CalliRewrite: Recovering Handwriting Behaviors from Calligraphy Images without Supervision Supplemental Material

Yuxuan Luo[1*], Zekun Wu[1*], Zhouhui Lian[1†]

I. Coarse Sequence Extraction

A. Architecture Introduction

Our model for coarse sequence extraction utilizes the General Virtual Sketching Framework (GVS) [1] (visualized in Figure 1a). It employs a convolutional neural network (CNN) coupled with a Long Short-Term Memory (LSTM) model to decompose line-art images into quadratic Bézier representations. At each time step t, the model processes a dynamic image patch, yielding a vector $a_t = (x_c, y_c, \Delta x, \Delta y, w, \Delta s, p)_t$. This vector is transformed into quadratic Bézier strokes for rendering and state updating. Here, $w \in [0, 1]$ represents stroke width, and $\Delta s \in [0, k]$ (where $k > 1$) denotes the window scaling factor.

The dynamic image patch is generated through a dynamic window and aligned cropping approach. To address misalignment concerns, Mo et al. introduced an aligned cropping operator inspired by He et al. [2]. This operator subdivides the window into spatial bins using the window size W_{t-1} and a resampling size $W_r = 128 \times 128$. Points within each bin are sampled via bilinear interpolation, and averaged values within each bin form the final fixed-size image patch W_r, effectively mitigating quantization artifacts.

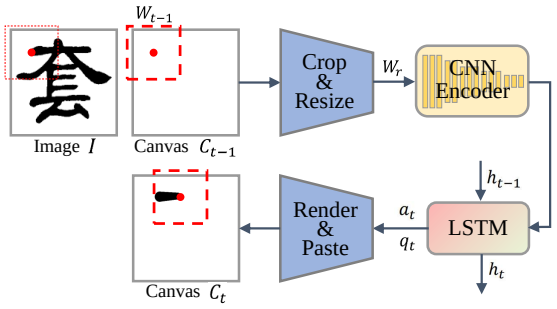
The differentiable pasting operation also relies on bilinear image interpolation. The key distinction between these modules lies in their utilization of 2D interpolations under distinct coordinate systems, necessitating a coordinate system transformation. For further insights, refer to Figure 1b.

B. Unsupervised Loss Design

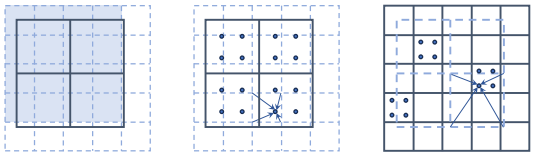
The existing GVS loss functions, including the raster loss $\mathcal{L}_{\text{ras}}$, out-of-bound penalty $\mathcal{L}_{\text{out}}$, and regularization loss $\mathcal{L}_{\text{reg}}$, do not fully address human-like writing behaviors and struggle with thicker strokes in Chinese calligraphy. To enhance continuity and stroke sequencing, we augment finetuning with 1000 glyphs sourced from KaiTi-GB2312 [3] and KanjiVG [4]. We also introduce smooth and angle loss to facilitate learning comprehensive stroke decomposition, adhering to more natural writing order conventions.

Smoothness Loss, as detailed in our paper, addresses the intersection problem. Here, we provide an additional example to illustrate its rationale, along with the necessity of applying a scaling factor of 0.5 before the smoothness loss.

[1]Wangxuan Institute of Computer Technology, Peking University
[*]Equal contribution
[†]Corresponding author: lianzhouhui@pku.edu.cn



(a) Coarse Sequence Extraction pipeline.



(b) Different types of cropping and pasting operations.

Fig. 1: (a): Model pipeline includes aligned cropping, CNN-based encoder, LSTM and differentiable pasting. (b): Left: Quantization leads to misalignment; Mid: Aligned Cropping uses sampling and bilinear interpolation; Right: Differentiable pasting, similar to Aligned Cropping.

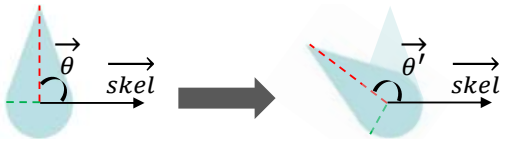
Angle Loss has been a focal point in cognitive science, particularly regarding writing order. In the 1980s, an oscillation model [5] proposed for cursive Latin writing depicted angular motion within 0° to 60° range, either clockwise or counterclockwise. Thomassen et al. [6] systematically studied stroke order in basic geometric patterns, identifying five common rules in cursive scripts: Threading, Starting from the far left, Anchoring, Starting with vertical strokes, and Beginning strokes from the top. Teulings et al. [7] discovered orthogonal axes of wrist and finger movement, averaging +45 and -45 degrees relative to the baseline. Li et al. [8] divided Chinese strokes into seven ranges, whereas Neo et al. [9] categorized stroke directions into eight quadrants. Thomassen et al. [10] discussed directional preference development, suggesting left-handed individuals may exhibit distinct writing behaviors.

We formulate the angle loss following the right-handed hypothesis and the prevalent left-to-right, up-to-down convention. Each initial direction is constrained within $+75^\circ$ to -165° range. Instances exceeding this boundary undergo cosine similarity ($\mathcal{C}$) computation between stroke direction

$\vec{S}_t = (\overrightarrow{\mathcal{O}_0\mathcal{O}_2})_t$ and a 135° normal vector $\vec{a} = (-\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2})$.

$$\mathcal{L}_{\text{ang}} = \frac{1}{T}\sum_{t=1}^{T}\begin{cases} \mathcal{C}(\vec{S}_t, \vec{a}), & \mathcal{C}(\vec{S}_t, \vec{a}) \geq 0.5 \\ 0, & \mathcal{C}(\vec{S}_t, \vec{a}) < 0.5.. \end{cases}$$

(a) $\mathcal{C} \in (-1, -0.7]$.



(b) $\mathcal{C} \in (-0.7, 0.96]$.

(c) $\mathcal{C} \in (0.96, 1]$.

Fig. 2: In the droplet-shaped calligraphy brush model, the dynamics are modeled. The angle of the brush tip changes according to three different situations corresponding to the increasing motion changes of the brush.

II. Modeling Writing Utensils

We undertake the modeling of the fude pen, flat-tip marker, and calligraphy brush in our virtual and physical environment. In this section, we delineate our modeling procedures and explain the calibration process for robotic demonstration.

A. Geometry Properties

We describe geometric properties of writing instruments using parameters r, l, and θ. The fude pen is represented by an ellipse with semi-minor axis r and semi-major axis l. The flat-tip marker is depicted by a rectangle with half-width r and half-length l. The inclination angle of these shapes is denoted by θ. The fude pen is modeled as soft body while the marker is rigid.

Modeling the calligraphy brush is worth discussing. Previous graphics-related studies have modeled the complex system with 3D meshes or hair clusters [11–13]. However, following Xie et al. [14], our approach centers on the two-dimensional contact shape for approximation. Here, r represents the circular belly's radius, l denotes the distance from the pen tip to the circle's center, and θ signifies the counterclockwise rotation angle to the pen tip in the global coordinate system. Through measurement, we establish the $l - r$ relationship using least square approximation:

$$l = 0.003 + 2.021 \times r.$$

This relation, defined in the tool property files, can be adjusted through calibration.

B. Defining Motion Dynamics

Utensil dynamics are shaped by the current geometric states and action direction. Formally, this dynamic behavior can be expressed as $\theta_{t+1} = \{(\theta_t, \vec{s_t}, \vec{a_t})$, where $\vec{s_t}$ encapsulates the current geometric properties r, l, and θ, while

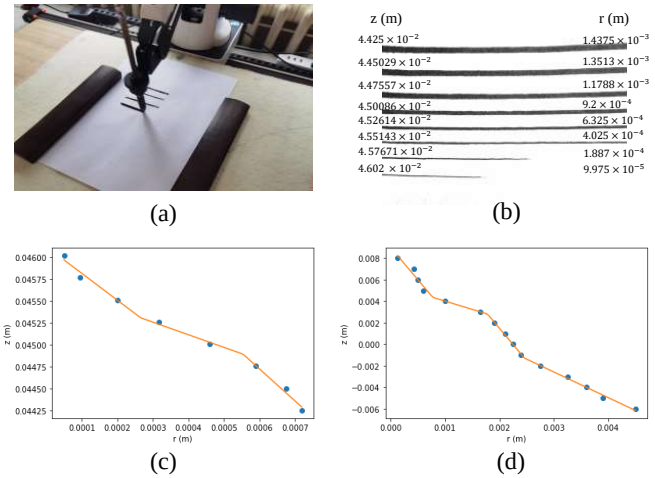


Fig. 3: Calibration pipeline. (a): Running calibration script on Dobot robotic arm. (b): Measuring radius and record coresponding z-axis value. (c): Fitting curve of the fude pen. (d): Fitting curve of the calligraphy brush.

$\vec{a_t} = (\delta_x, \delta_y)_t$ or $\vec{a_t} = (\delta_x, \delta_y, \phi)_t$ denotes the action at timestep t, with DoF varying depending on whether the tool is soft or rigid.

For the fude pen, the dynamic function corresponds to the two-dimensional action direction, i.e. $\theta_t = \arctan(\delta_y/\delta_x)$. For the rigid flat-tip marker, the differential dynamics $\Delta\theta = \theta_t - \theta_{t-1}$ is controlled directly by the RL algorithm.

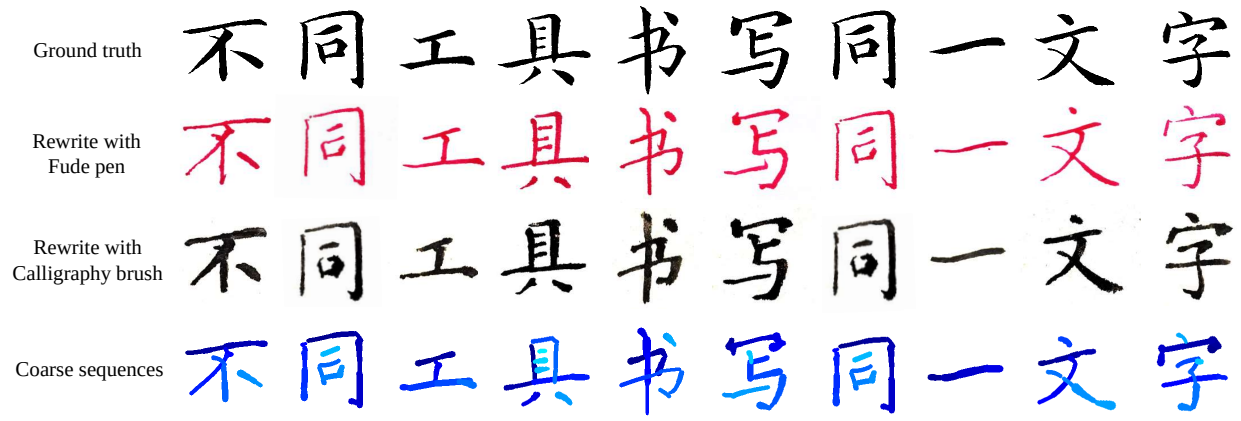
Modeling the movement of calligraphy brush is a tough task. Here we provide a crafted approximation: we derive three different situations of brush dynamics according to the angle between the trajectory and the tip. The corresponding formulas are as follows:

$$\theta_{t+1} = \theta_t + \begin{cases} 10° \cdot \mathbf{S}, & \mathbf{C} \in [-1, -0.7] \\ \frac{5° \cdot \mathbf{S} \cdot (1+\mathbf{C})}{r}, & \mathbf{C} \in (-0.7, 0.96) \\ 165° - \theta_t, & \mathbf{C} \in [0.96, 1] \end{cases}$$

Where $\mathbf{C} = \mathcal{C}(\vec{\theta_t}, \vec{skel})$, representing the cosine similarity between the tip of the brush and the motion trajectory, and $\mathbf{S}$ denotes the sine "similarity" between the two vectors.

C. Calibration

In previous sections, we modeled tool properties in our RL environment, primarily relying on human-defined specifications. The sequence finetuning process was trained within this RL environment, and the ultimately optimized sequences were demonstrated on a 3 DoF robotic arm. To address the sim-to-real gap and to accurately replicate the training results, we calibrated the relationship between tool geometry r and robotic arm height z. We devised a script to control the arm's writing with different pen strokes (r) at varying heights z on paper. By measuring and fitting piecewise linear curves, we obtained the $r - z$ relationship, depicted in Figure 3. During robotic demonstration, we employed the fitted $r - z$ curve to translate optimized trajectories into 3 DoF control sequences for reproduction.



(a) Rewriting on Chinese Characters. CalliRewrite controls the fude pen to master the handling of brush strokes in chinese characters, including the starting, ending, and pause strokes.

(b) "Rewriting on English Characters. CalliRewrite replicates variations in stroke thickness and controls the beginnings and endings of horizontal strokes with a Chinese calligraphy brush.

Fig. 4: Crafting character detail with diverse tools. Whether with a fude pen or a calligraphy brush, CalliRewrite ensures nuanced stroke variations and captures unique design elements, tailored to each utensil's characteristics.

III. Constrained Sequence Optimization with Reinforcement Learning

We define the refinement process of a coarse sequence as a constrained sequence optimization problem, employing reinforcement learning (RL) algorithms (specifically, the Soft Actor-Critic algorithm) on the discretized sequence. In each episode, the reinforcement learning agent starts from the initial state, corresponding to the beginning of the sequence, and proposes actions for each state, which are then used to update the coordinates of the sequence points after scaling. The entire optimization process iterates multiple rounds to refine the original sequence. The basic action space of the RL agent consists of two dimensions, δ_x and δ_y, representing the vectors for moving the agent from the original discretized state points. In the implementation, we convert these vectors to polar coordinates r' and θ' to represent the length and direction of movement, where $r' \in [0, 1]$. The scaling factor τ can be adjusted to increase or decrease the magnitude of the reinforcement learning algorithm's fine-tuning on the sequence. A large τ leads to a broader exploration range during the initial stages of reinforcement learning, making it more prone to going out of bounds. Conversely, a small τ tends to cause the reinforcement learning algorithm to converge to local minima, resulting in suboptimal fine-tuning of the sequence. Through experimentation, we set the default value of τ to be 0.03 to achieve a good balance between exploration and exploitation.

IV. Supplementary Experiments

In this section, we provide more qualitative experiments to discuss our method's effectiveness.

A. Same glyph with different tools

In order to assess the adaptability of our model to different writing tools, we employed both a fude pen and calligraphy brush to reproduce the same characters and compared the resulting handwritten outputs. Examples of Chinese and English calligraphy fonts were provided on a 4cm $\times$ 4cm scale, and then re-rendered using a fude pen and calligraphy brush. The effects are illustrated in Figure 4. Through comparison, it is evident that the model successfully reproduced the corresponding fonts using both tools. Furthermore, when using the fude pen, the model was able to replicate the nuances and strokes characteristic of Chinese calligraphy, while also demonstrating proficiency in handling the unique starting and ending strokes of English characters when using the calligraphy brush.

REFERENCES

[1] H. Mo, E. Simo-Serra, C. Gao, C. Zou, and R. Wang, "General virtual sketching framework for vector line art," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–14, 2021.

[2] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[3] "Kaiti-GB2312," https://www.onlinewebfonts.com/download/8a1e9fe86f7a9489ec091ec4b78af185, accessed: August 25, 2023.

[4] "KanjiVG," http://kanjivg.tagaini.net/, accessed: August 25, 2023.

[5] J. M. Hollerbach, "An oscillation theory of handwriting," *Biological cybernetics*, vol. 39, no. 2, pp. 139–156, 1981.

[6] A. J. Thomassen, R. G. Meulenbroek, and H. J. Tibosch, "Latencies and kinematics reflect graphic production rules," *Human Movement Science*, vol. 10, no. 2-3, pp. 271–289, 1991.

[7] H.-L. Teulings, "Handwriting movement control," in *Handbook of perception and action*. Elsevier, 1996, vol. 2, pp. 561–613.

[8] J. Li, W. Sun, M. Zhou, and X. Dai, "Teaching a calligraphy robot via a touch screen," in *2014 IEEE International Conference on Automation Science and Engineering (CASE)*. IEEE, 2014, pp. 221–226.

[9] C. C. Neo, E. L. M. Su, P. I. Khalid, and C. F. Yeong, "Method to determine handwriting stroke types and directions for early detection of handwriting difficulty," *Procedia Engineering*, vol. 41, pp. 1824–1829, 2012.

[10] A. J. Thomassen and H.-L. H. Tuelings, "The development of directional preference in writing movements," *Visible Language*, vol. 13, no. 3, p. 299, 1979.

[11] H. T. Wong and H. H. Ip, "Virtual brush: a model-based synthesis of chinese calligraphy," *Computers & Graphics*, vol. 24, no. 1, pp. 99–113, 2000.

[12] N.-H. Chu and C.-L. Tai, "An efficient brush model for physically-based 3d painting," in *10th Pacific Conference on Computer Graphics and Applications, 2002. Proceedings.* IEEE, 2002, pp. 413–421.

[13] S. Xu, F. C. Lau, F. Tang, and Y. Pan, "Advanced design for a realistic virtual brush," in *Computer Graphics Forum*, vol. 22, no. 3. Wiley Online Library, 2003, pp. 533–542.

[14] N. Xie, H. Hachiya, and M. Sugiyama, "Artist agent: A reinforcement learning approach to automatic stroke generation in oriental ink painting," *IEICE TRANSACTIONS on Information and Systems*, vol. 96, no. 5, pp. 1134–1144, 2013.