ECE 445

SENIOR DESIGN LABORATORY

PROJECT PROPOSAL

Handwriting Robot With User-Customized Font Style

Team #35

ZHIXIANG LIANG
(zliang18@illinois.edu)
ZIFAN YING
(zifany4@illinois.edu)
XUANCHENG LIU
(zl124@illinois.edu)
MINGCHEN SUN
(msun52@illinois.edu)

TA: Tielong Cai, Tianci Tang

March 14, 2025

Contents

1 Introduction

1.1 Problem

Handwriting remains a personal and unique form of expression, yet current digital and automated writing solutions lack the ability to accurately replicate individual handwriting styles. Existing methods either rely on digital fonts that imitate handwriting or require complex, manual customizations [1]. This project addresses the need for an automated system that can learn and reproduce a person's unique handwriting style with high fidelity [2]. By integrating machine learning-based handwriting analysis with robotic writing mechanisms, this system enhances document personalization, enabling applications in personalized correspondence, secure document signing, and artistic reproduction.

1.2 Solution

Our solution consists of a complete pipeline integrating three core subsystems: a handwriting sample analysis subsystem, a few-shot font generation subsystem, and a robotic handwriting mechanism. Initially, the user's handwritten samples are processed and segmented into standardized grayscale images, serving as input to a few-shot font generation system. Using advanced machine learning techniques, this subsystem extracts and learns the user's handwriting style. Finally, generated stroke data is sent to a robotic handwriting subsystem, which physically replicates the handwriting using precise motor control.

The robotic handwriting subsystem comprises a two-axis rail system driven by high-resolution stepper motors and an additional axis for pen actuation. Motor control signals are generated by an ESP32 module, which also manages data communication with the host computer via UART, Bluetooth, or WLAN. Stroke data transmitted to the ESP32 follow the standard vector graphic SVG path format, specifying precise movement commands (coordinates and pen lift/drop states).

1.3 Visual-Aid

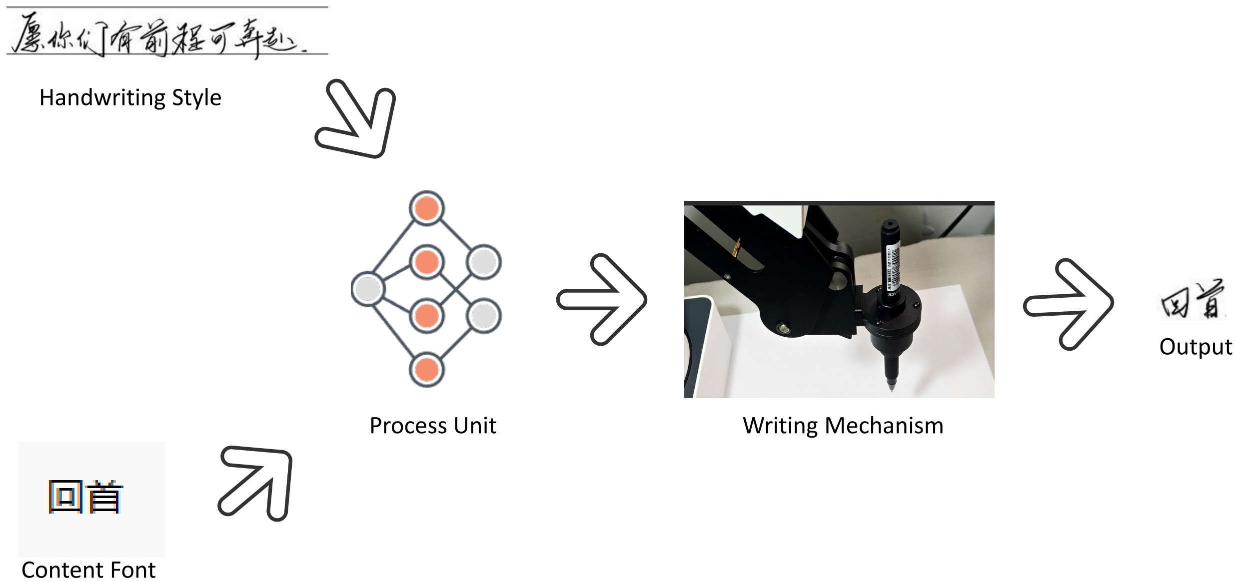

Figure 1: The Visual Illustration

1.4 High-Level Requirements

Few-shot Font Generation System

- The font generation inference should process each character in under 500 milliseconds, ensuring efficiency for practical usage scenarios.
- The system should reliably handle variations in handwriting input quality, maintaining style fidelity with less than 5% performance degradation in the presence of moderate image noise or distortion.

Robotic Handwriting Subsystem

- The subsystem must reproduce handwriting with positional accuracy within ±0.1 mm to achieve clear and authentic handwriting replication.
- Control signal synchronization must maintain a maximum timing deviation of less than 1 ms between ESP32 signals and motor response to ensure smooth and continuous pen movements.
- Communication between the ESP32 module and the host computer must achieve a data transmission reliability greater than 99.9% with latency below 100 ms, ensuring real-time handwriting reproduction.

2 Design

2.1 Block Diagram

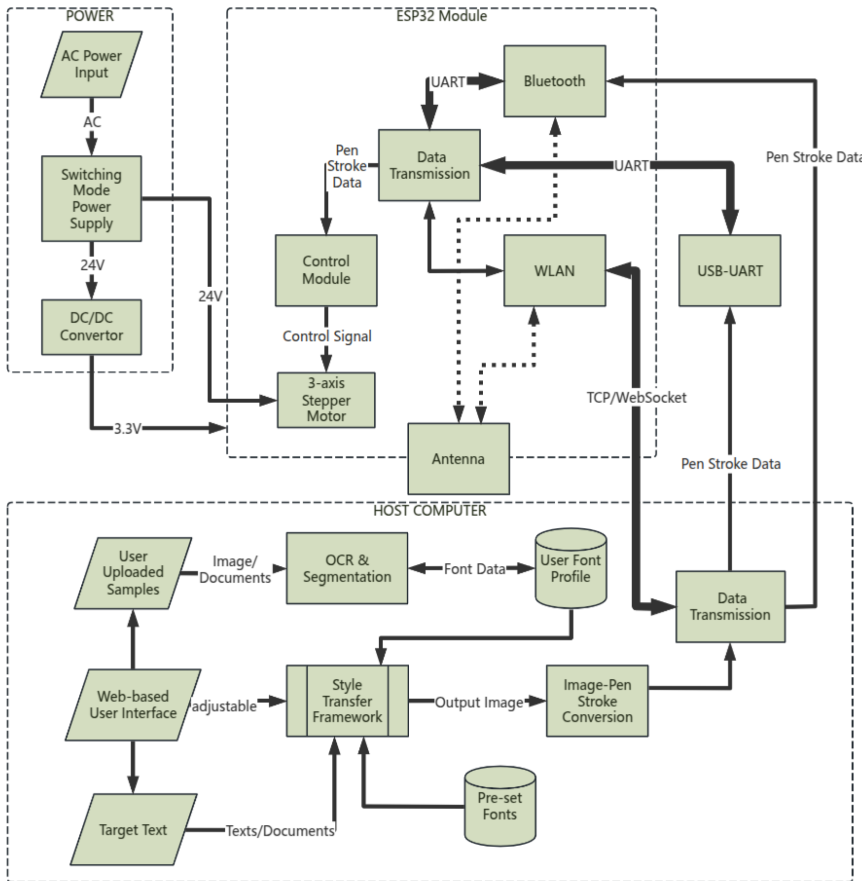

Figure 2: Block Diagram of Our System

2.2 Subsystem Overview

The Handwriting Robot System is composed of 2 main subsystems:

2.2.1 Few-shot Font Generation System

Our system utilizes a few-shot font generation approach to produce fonts that match the style specified by the user through a small number of sample fonts. Specifically, the font samples provided by the user are segmented into 128×128 grayscale images, which serve as references for style learning. Then, based on the content specified by the user, the system generates fonts that imitate the user's writing style as shown in Figure 3. Then we employ the *findContours* function in OpenCV to extract the contour point vectors of the generated images, obtaining the positions of the feature points. Since Chinese characters predominantly consist of straight lines, these feature points effectively represent the entire contour for subsequent use by robotic arms.

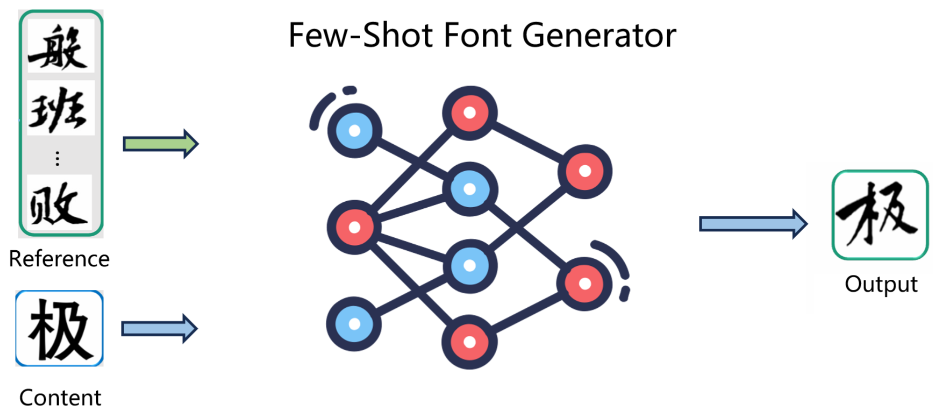

Figure 3: Few-shot Font Generator

For few-shot font generator, we employ a few-shot font generation system inspired by VQ-Font [3], which integrates two essential components:

- **Global Style Aggregator (GSA):** This component uses character similarity to guide and ensure consistency in the overall style across similar characters.
- **Local Style Aggregator (LSA):** Utilizing vector quantization and cross-attention, LSA captures intricate details, allowing for the learning of elements without the need for manual definition.

The content decoder (generator) is trained using Generative Adversarial Networks (GANs) combined with a self-reconstruction framework, enabling its application across various scripts without requiring extensive labeled data.

2.2.2 Robotic Handwriting Subsystem

The robotic subsystem translates digital stroke path data into physical pen movements, replicating handwriting on paper. This subsystem consists of:

- **Two-axis Rail System:** Utilizes precision rails driven by stepper motors along X and Y axes, enabling precise planar motion control.
- **Pen Actuation Axis:** Controls vertical pen motion for contacting or lifting from the writing surface, achieving clear strokes.
- **ESP32 Microcontroller Module:** Generates rising and falling edge signals for motor actuation, ensuring accurate step control, and manages communication with the host computer. The square wave control signal can be generated by Remote Control Transceiver (RMT) api [4] in a non-blocking manner.

The subsystem accepts stroke data in SVG format, comprising sequential pen coordinate data and lift/drop states, which ESP32 translates into precise stepper motor signals. The

SVG `path` element is capable of Bezier curve of second and third order, or discrete point paths [5].

2.3 Subsystem Requirements

2.3.1 Few-shot Font Generation System

The implementation of the few-shot font generation system is expected to align with the following requirements:

- **Training Few-shot Font Generator:** The training happens in two stages. The first stage is the pre-training of VQ-VAE [6]. This step uses 3,000 Chinese characters from a few fonts, each resized to 128x128 pixels, to learn local style components. It employs a Vector Quantized Variational Autoencoder (VQ-VAE) with a reconstruction loss and a latent loss, using an embedding dimension of 256, a batch size of 256, and 50,000 iteration steps.The second stage is the training of GAN [7]. After pre-training, the full font generation model is trained using a Generative Adversarial Network (GAN). It includes a fixed pre-trained content encoder, a style encoder trained from scratch, and other components like style aggregators and a decoder. The training uses a batch size of 48, 8 attention heads in three stacked transformer layers, and runs for 500,000 iteration steps.
- **Extracting Handwritten Characters:** Begin by having the user write Chinese characters on grid paper and then capturing an image of this paper, which is subsequently converted to grayscale. Following this, identify the grid edges and segment each character area accordingly, removing any unnecessary blank spaces to isolate the character itself. Finally, resize the extracted character images to 128×128 pixels, ensuring their original aspect ratio is preserved, to create uniform grayscale images.
- **Inference with Desired Content:** Given a target character and 3-5 reference glyphs, the font generator first encodes the content separately, extracts style features from references, and combines them for generation. The Global Style Aggregator (GSA) uses similarity to weight overall style while the Local Style Aggregator (LSA) applies vector quantization and cross-attention to transfer details. This inference method is efficient, processing components in one pass, ideal for font library creation. The similarity-guided GSA enhances flexibility, adapting style to content structure. For optimal inference, we will use 3-5 clear references and avoid rare or intricate inputs.

2.3.2 Robotic Handwriting Subsystem

The robotic subsystem must adhere to these specifications:

- **Motor Resolution:** Stepper motors must provide at least 200 steps per revolution (1.8 degrees per step) or higher to achieve fine control for detailed handwriting.
- **Control Signal Accuracy:** ESP32-generated control signals must accurately synchronize motor movements with timing errors below 1 ms.

- **Communication Reliability:** Data transmission between ESP32 and the host computer should maintain a packet loss rate below 0.1% via UART, Bluetooth, or WLAN.
- **SVG Path Format Compliance:** The subsystem must fully support SVG path command formats including absolute (M, L) and relative (m, l) movements, as well as pen lift/drop (Z/z commands).
- **Mechanical Stability:** Rail systems using belts or screws must maintain positional accuracy within ±0.1 mm to replicate handwriting accurately.

2.4 Tolerance Analysis

2.4.1 Few-shot Font Generation System

- **Reference Sample Variations:**

  The performance metrics of the model improve with an increase in reference samples from 1 to 8, plateauing between 5 to 8 samples. The model exhibits high tolerance with 3 to 8 references but performs unsatisfactorily with fewer than 3 due to insufficient style cues. Hence, at least 3 references are needed for reliable style transfer, which limits the technique's application in scenarios with very limited data.

- **Quality and Noise in Input Data:**

  The model is designed to work with high-quality reference images and has not been tested on noisy inputs. The style encoder's ability to handle noise may be limited due to its reliance on precise feature extraction from clean images. It is crucial for us to accurately extract and segment user-provided inputs to yield outputs that contain the user's stylistic font for the model's use.

- **Training Parameters and Resource Intensity:**

  The model's training process involves substantial resources, including large batch sizes and many iterations. The model likely has low tolerance for reduced training settings, as these could disrupt convergence and affect the stability of the GAN training. This indicates a high demand for computational resources during training.

2.4.2 Robotic Handwriting Subsystem

- **Mechanical Tolerance:** Precision in handwriting replication depends critically on mechanical tolerances. Rails and belts/screws must ensure minimal backlash and slippage. Any mechanical tolerance exceeding ±0.1 mm may visibly distort handwriting. Careful design and routine calibration are required.
- **Timing and Signal Accuracy:** Stepper motors rely on precise timing of control signals. Variations greater than 1 ms in signal generation or processing can degrade handwriting quality. This subsystem requires stringent synchronization between the ESP32 signals and the motor response, necessitating real-time firmware optimization.

- **Communication Stability:** The subsystem's performance is sensitive to communication reliability. If communication latency or packet loss exceeds thresholds (0.1%), pen movements may become irregular, resulting in discontinuities or unintended strokes. Robust error-checking protocols and efficient data encoding should mitigate these risks.

3 Ethics and Safety

3.1 Ethical Considerations

Our handwriting robot raises several ethical considerations which require careful handling. First, there is the potential for misuse in document forgery or identity impersonation. As this technology can precisely replicate individuals' handwriting styles, it's essential to implement security measures that restrict usage to authorized individuals. Additionally, it is important to include traceability or watermarking capabilities within generated outputs to prevent or detect unauthorized use or forgery attempts.

Second, data privacy is paramount, especially given that users' handwriting samples represent sensitive personal biometric information. Users' handwritten samples must be securely stored, and explicit consent should be obtained for data usage and storage, strictly adhering to privacy standards such as GDPR or similar local regulations.

3.2 Safety Considerations

The handwriting robot involves moving mechanical parts driven by stepper motors, which pose potential safety risks such as physical injury due to unintended movements or malfunction. To mitigate this, the following measures must be enforced:

- **Emergency Stops:** The design must include accessible and clearly marked emergency stop mechanisms to immediately disable all motorized movements in the event of a malfunction.
- **Operational Limits:** Implementing software limits and physical end-stops to prevent motors from exceeding operational ranges, reducing the risk of mechanical damage or personal injury.
- **Electrical Safety:** Adequate insulation and protective casing for electronic components, particularly the ESP32 microcontroller and stepper motor drivers, are necessary to prevent electrical hazards.
- **Component Reliability:** Regular maintenance schedules and quality assurance checks on belts, screws, and rail systems should be performed to detect early signs of wear or damage, ensuring long-term safe operation.

By proactively addressing these ethical and safety concerns, we ensure responsible development and use of our handwriting robotic system, aligning with IEEE standards and the broader societal good.

References

[1] K. Deekshitha, C. Patange, M. Harshitha, and P. Y.J, “Automated handwriting machine,” in *2023 IEEE 8th International Conference for Convergence in Technology (I2CT)*, 2023, pp. 1–4. DOI: 10.1109/I2CT57861.2023.10126330.

[2] R. A. Xiong, “Cost-effective design of scribe ai – robotic handwriting systems using raspberry pi pico and 3d printing,” in *2024 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*, IEEE, Nov. 2024, pp. 1–6. DOI: 10.1109/lisat63094.2024.10807994. [Online]. Available: http://dx.doi.org/10.1109/LISAT63094.2024.10807994.

[3] W. Pan, A. Zhu, X. Zhou, B. K. Iwana, and S. Li, *Few shot font generation via transferring similarity guided global style and quantization local style*, 2023. arXiv: 2309.00827 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2309.00827.

[4] *Remote Control Transceiver (RMT) - ESP32 - &x2014; ESP-IDF Programming Guide v5.2.5 documentation — docs.espressif.com*, https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/peripherals/rmt.html, [Accessed 17-03-2025].

[5] *Paths &x2014; SVG 2 — w3.org*, https://www.w3.org/TR/SVG2/paths.html, [Accessed 17-03-2025].

[6] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, *Neural discrete representation learning*, 2018. arXiv: 1711.00937 [cs.LG]. [Online]. Available: https://arxiv.org/abs/1711.00937.

[7] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, *et al.*, *Generative adversarial networks*, 2014. arXiv: 1406.2661 [stat.ML]. [Online]. Available: https://arxiv.org/abs/1406.2661.