2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

CalliRewrite: Recovering Handwriting Behaviors from Calligraphy Images without Supervision

Yuxuan Luo[1,2*], Zekun Wu[1*], Zhouhui Lian[1†]

***Abstract*— Human-like planning skills and dexterous manipulation have long posed challenges in the fields of robotics and artificial intelligence (AI). The task of reinterpreting calligraphy presents a formidable challenge, as it involves the decomposition of strokes and dexterous utensil control. Previous efforts have primarily focused on supervised learning of a single instrument, limiting the performance of robots in the realm of cross-domain text replication. To address these challenges, we propose CalliRewrite: a coarse-to-fine approach for robot arms to discover and recover plausible writing orders from diverse calligraphy images without requiring labeled demonstrations. Our model achieves fine-grained control of various writing utensils. Specifically, an unsupervised image-to-sequence model decomposes a given calligraphy glyph to obtain a coarse stroke sequence. Using an RL algorithm, a simulated brush is fine-tuned to generate stylized trajectories for robotic arm control. Evaluation in simulation and physical robot scenarios reveals that our method successfully replicates unseen fonts and styles while achieving integrity in unknown characters. To access our code and supplementary materials, please visit our project page: https://luoprojectpage.github.io/callirewrite/.**

I. INTRODUCTION

Calligraphy robots exemplify technology and arts, advancing artificial intelligence, exploring human-robot interaction, and serving as educational tools.

This paper aims to address the gaps in limited writing utensil manipulation and stroke-level learning, which previous research has not fully explored. Besides conventional algorithms like developmental algorithm [1], clustering [2], dynamic programming, and Gaussian process [3], some researchers have also investigated deep learning techniques such as GAN [4, 5] and LSTM [6] for stylizing and controlling pen strokes. However, these approaches heavily rely on laborious annotations and heavily supervised learning, making them less effective in scenarios like understanding and parsing ancient scripts.

On the contrary, we focus on unsupervised low-resource learning, mining knowledge from plain images, and self-discovering dexterous control over various utensils. We draw inspiration from how humans break down tasks and skillfully use tools. Even when humans lack precise knowledge of stroke order, such as when children first practice writing their names or scholars transcribe newly found ancient characters, they demonstrate effective planning strategies. Moreover, a proficient artist can replicate a specific writing style with different instruments, for example, etching brush-written fonts onto a tablet with a knife. These examples highlight humans' crucial skills in task planning and motion control and are critical to robotics research.

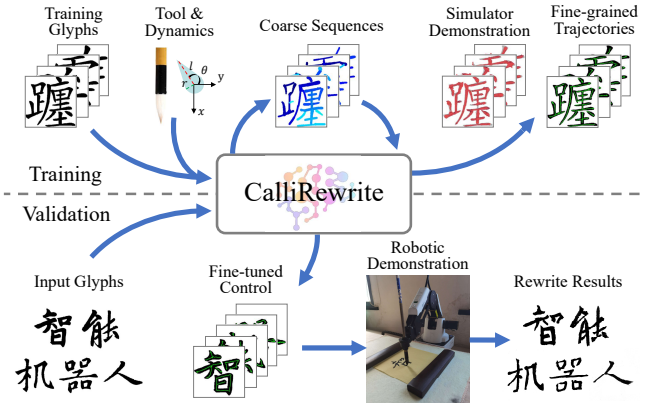


Fig. 1: Our method, CalliRewrite, enables unsupervised and tool-dependent robot calligraphy. During training, we provide only glyph images and virtual writing utensils. Our two-stage model gradually proposes and fine-tunes applicable writing trajectories conditioned on specific tools. In validation, we apply the parametric fine control on the Dobot robotic arm, which accurately redraws the characters.

This paper proposes CalliRewrite: an unsupervised approach enabling robotic arms to replicate diverse calligraphic glyphs with generalization on manipulating different writing tools. Our method employs a hierarchical structure encompassing a CNN-encoded LSTM model to deduce stroke-level orders, and a reinforcement learning (RL) pipeline to fine-tune the coarse sequences into tool-aware stylized control, controlling the brush agent with soft-actor-critic (SAC) algorithm. Utilizing self-supervised loss functions and formulating the fine-tuning into a constrained sequence optimization, we enable the model to self-discover applicable stroke orders without any annotations and discover dexterous control on different utensils, especially on Chinese paintbrush, which demands superb skill.

We devise a progressive training for low-resource stroke decomposition, pretrained on QuickDraw to learn universal redrawing and better decomposition on 1000 training glyphs. Our evaluation covers diverse calligraphy scripts: Chinese, English, Ancient Egyptian, and Tamil. Our metrics cover stroke number ratio (SNR) and Chamfer Distance, depicting stroke continuity and overall redrawn fidelity respectively. We not only carry out verification on simulations but also rewrite glyphs with a Dobot-Magician robot arm, following the generated control sequence with corresponding utensils.

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

Experiments reveal that our model surpasses other unsupervised methods in stroke segmentation and performs comparably with supervised models. Additionally, our calligraphy robot can faithfully rewrite unseen fonts or sketches and adeptly control diverse writing implements in unsupervised and low-resource scenarios.

II. RELATED WORK

Our work spans glyph comprehension and tool manipulation. Glyph comprehension involves recognizing and decomposing characters, while tool manipulation focuses on modeling utensils and learning dexterous control.

A. Glyph Comprehension

Glyph comprehension involves character reconstruction and decomposition and is challenging due to human writing's complexity and variability. Traditional models based on expert systems generated glyphs through defined norms and templates [7]. Deep learning has enabled methods via convolution neural networks (CNNs) and Generative Adversarial Networks (GANs), including "Zi2Zi" [8], "Rewrite" [9], PEGAN [10], HAN [11], AEGG [12], DC-Font [13]. These methods employed image pyramids, hierarchical losses, and refined networks to enhance glyph restoration.

Additionally, Semantic Segmentation [14] employed CNN and Markovian Random Field for vectorized characters. Berio et al. [15] split strokes by overlapping and intersecting regions. VecFontSDF [16] created premium vector fonts using signed distance functions (SDF). DeepVecFont [17] and DeepVecFont-V2 [18] generated highly editable vector fonts with multimodality. FontTransformer [19] produced high-resolution Chinese glyph images through few-shot learning.

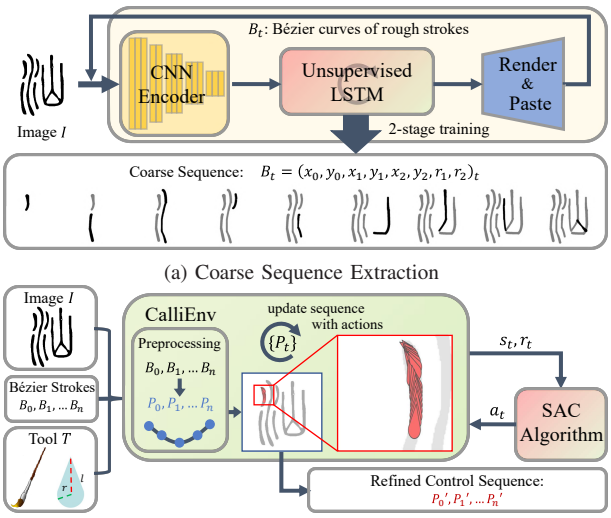
Some researchers [20–22] adopted trajectories or demonstrations, while others [2, 3] employed manually-split strokes. However, annotating data is laborious, and these approaches have limitations in handling out-of-distribution scenarios like ancient characters. Thus, unsupervised methods are crucial.

Painting and calligraphy share similar origins, but painting allows more freedom in stroke order. Various unsupervised methods used RNN, transformer, and RL to solve the image repainting task, like Sketch-RNN [23], General Virtual Sketching Framework [24], and Learning to Paint [25]. Schaldenbrandt et al. [26] extended this to robot artists. Xie et al. [27] proposed deformable generative networks for unsupervised font generation.

Despite these successes, generating well-segmented and realistic writing norms with minimal supervision remains challenging, waiting for improved designs to capture the complexity of human writing.

B. Dexterous Manipulation

Like dexterous manipulation and automated navigation, calligraphy robots also demand meticulous control. The learned strategies should adjust to different writing implements, such as reproducing a brush-written character using a chisel on a stone tablet. However, in calligraphy robots, the significance of tools is seldom discussed [28]. Traditional



(a) Coarse Sequence Extraction

(b) Tool-aware Fine-tuning

Fig. 2: Method overview: (a) Generating coarse writing trajectories using unsupervised LSTM with tailored loss for human-like glyph decomposition. (b) Reinforcement learning with utensil properties for fine-tuning dexterous control.

trajectory optimization [29, 30] and search algorithms [31, 32] fell short, while reinforcement learning (RL) offered a paradigm for adaptive tool-based learning [33]. To reduce excessive degrees of freedom (DoF) in exploration [34], some have attempted a two-stage learning approach, incorporating traditional methods like Bayesian Optimization, Greedy algorithms, or Markovian processes with RL algorithms or curriculum learning [35–37]. Others have opted for learn-from-demonstration and inverse RL due to a weariness of reward engineering [38–40].

To autonomously discover tool-specific control techniques, we adopt a coarse-to-fine strategy. Our method applies unsupervised glyph comprehension to initiate stroke guidance and RL trajectory refinement on diverse writing instruments.

We refer to writing tools as agents. While physical models can faithfully simulate deformations and dynamics [41–43], they are computationally complex. Therefore, we simplify it by modeling only the geometries and transformations of the contact surface of writing utensils on the canvas [44, 45].

III. METHOD

Figure 2 outlines our two-stage approach: Coarse Sequence Extraction and Tool-aware Fine-tuning. Given the calligraphy glyph I, a CNN-LSTM model segments initial strokes into quadratic Bézier curves $\{B_t\}$. We employ a fusion of self-supervised objectives and a progressive training strategy to enhance writing order and integrity with limited training data.

In the second phase, we formulate the task into a constrained reinforcement learning problem, implement various utensils, and tailor the environment with well-crafted rewards. Utilizing the SAC model as a controller, we leverage coarse sequences to curtail ineffective exploration.

The notations adopted in this paper are listed in Table I.

TABLE I: Notations

Notation	Description	Notation	Description	Notation	Description	Notation	Description
I'	redrawn glyph	t	current time step	o_{tool}	occupied utensil pixels	$\vec{a}_t = (\delta_x, \delta_y, (\theta))_t$	action space
γ	discount factor	I	input glyph image	o_{in}	pixels inside glyph	$\mathcal{O}_i = (x_i, y_i)$	control points
$\mathcal{C}$	cosine similarity	ρ_t	trajectory curvature	o_{out}	pixels outside glyph	$(r, l, \theta)_t$	utensil geometry
T	total time step	ϕ	perceptual model	IoU	intersection over union	$\vec{v}_t = (v_x, v_y)_t$	stroke direction
p_t	pen state	S_t	9-dim state space	α	entropy temperature	B_t	quadratic Bézier
w_i	stroke width	h_t	writing progress	τ	soft update coefficient	$\{\mathbf{P}_i\}$	discrete points on B_t

A. Extracting Coarse Sequences

1) Model Design: Mo et al. [24] introduced a framework for generating vector line art but neglected to preserve writing order due to simple loss terms. Our approach retains the foundational architecture while introducing refined objectives to achieve more continuous and feasible decomposition results comparable to human behaviors.

Specifically, we adopt a CNN-LSTM backbone and borrow the dynamic window generation and the differentiable cropping-pasting component. In time step t, current canvas with glyph I are convolved to infer a quadratic Bézier stroke $B_t = (p, x_0, y_0, x_1, y_1, x_2, y_2, w_0, w_1)_t$, where p indicates the pen state for drawing ($p = 1$) or lifting ($p = 0$) the utensil; $\mathcal{O}_i, i \in \{1, 2, 3\}$ denotes the control points and w_0, w_1 represent the initial and final widths, respectively. Drawing from Gestalt theory [46] and physical constraints, we formulate a set of unsupervised losses trained alongside a progressive approach.

2) Unsupervised Losses:

Perceptual loss. We utilize the VGG-16-based perceptual loss [47] to ensure redrawn similarity. It compares the similarity between the rendered output $\hat{y}$ and the target glyph y. The jth perceptual loss is

$$\mathcal{L}^j_{\text{perc}} = \frac{1}{D_j \times H_j \times W_j} \left\| \phi_j(\hat{y}) - \phi_j(y) \right\|_1 . \tag{1}$$

To prevent excessive loss fluctuations, we normalize each loss value for each layer with t and compute the total loss at layers `relu1_2`, `relu2_2`, `relu3_3` and `relu5_1` ($J = \{12, 22, 33, 51\}$) of the VGG-16 model:

$$\mathcal{L}_{\text{perc}} = \sum_{j \in J} \mathcal{L}^j_{\text{perc-norm}} = \sum_{j \in J} \frac{\mathcal{L}^j_{\text{perc}}}{\mathcal{L}^j_{\text{perc-mean}}}. \tag{2}$$

Regularization loss. We introduce the stroke regularization term to reduce redundant strokes by penalizing lifting actions ($p_t = 0$):

$$\mathcal{L}_{\text{reg}} = \frac{1}{T} \sum_{t=1}^{T} (1 - p_t). \tag{3}$$

Smoothness loss. Decomposing overlapping strokes and intersections is pivotal in glyph recognition. Rooted in the Gestalt theory, the continuity principle underscores our inclination to perceive connected and unbroken elements. To achieve this, we employ the cosine similarity $\mathcal{C}$ between $\overrightarrow{\mathcal{O}_1\mathcal{O}_0}$ and $\overrightarrow{\mathcal{O}_1\mathcal{O}_2}$ to yield generating smoother strokes:

$$\mathcal{L}_{\text{smo}} = \sum_{t=1}^{T} p_t \cdot \frac{1 + \mathcal{C}(\overrightarrow{\mathcal{O}_1\mathcal{O}_0}, \overrightarrow{\mathcal{O}_1\mathcal{O}_2})}{2}. \tag{4}$$

To mitigate false penalization of real-exist curves, we incorporate a scaling factor of 0.5. Consequently, while decomposing a curvature into multiple strokes does reduce $\mathcal{L}_{\text{smo}}$, the consequential increase in $\mathcal{L}_{\text{reg}}$ is more pronounced. Overall, this results in a higher cumulative loss.

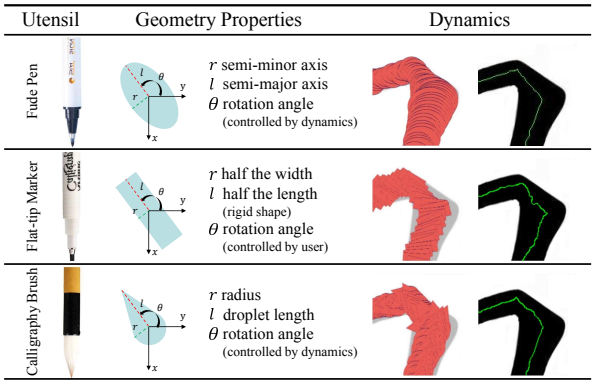


Fig. 3: Brief introduction of our modeled writing utensils. More information can be found in the supplemental material.

Angle loss. Physical constraints in the human arm and muscles make us write with a starting direction from a certain angle range, approximately around +75° and -165°. Therefore, for adjacent generated sequences B_{t-1} and B_t, if $p_{t-1} = 0$ and $p_t = 1$, we employ the cosine similarity $\mathcal{C}$ between $\vec{S}_t = (\overrightarrow{\mathcal{O}_0\mathcal{O}_2})_t$ and $\vec{a} = (-\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2})$ to constrain the proper writing direction:

$$\mathcal{L}_{\text{ang}} = \frac{1}{T} \sum_{t=1}^{T} \begin{cases} \mathcal{C}(\vec{S}_t, \vec{a}), & \mathcal{C}(\vec{S}_t, \vec{a}) \geq 0.5 \\ 0, & \mathcal{C}(\vec{S}_t, \vec{a}) < 0.5. \end{cases} \tag{5}$$

3) Progressive Training: We adopt a progressive training approach. Initially, the model is trained on the QuickDraw dataset using $\mathcal{L}_{\text{phase1}}$. Subsequently, fine-tuning is performed on 1000 glyphs extracted from the KaiTi-GB2312 [48] and KanjiVG [49] datasets, guided by $\mathcal{L}_{\text{phase2}}$. In the first phase, the model gains fundamental decomposition capabilities. This allows the model in the second phase to improve its decomposition results while efficiently handling variations in calligraphy stroke thickness. Subsequent experiments have confirmed the necessity of this approach. During training, λ_1 and λ_2 increase from 0 to 0.5 as training goes on.

$$\mathcal{L}_{\text{phase1}} = \mathcal{L}_{\text{perc}} + \lambda_1 \cdot \mathcal{L}_{\text{reg}} , \tag{6}$$

$$\mathcal{L}_{\text{phase2}} = \mathcal{L}_{\text{perc}} + \mathcal{L}_{\text{reg}} + \lambda_2 (0.5 \cdot \mathcal{L}_{\text{smo}} + \mathcal{L}_{\text{ang}}). \tag{7}$$

B. Tool-Aware Fine-tuning

Previous methods applied segmented strokes to robotic arms without accounting for the impact of writing instruments on control trajectory. We use RL methods to master the plug-and-play utensil models and fine-tune coarse trajectories to improve alignment with real-world robotic scenarios.

TABLE II: The SAC Model and Training Configurations

Actor-Network	($\|s\|$, 64, 128, $\|a\|$)
Critic-Network	($\|s\| + \|a\|$, 128, 128, 128, 1)

Parameter	Value	Parameter	Value
α	0.95	Optimizers	Adam
γ	0.95	Replay Buffer	2^{20}
τ	0.95	Batch Size	2^{10}
policy_lr	1e-4	Training Epochs	150
q_lr	1e-4	Steps per Epoch	10000

1) Constrained Sequence Optimization: Calligraphy writing requires thin and precise strokes and challenges sampling effectiveness. Naive exploration leads to out-of-bound collections. Therefore, we focus on the exploration process around the coarse trajectory, avoiding the disruption of ineffective experiences while encouraging the agent to practice diverse utensil control within the boundary. We discretize B_t into $\{\mathbf{P}_i\}_t$: $\mathbf{P}_i = (p, x, y)_i$ and restrain the exploration zone as $\mathbf{P}_t + \vec{a}_t$, where the RL model predicts the action $\vec{a}_t$.

2) Model Design: We use Soft Actor-Critic (SAC) due to its stability and quick convergence. It adheres to the actor-critic paradigm, featuring two critics to decrease overestimation bias. The objective is to maximize entropy and total reward to encourage exploration and prevent overfitting. The importance of entropy is weighted by α to promote randomness. The configurations about the networks and other hyperparameters are available in Table II.

3) CalliEnv Environment: We customize the RL environment with several crucial designs.

Plug-and-play utensil models. We instantiate writing tools by considering their geometric attributes, encompassing two length measurements (r and l), an angular parameter (θ), and their dynamic functions. We select three tools, displayed in Figure 3: the fude pen, flat-tip marker, and calligraphy brush, covering rigid and flexible forms varying from circular to quadrangular shapes. The parameters and movement dynamics are all approximation results from the real world, which will be discussed in the supplementary.

State and action spaces. The vector state at timestep t, denoted as S_t, comprises a 9-dimensional tensor: $S_t = (h, r, l, \theta, \rho, \delta_x, \delta_y, v_x, v_y)_t$, where h denotes the current stroke progress; r, l, and θ represents utensil geometry; ρ signals the current curvature; δ_x, δ_y represent prior actions as offsets; and v_x, v_y signify the moving direction within the ongoing stroke.

For most circumstances, the action space defines the displacement coordinates around the point $\mathbf{P}_t$, represented by $\vec{a}_t = (\delta_x, \delta_y)_t$. In the case of tools like a flat-tip marker, the user's wrist orientation, rather than dynamics, controls the angular attribute θ. Thus, we add a new DoF θ into $\vec{a}_t$.

The RL algorithm controls the agent to move through each $\mathbf{P}_t$ and apply actions, adjusting the scattered points to $\mathbf{P}_t + \vec{a}_t$. The model manipulates the writing utensil iteratively to generate precise control points $\{\mathbf{P}'_t\}$ for writing.

4) Reward Design: We tailor the rewards for the RL algorithm to learn skillful control and boost convergence.

Adaptive shape reward. In calligraphy, versatile control is applied at stroke beginnings and ends with fewer variations in the middle. To encourage exploring diverse approaches, we formulate a composite reward, fostering smooth strokes in the middle while imposing penalties for actions that exceed glyph boundaries at the endpoints, seeking control to align with the writing tool:

$$\mathcal{R}_{\text{ada}} = \begin{cases} -\sqrt{\frac{r_{\max}}{r_t}} \cdot \left\| \frac{1}{r_t} - \frac{1}{r_{t-1}} \right\|, & h_t \in [0.2, 0.8] \\ \frac{o_{\text{in}} - o_{\text{out}}}{o_{\text{tool}}}, & h_t \in [0, 0.2) \cup (0.8, 1], \end{cases} \tag{8}$$

where o_{tool} represents pixels occupied by the writing tool, o_{in} represents the overlapping pixels between the tool and the glyph, and o_{out} covers the tool area outside stroke boundaries.

Terminal reward. To ensure redrawn similarity, we apply the Intersection over Union (IoU) [50] to compare the redrawn and initial glyph. We multiply the termination reward by a factor of 80 to improve the sensitivity of the SAC algorithm to reward shaping:

$$\mathcal{R}_{\text{fin}} = 80 \cdot \text{IoU}(I', I). \tag{9}$$

In summary, the total reward for a period is:

$$\mathcal{R} = \frac{1}{T} \sum_{t=1}^{T} \mathcal{R}_{\text{ada}} + \mathcal{R}_{\text{fin}} \,. \tag{10}$$

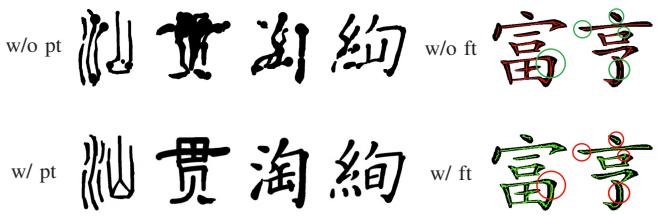


Fig. 4: (a) Progressive Training ("pt") Comparison. (b) Impact of Fine-Tuning ("ft") on Stroke Connectivity and Tool-Aware Control, including joining disconnected strokes and adding brush-related control at both ends of the strokes.

IV. EXPERIMENTS

We quantitatively and qualitatively evaluate our approach to diverse Chinese and English scripts, affirming its versatility. Trained solely on QuickDraw and Kaiti-GB2312, our method successfully rewrites characters in Tamil and Ancient Egyptian using the Dobot Magician robot arm.

A. Datasets and Implementation Details

1) Training: Training efficiently with limited data poses challenges. We construct a dataset of 1000 glyphs from KaiTi-GB2312 [48] and KanjiVG [49]. Employing progressive training, we first train the sequence extraction module on QuickDraw for 75,000 steps and then perform an additional 30,000 steps on our dataset. We use the Adam optimizer with a learning rate of 1e-4, set the batch size to 32, and train the first phase on two RTX-2080 GPUs.

Figure 4 illustrates results for models trained directly for 75,000 steps on 1000 glyphs and those subjected to progressive training. Progressive training enhances the model's ability to reconstruct glyphs under low-resource conditions.

We build the CalliEnv environment with OpenAI Gym and Tianshou for modularity and multi-process training (Table II). The environment utilizes glyph images and decom-

TABLE III: Low Resource Training

Train Size	SNR (↓)	CD (↓)
500	1.260	2.134
1000	**1.200**	**1.979**
2000	1.204	2.153
5000	1.238	2.135

TABLE IV: Ablation on Unsupervised Losses

Losses	SNR (↓)	CD (↓)
–	5.602	2.092
$\mathcal{L}_{\text{smo}}$	1.341	2.448
$\mathcal{L}_{\text{ang}}$	1.376	2.134
$\mathcal{L}_{\text{smo}} + \mathcal{L}_{\text{ang}}$	**1.199**	**1.979**

position results to generate refined control, with the trained policy for validation.

2) Validation: We evaluate stroke integrity and pixel-wise similarity using Stroke Number Ratio (SNR) and Chamfer Distance (CD). SNR expresses the ratio of partition numbers N_s to the number of ground-truth strokes N_{gt} as SNR $= \frac{N_s}{N_{gt}}$. A lower SNR value indicates a better stroke continuity. Chamfer Distance, commonly used to compute the similarity between point clouds, is utilized here to calculate the contour similarity between generated and reference glyph images. The definition proposed by Fan et al. [51] is adopted to extract and compare the contours.

We use 200 Chinese glyphs in the Kaiti style with annotated stroke numbers for quantitative testing. During robotic experiments, we utilize various languages and scripts like Chinese calligraphy, English, Standard Tamil and ancient Egyptian hieroglyphs. The control sequences are verified through physical tests and renderings on a Dobot-Magician robot arm with writing tools. The results confirm the effectiveness of our method both qualitatively and quantitatively.

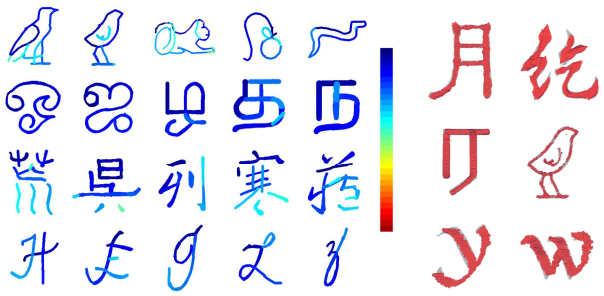
B. Extracting Coarse Sequences

1) Low Resource Training: We examine our model's performance over different training set sizes from 500 to 2000 glyphs. Table III shows that our model achieves optimal SNR and CD with just 1000 images while increasing or decreasing the dataset size has minor impacts on the overall performance. This indicates that an LSTM model is enough for the task of learning from small and unsupervised datasets.

2) Ablation on Unsupervised Losses: Unsupervised losses $\mathcal{L}_{\text{smo}}$ and $\mathcal{L}_{\text{ang}}$ significantly impact decomposition quality. We perform an ablation study on them to validate our design. Table IV confirms that combining $\mathcal{L}_{\text{smo}}$ and $\mathcal{L}_{\text{ang}}$ yields the highest effectiveness in terms of both SNR and CD.

3) Comparison to Other Methods: We compare our model to various methods, including "Learning to Paint" (LTP) [25], "General Virtual Sketching Framework" (GVS) [24], and "VectorNet" [14]. VectorNet specializes in character segmentation and uses an OverlapNet for stroke intersection prediction, while the former two methods represent stroke-based image repainting models. Both methods are evaluated on 200 Chinese glyphs in the Kaiti style with ground-truth stroke number.

In comparison, we limit the maximum strokes of LTP and GVS to 60 and 49 segments. We compute the average SNR and compare Chamfer distances between the reference and the reconstructed image, or vectorized rendered glyph with the size of 256×256 pixels. Table V shows that our model outperforms unsupervised methods in SNR and Chamfer distance and is compatible with supervised models, with Figure 6 unfolds our decomposed strokes with better integrity and order while maintaining a high reconstruction quality in rendered images, as also seen in Figure 5(a).



(a) (b)

Fig. 5: (a) Visualization on decomposed coarse sequences. From blue to red: stroke order from the 1st to the 100th. (b) Tool-aware fine-tuning with three writing utensils.

TABLE V: Quantitative Comparison between Methods

	Supervision	SNR (↓)	CD (↓)
LTP [25]	RL	5.660	6.978
GVS [24]	Unsupervised	5.602	2.092
Ours	Unsupervised	**1.199**	**1.979**
VectorNet [14]	Supervised	1.039	1.614

C. Fine-tuning Dexterous Control

1) Devising Mastery over Instruments: We test various tools in CalliEnv using the SAC algorithm, including calligraphy brushes, fude pens, and flat-tip markers. Different writing tools are used based on the font type: brushes for Chinese, fude pens for rounder English, Ancient Egyptian and Tamil, and flat-tip markers for angular English. Figure 5(b) displays the visual results of the rewriting process for each tool after 64 training epochs.

We also explore diverse tools on the same character shape. In Figure 3, we display the experiments on the Chinese character "Heng Zhe". It's evident that an Ellipse-shaped brush can produce smoother strokes, the marker learns to change direction during turns, and a brush introduces more details at the turning points, including representations of lifting and turning the brush with sharp angles.

2) Rectifying Coarse Sequences: Besides accommodating the coarse sequences to specific instruments, the fine-tuning process can also rectify errors, connecting false breakpoints and stretching the trajectory from a wrong place. As the tool explores the fine-tuned trajectories, it will guide the trajectory toward areas with higher rewards. In Figure 4(b), we observe that the brush model learns the intricate movements of pen turning at the beginning and end of writing.

D. Robotic Demonstration

We use a 4-DoF Dobot Magician robot to replicate scripts with physical tools. After optimization, fine control points (x, y, r) are transferred to 3D positions (x, y, z) via calibration for further inverse-kinematics calculation. We adopt the robotic arm with writing tools to leave traces on the paper at varying z heights, measure widths, calculate r, and establish

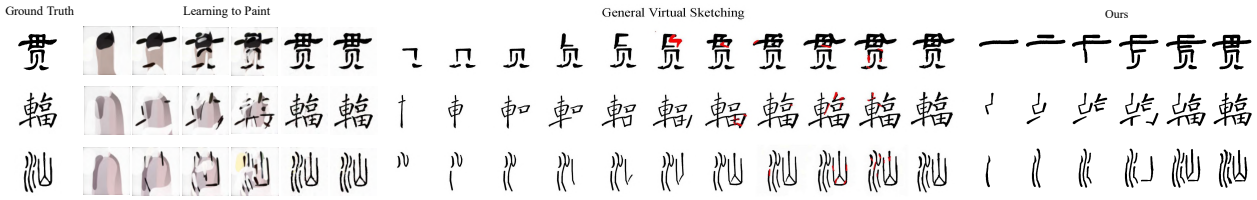


Fig. 6: Rendered comparison on diverse unsupervised models: LTP can not generate semantic-aware stroke orders. GVS partially restores ordering but suffers from overlapping and incorrect decomposition (highlighted), affecting stroke integrity. In contrast, our method delivers superior stroke splitting, order, fewer overlaps, and quality similar to GVS in rendering.

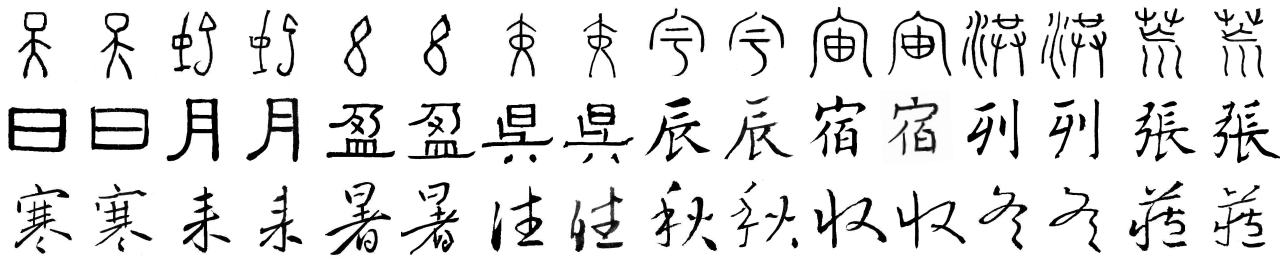


Fig. 7: Rewriting various Chinese scripts: the Oracle (left, up); Seal (right, up); Clerical (left, mid); Regular (right, mid); running (left, down) and cursive (right, down). Each pair consists of an input glyph (left) and a robot-generated result (right).

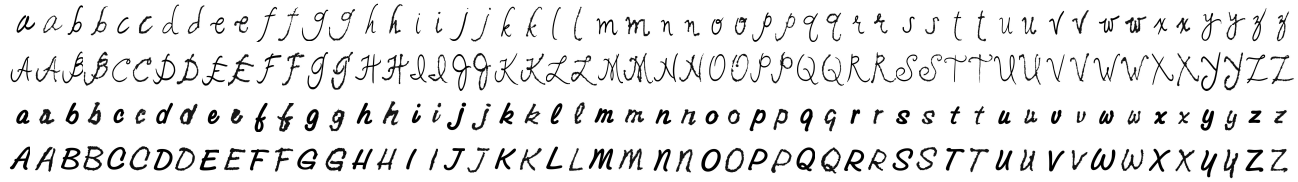


Fig. 8: Rewriting two English scripts with fude pen. Each pair includes an input glyph (left) and a robot arm replica (right).

Fig. 9: Results on Ancient Egyptian (left) and Tamil (right) characters replicated by Dobot Magician robot arm.

the z-r connection through linear fitting. We conduct experiments in real-world scenarios, rewriting glyphs in various languages and fonts using calligraphy brushes and fude pens.

1) Rewriting Chinese Glyphs: Chinese Calligraphy poses challenges to learners and robots due to its intricate techniques, such as lifting, turning, pressing, and mastering the inelastic soft brush. We select the first 24 characters from the *Thousand Character Classic* with Oracle to Cursive scripts, apply a real calligraphy brush on the robot arm, and set the redrawn size to 8×8 cm^2. Figure 7 shows our ability to rewrite styles close to the ground truth, with slight distortion in a few characters due to zero adaptation for sim2real.

2) Rewriting English Characters: We test the fude pen and the flat-tip marker on two English cursive script datasets. The rewriting size of each character is 2×2 cm^2. The robot-written results are shown in Figure 8. The letters redrawn by the robotic arm coincide well with the original glyphs.

3) Test on Other Languages: We also test our model on unseen characters or hieroglyphs, such as Ancient Egyptian and Tamil. Figure 5 and Figure 9 visualize the segmented and rewritten results. In cases when characters are unfamiliar or even like sketches, CalliRewrite still possesses compatible segmentation abilities and can rewrite faithfully.

Fig. 10: Failure cases. The right character is harmed by false stroke segmentation, while the left is also caused by error accumulation.

V. CONCLUSIONS

This paper introduced an unsupervised approach for faithfully reproducing diverse characters using different tools, achieving impressive results without any annotation requirements. However, As shown in Figure 10, there also exist some failure cases in our method. This is mainly due to inaccurate stroke segmentation and error accumulation in the robotic arm. These challenges may arise from the model's inherent difficulty in perfectly segmenting typical fonts in unsupervised settings, imprecision in tool modeling, and a lack of real-world training scenarios to bridge the sim2real gap. We leave these challenges as our future work.

VI. ACKNOWLEDGMENT

This work was supported by National Natural Science Foundation of China (Grant No.: 62372015), Center For Chinese Font Design and Research, and Key Laboratory of Intelligent Press Media Technology.

REFERENCES

[1] R. Wu, F. Chao, C. Zhou, Y. Huang, L. Yang, C.-M. Lin, X. Chang, Q. Shen, and C. Shang, "A developmental evolutionary learning framework for robotic chinese stroke writing," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 14, no. 3, pp. 1155–1169, 2021.

[2] L. Gan, W. Fang, F. Chao, C. Zhou, L. Yang, C.-M. Lin, and C. Shang, "Towards a robotic chinese calligraphy writing framework," in *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2018, pp. 493–498.

[3] Y. Yang, W. Chen, L. Zhou, B. Zheng, W. Xiao, Y. Huang, and Z. Sun, "A hybrid control framework teaching robot to write chinese characters: from image to handwriting," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2021, pp. 1161–1166.

[4] A. Bidgoli, M. L. De Guevara, C. Hsiung, J. Oh, and E. Kang, "Artistic style in robotic painting; a machine learning approach to learning brushstroke from human artists," in *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2020, pp. 412–418.

[5] R. Wu, C. Zhou, F. Chao, L. Yang, C.-M. Lin, and C. Shang, "Integration of an actor-critic model and generative adversarial networks for a chinese calligraphy robot," *Neurocomputing*, vol. 388, pp. 12–23, 2020.

[6] F. Chao, Y. Huang, C.-M. Lin, L. Yang, H. Hu, and C. Zhou, "Use of automatic chinese character decomposition and human gestures for chinese calligraphy robots," *IEEE Transactions on Human-Machine Systems*, vol. 49, no. 1, pp. 47–58, 2018.

[7] S. Xu, F. C. Lau, W. K. Cheung, and Y. Pan, "Automatic generation of artistic chinese calligraphy," *IEEE Intelligent Systems*, vol. 20, no. 3, pp. 32–39, 2005.

[8] "zi2zi," https://github.com/kaonashi-tyc/zi2zi, 2017.

[9] "Rewrite," https://github.com/kaonashi-tyc/rewrite, 2016.

[10] D. Sun, Q. Zhang, and J. Yang, "Pyramid embedded generative adversarial network for automated font generation," in *2018 24th International Conference on Pattern Recognition (ICPR)*. IEEE, 2018, pp. 976–981.

[11] J. Chang, Y. Gu, Y. Zhang, Y.-F. Wang, and C. Innovation, "Chinese handwriting imitation with hierarchical generative adversarial network." in *BMVC*, 2018, p. 290.

[12] P. Lyu, X. Bai, C. Yao, Z. Zhu, T. Huang, and W. Liu, "Auto-encoder guided gan for chinese calligraphy synthesis," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 1095–1100.

[13] Y. Jiang, Z. Lian, Y. Tang, and J. Xiao, "Dcfont: an end-to-end deep chinese font generation system," in *SIGGRAPH Asia 2017 Technical Briefs*, 2017, pp. 1–4.

[14] B. Kim, O. Wang, A. C. Öztireli, and M. Gross, "Semantic segmentation for line drawing vectorization using neural networks," in *Computer Graphics Forum*, vol. 37, no. 2. Wiley Online Library, 2018, pp. 329–338.

[15] D. Berio, F. F. Leymarie, P. Asente, and J. Echevarria, "Strokestyles: Stroke-based segmentation and stylization of fonts," *ACM Transactions on Graphics (TOG)*, vol. 41, no. 3, pp. 1–21, 2022.

[16] Z. Xia, B. Xiong, and Z. Lian, "Vecfontsdf: Learning to reconstruct and synthesize high-quality vector fonts via signed distance functions," *arXiv preprint arXiv:2303.12675*, 2023.

[17] Y. Wang and Z. Lian, "Deepvecfont: Synthesizing high-quality vector fonts via dual-modality learning," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 6, pp. 1–15, 2021.

[18] Y. Wang, Y. Wang, L. Yu, Y. Zhu, and Z. Lian, "Deepvecfont-v2: Exploiting transformers to synthesize vector fonts with higher quality," *arXiv preprint arXiv:2303.14585*, 2023.

[19] Y. Liu and Z. Lian, "Fonttransformer: Few-shot high-resolution chinese glyph image synthesis via stacked transformers," *Pattern Recognition*, vol. 141, p. 109593, 2023.

[20] S. Tang, Z. Xia, Z. Lian, Y. Tang, and J. Xiao, "Fontrnn: Generating large-scale chinese fonts via recurrent neural network," in *Computer Graphics Forum*, vol. 38, no. 7. Wiley Online Library, 2019, pp. 567–577.

[21] A. Kotani and S. Tellex, "Teaching robots to draw," in *2019 International conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 4797–4803.

[22] J. Singh, C. Smith, J. Echevarria, and L. Zheng, "Intelli-paint: Towards developing more human-intelligible painting agents," in *European Conference on Computer Vision*. Springer, 2022, pp. 685–701.

[23] D. Ha and D. Eck, "A neural representation of sketch drawings," *arXiv preprint arXiv:1704.03477*, 2017.

[24] 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.

[25] Z. Huang, W. Heng, and S. Zhou, "Learning to paint with model-based deep reinforcement learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 8709–8718.

[26] P. Schaldenbrand and J. Oh, "Content masked loss: Human-like brush stroke planning in a reinforcement learning painting agent," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 1, 2021, pp. 505–512.

[27] Y. Xie, X. Chen, L. Sun, and Y. Lu, "Dg-font: Deformable generative networks for unsupervised font generation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 5130–5140.

[28] X. Wang, Y. Yang, W. Wang, Y. Zhou, Y. Yin, and Z. Gong, "Generative adversarial networks based motion learning towards robotic calligraphy synthesis," *CAAI Transactions on Intelligence Technology*, 2023.

[29] F. H. Clarke and I. Ekeland, "Nonlinear oscillations and boundary value problems for hamiltonian systems," *Archive for Rational Mechanics and Analysis*, vol. 78, no. 4, pp. 315–333, 1982.

[30] M. A. Johnson and M. H. Moradi, *PID control*. Springer, 2005.

[31] S. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *Research Report 9811*, 1998.

[32] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[33] R. Wu, W. Fang, F. Chao, X. Gao, C. Zhou, L. Yang, C.-M. Lin, and C. Shang, "Towards deep reinforcement learning based chinese calligraphy robot," in *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2018, pp. 507–512.

[34] G. Matheron, N. Perrin, and O. Sigaud, "Pbcs: Efficient exploration and exploitation using a synergy between reinforcement learning and motion planning," in *International Conference on Artificial Neural Networks*. Springer, 2020, pp. 295–307.

[35] Z. Yang, K. Yin, and L. Liu, "Learning to use chopsticks in diverse gripping styles," *ACM Transactions on Graphics (TOG)*, vol. 41, no. 4, pp. 1–17, 2022.

[36] L. Zhang, Z. Hou, J. Wang, Z. Liu, and W. Li, "Robot navigation with reinforcement learned path generation and fine-tuned motion control," *IEEE Robotics and Automation Letters*, 2023.

[37] I. Akinola, Z. Wang, and P. Allen, "Clamgen: Closed-loop arm motion generation via multi-view vision-based rl," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 2376–2382.

[38] Y. Sun, H. Qian, and Y. Xu, "Robot learns chinese calligraphy from demonstrations," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 4408–4413.

[39] F. Chao, Y. Huang, X. Zhang, C. Shang, L. Yang, C. Zhou, H. Hu, and C.-M. Lin, "A robot calligraphy system: From simple to complex writing by human gestures," *Engineering Applications of Artificial Intelligence*, vol. 59, pp. 1–14, 2017.

[40] F. Xie, P. L. Meur, and C. Fernando, "End-to-end manipulator calligraphy planning via variational imitation learning," *arXiv preprint arXiv:2304.02801*, 2023.

[41] 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.

[42] 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.

[43] 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.

[44] S. Wang, J. Chen, X. Deng, S. Hutchinson, and F. Dellaert, "Robot calligraphy using pseudospectral optimal control in conjunction with a novel dynamic brush model," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 6696–6703.

[45] K. W. Lo, K. W. Kwok, S. M. Wong, and Y. Yam, "Brush footprint acquisition and preliminary analysis for chinese calligraphy using a robot drawing platform," in *2006 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2006, pp. 5183–5188.
[46] W. Köhler, "Gestalt psychology," *Psychologische Forschung*, vol. 31, no. 1, pp. XVIII–XXX, 1967.
[47] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*. Springer, 2016, pp. 694–711.
[48] "Kaiti-GB2312," https://www.onlinewebfonts.com/download/8a1e9fe86f7a9489ec091ec4b78af185, accessed: August 25, 2023.
[49] "KanjiVG," http://kanjivg.tagaini.net/, accessed: August 25, 2023.
[50] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 658–666.
[51] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3d object reconstruction from a single image," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 605–613.