes.

In Figure 6, we also noticed that: during the training process, some stroke's writing sequence was in accordance with human writing habits even without the stroke's intermediate sequence process. This proves that it is possible to write strokes that conform to human writing habits even when LSTM does not have the intermediate sequence process. However, not all the strokes were written by following human's writing sequence; more future efforts will focus on this problem.

Figure 7 illustrates the evaluation results of the LSTM network's output for the "vertical, turn-right and hook stroke". The starting value of the LSTM network is about 0.7, that is, since the LSTM network has not been fully trained, the CNN network can have a high possibility to determine whether a stroke is from the robot or not. As the LSTM network continued to be strengthened, the loss function of the CNN network gradually decreases and becomes stable. Since the robot participated in the generation process of the LSTM network, some unexpected errors still existed in the written results; therefore, the errors prevented the CNN network from achieving the standard loss. However, in this experiment, even the loss cannot achieve the lowest value, the written results can still be accepted by human users.

Figure 8 shows the final writing results of the six strokes, all of which are trimmed but not binarized. The variety of results was obtained based on the random vector of the c_0 in the LSTM. The results show that: the trajectories of the same stroke are different, and the part of results are written in accordance with human writing habits. Meanwhile, we found that the trajectories of some strokes are very close to those written by humans. For example, the first stroke in (e) and the last stroke in (f) owned beautiful appearances.

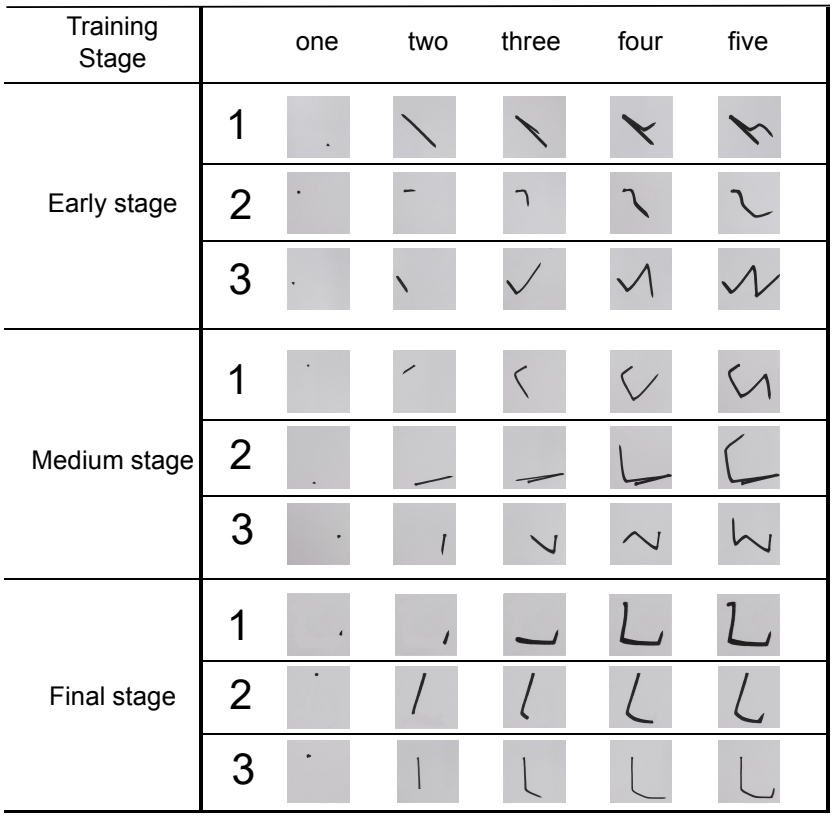


Figure 6. The writing process and results of the robot with vertical hook hooks.

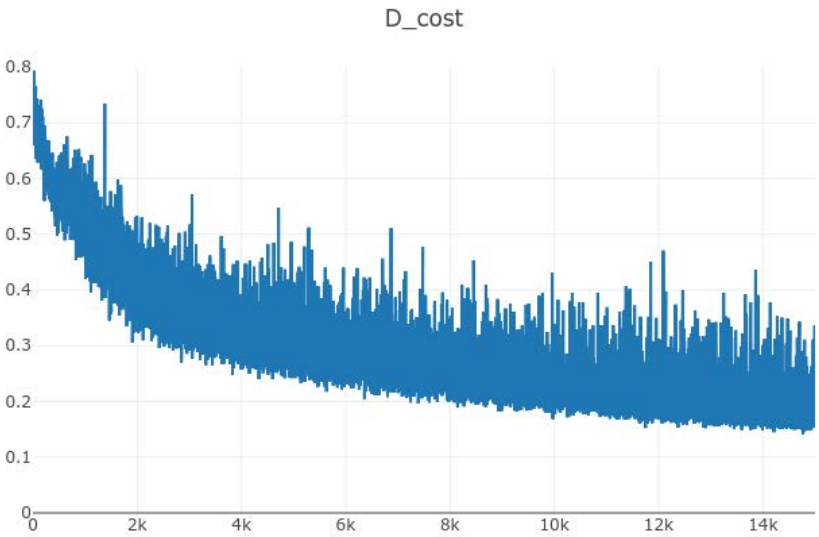


Figure 7. The training epochs evaluation results of D network's output for vertical, turn-right and hook stroke.

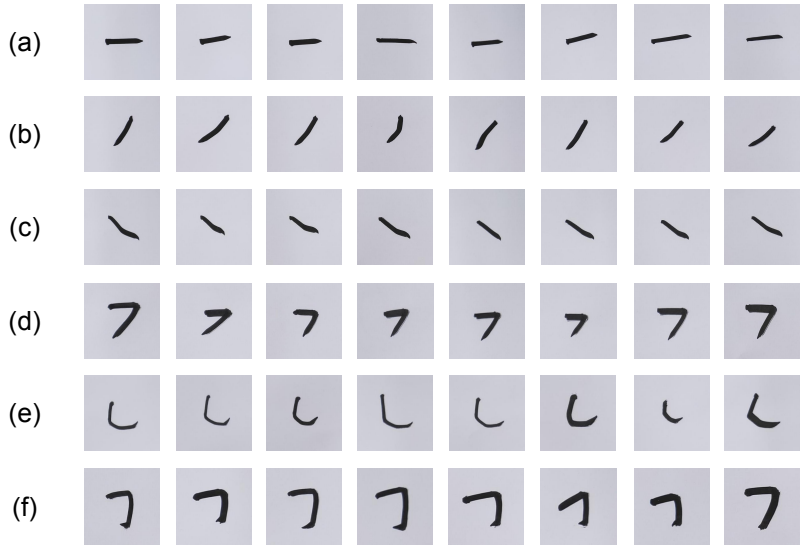


Figure 8. The results of the six strokes written by the robot: each row of rows (**a**–**f**) is a type of stroke with 10 various calligraphic styles and each sub-image is the writing of a single one stroke.

4. Conclusions

This paper proposes a system of robotic calligraphy based on the LSTM and generative adversarial network. The system enabled the robot to learn and generate trajectory writing motions of Chinese character strokes independently. Without the writing sequence information, the method can realize the conversion from input stroke images to the robot motion sequences, thereby enabling the robot to write high-quality Chinese character strokes. Meanwhile, the robot system can write some strokes that conform to the human writing sequences without any human pre-defined rules. Experimental results based on six strokes demonstrated the effectiveness of the proposed method with several results attaining the human writing level.

Although our approach is effective, it can be improved regarding the following two aspects. First of all, our approach can only write Chinese character strokes at present. However, how to write a complete Chinese character is still a problem. It is worth further consideration. Secondly, the proposed work cannot guarantee all the strokes can have a correct writing sequence; thus, more future efforts are required.

Author Contributions: Conceptualization, F.C. and G.L.; methodology, F.C. and L.Z.; software, G.L.; validation, X.C., and C.-M.L.; formal analysis, L.Y.; investigation, C.S.; resources, G.L.; data curation, X.C.; writing—original draft preparation, F.C. and G.L.; writing—review and editing, F.C., L.Z., and L.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (No. 61673326, 61673322, and 91746103), the Fundamental Research Funds for the Central Universities (No. 20720190142), and the European Union's Horizon 2020 research and innovation program under the Marie Sklodowska-Curie grant agreement No. 663830.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Pujol, F.A.; Tomás, D. Introducing Sustainability in a Robotic Engineering Degree: A Case Study. *Sustainability* **2020**, *12*, 5574. [CrossRef]
2. Rivera, R.G.; Alvarado, R.G.; Martínez-Rocamora, A.; Auat Cheein, F. A Comprehensive Performance Evaluation of Different Mobile Manipulators Used as Displaceable 3D Printers of Building Elements for the Construction Industry. *Sustainability* **2020**, *12*, 4378. [CrossRef]

3. Zhang, Q.; Li, H.; Wan, X.; Skitmore, M.; Sun, H. An Intelligent Waste Removal System for Smarter Communities. *Sustainability* **2020**, *12*, 6829. [CrossRef]
4. Gualtieri, L.; Palomba, I.; Merati, F.A.; Rauch, E.; Vidoni, R. Design of Human-Centered Collaborative Assembly Workstations for the Improvement of Operators' Physical Ergonomics and Production Efficiency: A Case Study. *Sustainability* **2020**, *12*, 3606. [CrossRef]
5. Chao, F.; Huang, Y.; Zhang, X.; Shang, C.; Yang, L.; Zhou, C.; Hu, H.; Lin, C.M. A robot calligraphy system: From simple to complex writing by human gestures. *Eng. Appl. Artif. Intell.* **2017**, *59*, 1–14. [CrossRef]
6. Zeng, H.; Huang, Y.; Chao, F.; Zhou, C. Survey of robotic calligraphy research. *CAAI Trans. Intell. Syst.* **2016**, *11*, 15–26. [CrossRef]
7. Jian, M.; Dong, J.; Gong, M.; Yu, H.; Nie, L.; Yin, Y.; Lam, K. Learning the Traditional Art of Chinese Calligraphy via Three-Dimensional Reconstruction and Assessment. *IEEE Trans. Multimed.* **2020**, *22*, 970–979. [CrossRef]
8. Gao, X.; Zhou, C.; Chao, F.; Yang, L.; Lin, C.M.; Xu, T.; Shang, C.; Shen, Q. A data-driven robotic Chinese calligraphy system using convolutional auto-encoder and differential evolution. *Knowl. Based Syst.* **2019**, *182*, 104802, [CrossRef]
9. Gan, L.; Fang, W.; Chao, F.; Zhou, C.; Yang, L.; Lin, C.M.; Shang, C. Towards a Robotic Chinese Calligraphy Writing Framework. In Proceedings of the 2018 IEEE International Conference on Robotics and Biomimetics (ROBIO), Kuala Lumpur, Malaysia, 12–15 December 2018; pp. 493–498.
10. Zhang, X.; Li, Y.; Zhang, Z.; Konno, K.; Hu, S. Intelligent Chinese calligraphy beautification from handwritten characters for robotic writing. *Vis. Comput.* **2019**, *35*, 1193–1205. [CrossRef]
11. Li, J.; Min, H.; Zhou, H.; Xu, H. Robot Brush-Writing System of Chinese Calligraphy Characters. In *Intelligent Robotics and Applications*; Yu, H., Liu, J., Liu, L., Ju, Z., Liu, Y., Zhou, D., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 86–96.
12. Rahmatizadeh, R.; Abolghasemi, P.; Bölöni, L.; Levine, S. Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018; pp. 3758–3765.
13. Chao, F.; Lv, J.; Zhou, D.; Yang, L.; Lin, C.; Shang, C.; Zhou, C. Generative Adversarial Nets in Robotic Chinese Calligraphy. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018; pp. 1104–1110. [CrossRef]
14. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
15. Gregor, K.; Danihelka, I.; Graves, A.; Rezende, D.J.; Wierstra, D. DRAW: A Recurrent Neural Network for Image Generation. *arXiv* **2015**, arXiv:1502.04623.
16. Im, D.J.; Kim, C.D.; Jiang, H.; Memisevic, R. Generating images with recurrent adversarial networks. *arXiv* **2016**, arXiv:1602.05110.
17. Lian, Z.; Zhao, B.; Xiao, J. *Automatic Generation of Large-Scale Handwriting Fonts via Style Learning*; Siggraph Asia 2016 Technical Briefs (SA'16); ACM: New York, NY, USA, 2016; pp. 1–4, [CrossRef]