Bézier Curve-Based Stroke Extraction for Handwritten Characters

Yuxuan TENG
1424512@ed.tus.ac.jp
Takuya MATSUZAKI
matuzaki@rs.tus.ac.jp

Tokyo University of Science
Tokyo, JP

Abstract

Accurate stroke extraction from handwritten character images is essential for realistic handwriting generation and stroke-based character recognition. Traditional pixel-based segmentation methods often produce fragmented strokes, failing to preserve accurate topological structures. To solve this, we propose a novel Bézier curve-based stroke extraction method. Our approach employs deep learning-based image registration to initially align handwritten images with vector-based reference fonts, followed by precise Bézier curve fitting to the skeleton of the character image. Evaluations on a Japanese handwritten character dataset show that characters reconstructed from extracted strokes achieve high OCR recognition accuracy (95.58%), indicating accurate representation of the original handwriting structures. Additionally, geometric similarity evaluated via feature-point matching confirms precise stroke extraction, though accuracy slightly decreases for complex characters with many strokes. Moreover, our Bézier-based representation naturally supports interpolation between character images by linearly interpolating corresponding control points, enabling the synthesis of diverse handwritten samples. Such synthesized variations hold promise for augmenting OCR training datasets as well as providing a succinct representation of a sample space for stroke-based character image generation.

1 Introduction

Extracting strokes from character images plays an important role in tasks such as character generation and optical character recognition (OCR). In particular, understanding the topological structures and stroke order of handwritten characters is essential for accurate recognition and controllable character synthesis. However, accurately recovering stroke information from raster images remains a challenging problem, especially for noisy or stylistically diverse handwriting. To address these challenges, we aim to extract smooth and structurally accurate strokes directly from raster images using a parametric representation suitable for downstream generation and recognition tasks.

A large body of prior work focuses on pixel-based stroke extraction, including traditional skeletonization-based methods [1, 5, 8, 13] as well as more recent pixel-wise semantic segmentation approaches using deep neural networks [9]. Skeletonization algorithms iteratively thin character images to extract the central axes of strokes, but they are highly sensitive to

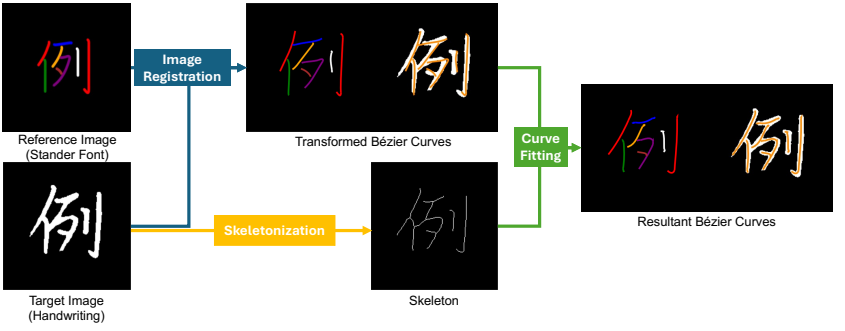


Figure 1: Overview of the proposed stroke extraction method. The process consists of two main stages: (1) image registration aligns the raster input with a reference font represented by Bézier curves, and (2) curve fitting refines the control points to match the target stroke skeleton.

noise and often suffer from stroke breakage at intersections or shape distortion, especially for curved strokes. In addition, preserving curvature often requires a high density of points, resulting in large data volume. Semantic segmentation methods output pixel masks for each stroke, which are less structured and difficult to edit. In contrast to pixel-based approaches, some methods aim to model strokes directly as parametric curves. BézierSketch [4], for example, fits Bézier curves to online pen trajectories using a bidirectional RNN, but it requires trajectory data and cannot handle raster images. Other methods, such as InkSight [10], are applicable to raster inputs and predict stroke trajectories as coordinate sequences. However, since these outputs are not parametric curves, they lack controllability, editability, and compactness.

To address these limitations, we propose a cubic Bézier curve-based stroke extraction method that compares a handwritten character raster image with its corresponding reference font data. The method proceeds in two stages (overview shown in Figure 1).

1. Image Registration: We use a deep learning-based model to roughly align a handwritten character image (target) with font data, which is represented by cubic Bézier curves (reference). This provides an initial alignment of the reference strokes to the target image.

2. Curve Fitting: Based on the initial alignment, each reference Bézier stroke is precisely fitted to the target image by minimizing the distance to the character's skeleton. Specifically, we re-optimize the control points of each stroke's Bézier segments so that they precisely follow the stroke shapes in the target image, using a distance-minimisation algorithm with smoothness regularization.

Compared to conventional methods, the proposed approach offers several advantages. By leveraging standard reference fonts, it ensures structural consistency and stroke-level correspondence. The use of cubic Bézier curves yields a smooth and compact representation of stroke geometry. Moreover, the extracted strokes are directly editable and interpolatable, making them well-suited for downstream applications such as stroke-based synthesis and OCR data augmentation. To our knowledge, this is the first method that combines reference-

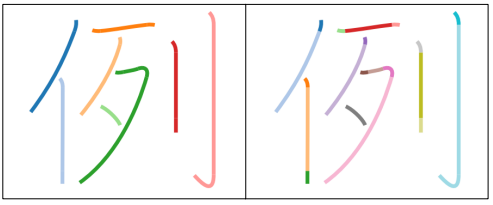


Figure 2: An example from KanjiVG. Left: Each stroke is shown in a different color. Right: Each curve segment is shown in a different color.

guided alignment with parametric Bézier curve fitting from raster images, offering a unified and expressive vector representation for handwritten characters.

The rest of the paper is organized as follows. In Section 2, we introduce the reference font data used in our method. Section 3 explains the image registration method and discusses its limitations. Section 4 details the proposed refinement method, followed by experiments and evaluations in Section 5 demonstrating the effectiveness of our approach.

2 Reference Data: KanjiVG Font Data

KanjiVG [2] is a vector-based font dataset designed for Japanese characters, comprising a total of 6,763 characters including Latin alphabets, kanji (Chinese characters), kana, and symbols. Each character is defined as an ordered sequence of strokes, with the stroke order following standard writing conventions (Figure 2 left). As shown in the right panel of Figure 2, each stroke is composed of one or more connected cubic Bézier curves, referred to as "curve segments." In particular, the start and end points of a stroke are often represented as separate segments, reflecting the pen's entry and exit.

3 Initial Alignment by Image Registration

To obtain an initial alignment between the standard font and the handwritten character image, we adopt the Structure Deformable Image Registration Network (SDNet), a module from the stroke extraction method proposed by Li et al. [9]. SDNet is a deformable image registration network primarily used to align strokes in a reference image (a standard font image) with those in a target image (a handwritten character image) during the stroke extraction process.

The output of SDNet is a transformation field that maps each pixel in the target image to a corresponding location in the reference image. Using this field and applying interpolation, we can compute the coordinates on the target image that correspond to each point in the reference image. Specifically, we used bilinear interpolation to estimate the transformed coordinates of the Bézier control points in the KanjiVG reference. However, as shown in Figure 3, the Bézier curves reconstructed from these transformed control points do not always exactly match the strokes in the handwritten target image.

There are two primary causes of this misalignment. First, the second and third control points of a cubic Bézier curve generally do not lie on the stroke itself, making it difficult for SDNet to capture their displacement correctly. As SDNet is primarily designed to align

(a) (b) (c) (d)

Figure 3: Image registration using SDNet. (a) Reference image (KanjiVG). (b) Target image (ETL). (c) Target image after deformation using SDNet. (d) Original target image overlaid with Bézier curves transformed by SDNet.

observable stroke regions, it cannot fully model the movement of control points that lie outside the stroke.

Second, the overall shape of a cubic Bézier curve depends on the mutual relative configuration of all four control points. Any change in their relative positions—such as angles or distances—during transformation can lead to distortion of the curve's geometry. For example, if part of the curve exhibits a sharp turn or smooth bend, and the transformation fails to maintain the geometric relationships between control points, the resulting curve may lose its original curvature and deviate significantly from the handwritten stroke.

4 Curve Fitting

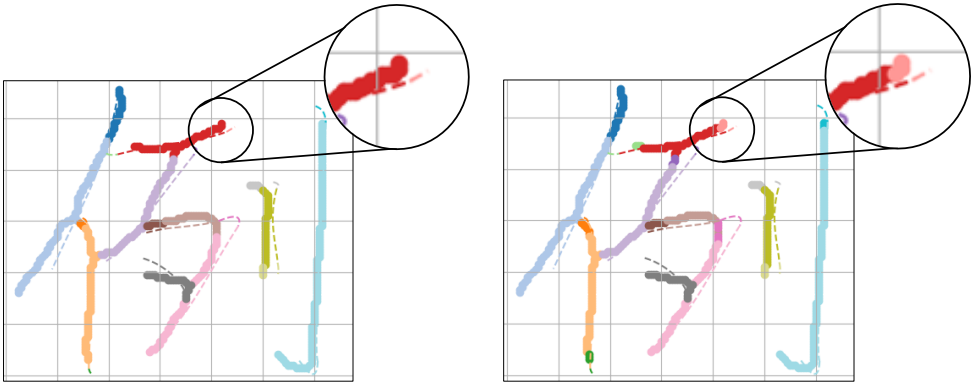
To address the misalignment between transformed Bézier curves and actual strokes in the target image using SDNet, we introduce a refinement step using curve fitting. Specifically, following the initial alignment with SDNet, we enhance the accuracy of the alignment by fitting the curves more precisely to the handwritten character.

First, we extract the skeleton of the target image using the skeletonization algorithm by Zhang and Suen [14]. This algorithm performs thinning on a binarized image to obtain the main structural pixels that constitute the character skeleton as a set of discrete points. We refer to these as target points in the following. Next, to align the transformed Bézier curves with these target points, we compute the closest point on the transformed Bézier curve for each target point and perform point-wise matching. The matching method is based on the algorithm proposed by Schneider [6], which efficiently finds the nearest point on a Bézier curve C to a given point P. It consists of the following steps:

1. Curve approximation: The Bézier curve is first divided into multiple segments. For a given point, the closest segment endpoint is selected as the initial candidate.

2. Optimization by bisection: Starting from the selected candidate, iteratively search for the precise nearest point on the curve using bisection. Specifically, compute the midpoint on the segment corresponding to the curve parameter $t = 0.5$, subdivide the curve at this point, and recursively narrow the range of t to locate the point on the curve closest to P.

3. Flatness evaluation and segmentation: The flatness of each segment is evaluated using the polygon formed by its control points. If a segment is not sufficiently flat, it is



(a) Before (b) After

Figure 4: Necessity of redistributing skeleton points across Bézier segments. Dotted lines indicate Bézier curves initially aligned via SDNet, with colors denoting different curve segments. Skeleton points in the target image are similarly color-coded. (a) Before redistribution: the third segment (pink) of the third stroke lacks skeleton points, preventing accurate fitting. (b) After redistribution: all segments are assigned skeleton points, allowing proper curve fitting.

further subdivided, and the nearest point search is repeated within each subsegment. This allows the method to accommodate complex curves, and the final closest point is chosen from the candidates across all segments.

In addition, to improve the computational efficiency of Schneider's algorithm, we employ R-trees [7] to accelerate the nearest-point search. This optimization resulted in a speed-up of approximately 3.5 times.

After establishing the point-to-curve correspondences, we observed that certain curve segments lacked corresponding skeleton points due to misalignments from SDNet's registration (Figure 4(a)). As a result, these segments could not be optimized during fitting. To address this, we sorted the target points on the skeleton according to the Bézier parameter t of their closest location on the curve, and redistributed them across the multiple curve segments that comprise each stroke. For segments with very few or no assigned points, we transferred a certain number of points from adjacent segments to ensure that each segment had enough points for fitting (Figure 4(b)).

Finally, we refined the alignment by minimizing the error between the Bézier curve and the matched skeleton points. The L-BFGS-B algorithm is used to optimize the control points of each curve by minimizing the following loss function:

$$\text{loss}_{\text{total}} = \frac{1}{N}\sum_{i=1}^{N} \min_{t}(||P_i - \text{Bezier}(t)||^2) + \lambda \sum_{j=1}^{M} ||C_j - C_{j-1}||^2$$

Here, P_i denotes a target point, and $\text{Bezier}(t)$ represents the corresponding point on the Bézier curve. The first term quantifies the average distance between the target points and their corresponding closest points on the Bézier curve, reflecting the overall fitting accuracy. The second term is a smoothness penalty, representing the squared distance between adjacent control points C_j within a single curve, which suppresses sharp angles and abrupt changes.

$\lambda = 0$ $\lambda = 0.5$ $\lambda = 1$

Figure 5: Effect of different smoothness weights λ on curve fitting. Larger values of λ produce smoother curves. However, if λ is too large, strokes with inherent curvature-such as '亅'-may become overly smooth and deviate from their original shape.

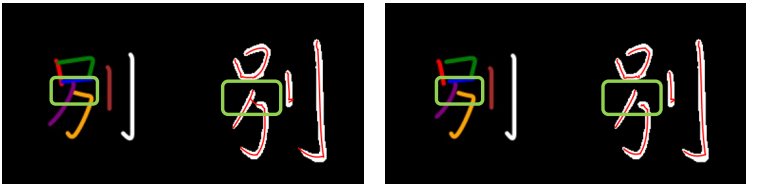


Figure 6: Preprocessing of images. Left: Before. Right: After. In the area highlighted in green, separating overlapping strokes improves the fidelity of the extracted result to the handwritten target image.

The parameter λ controls the strength of the smoothness constraint. By introducing this penalty term, we ensure a smoother curve shape (Figure 5).

5 Experiments

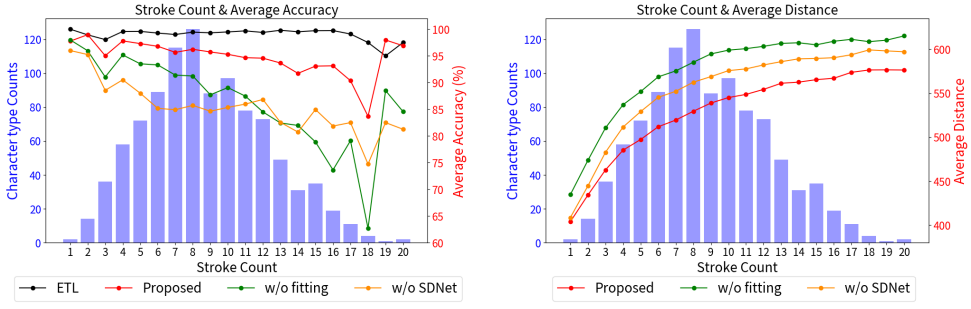
In this section, we introduce the datasets used in our experiments and evaluate the performance of the proposed stroke extraction method.

5.1 Data

We used KanjiVG as the reference font data. To improve the accuracy of stroke fitting, we applied preprocessing to slightly separate strokes that are not connected in the original data. Specifically, we adjusted the parametric range of each stroke so that the first curve segment starts at $t = 0.2$ and the last segment ends at $t = 0.8$, thereby preventing strokes from overlapping at their endpoints (Figure 6). As a result, the relative positioning of strokes is better preserved, leading to improved final fitting results.

For the target images, we used the ETL character database [10]. This database consists of nine subsets (ETL1∼ETL9) and includes approximately 1.2 million character images. From ETL9, we selected 200 samples for each of the 1,000 JIS Level-1 kanji, resulting in a total of 200,000 handwritten character images used in our experiments.

To prepare the target images, we first applied binarization to the handwritten character images using Otsu's binarization method [11], which automatically determines a threshold



(a) OCR Recognition Accuracy (b) Feature-point Matching Distance

Figure 7: Results vs. Stroke Count. The horizontal axis indicates the number of strokes in a character; the bar graph shows the number of character types per stroke count. (a): OCR accuracy on original ETL images (black), proposed method (red), w/o fitting (green), and w/o SDNet (orange). (b): Average distance in feature-point matching for the proposed method (red), w/o fitting (green), and w/o SDNet (orange).

based on the gray-scale histogram of the image.

5.2 Quantitative evaluation

In addition to the proposed method, we evaluated two baseline settings to examine the contribution of each stage:

1. **SDNet initialization only** (abbreviated as *w/o fitting*) — strokes were aligned to the reference font using SDNet without the subsequent curve fitting;

2. **Bounding-box initialization + curve fitting** (abbreviated as *w/o SDNet*) — the KanjiVG reference font was aligned to ETL characters by matching the bounding-box center and dimensions, followed by curve fitting.

These baselines allow us to separately assess the impact of initialization quality and curve optimization. The results of these comparisons are reported in both OCR accuracy and feature-point matching distance.

5.2.1 OCR Results

We first evaluated the quality of the extracted stroke by predicting the character label of images reconstructed from Bézier curves using an OCR model. The OCR model is a four-layer convolutional neural network (CNN), trained on a total of 3,091 character classes, including hiragana, katakana, and kanji, from the ETL database. In addition, to enhance generalization, the training data was augmented through Affine transformation, resulting in approximately 1,698,400 images used for training. We reconstructed character images by rasterizing the extracted Bézier curves. Specifically, each Bézier curve was rendered pixel by pixel based on its control points. These reconstructed images were then input into the OCR model to predict their character labels.

On unseen character images from the ETL dataset (excluded from training), the OCR model achieved an accuracy of 99.36%. In contrast, for character images reconstructed from

Figure 8: Stroke extraction results. Left: Target image. Middle: Extracted strokes overlaid on the target image. Right: Character image reconstructed from extracted Bézier curves.

the Bézier curves extracted by our method, the recognition accuracy was 95.58%, demonstrating that the extracted strokes retain the majority of the discriminative visual information.

We further observed that the OCR accuracy for reconstructed images tended to decrease as the number of strokes increased (Figure 7(a)). This trend was not observed on the original ETL images, suggesting that stroke extraction becomes more challenging for complex characters.

Regarding the baselines, w/o fitting achieved accuracy close to our method for very simple characters (1–2 strokes) but showed a sharp decline as the stroke count increased, dropping below 80% for characters with more than 12 strokes. This highlights the critical role of curve fitting in preserving structural fidelity as character complexity grows. In contrast, w/o SDNet maintained moderate accuracy across stroke counts but consistently lagged behind our method, especially for complex characters (10+ strokes). This indicates that SDNet-based initialization provides a more reliable starting alignment than a simple bounding-box match. Overall, our full method outperformed both baselines across nearly all stroke counts.

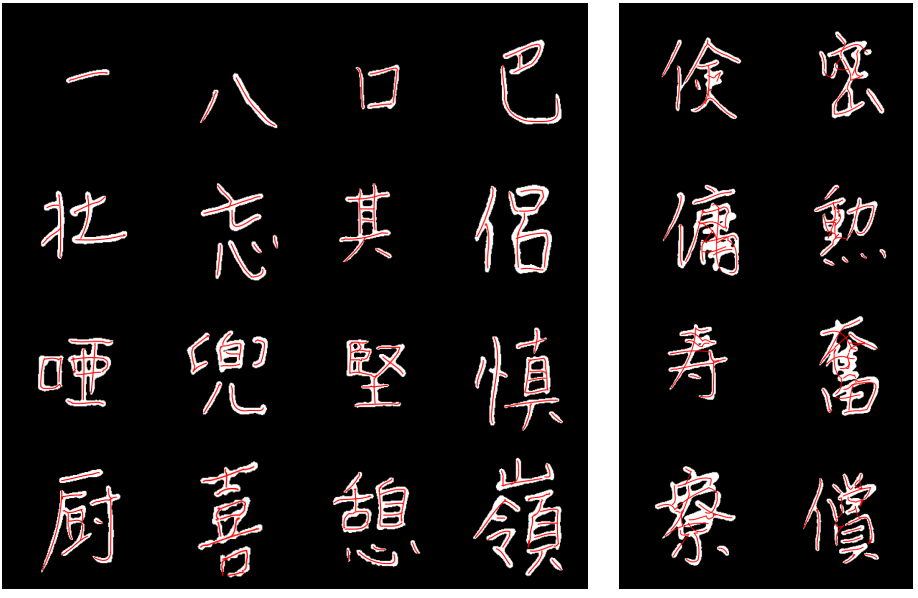
5.2.2 Feature Matching Results

While OCR accuracy provides a high-level evaluation of class separability, it does not always reflect the structural fidelity of the extracted strokes. For instance, in the case of the character '龍', which contains 19 strokes, the OCR model predicted the correct label; however, visual inspection revealed that the strokes were not extracted correctly (Figure 8). This indicated that high OCR accuracy does not necessarily imply high stroke extraction quality.

To obtain a more direct and geometric evaluation, we employed feature-point matching between the reconstructed image from extracted strokes and the original target image. The evaluation consists of three steps: feature point detection, feature description, and matching. For feature detection, we used AKAZE [2], which efficiently detects salient points such as corners and edges in nonlinear scale space, making it well-suited for complex character shapes. Next, we extracted feature descriptors based on the detected keypoints. Each descriptor is a 61-dimensional real-valued vector that represents local image features around the point. Finally, we used Brute-Force matching to compute the Euclidean distance between feature descriptors in the reconstructed and original images. The smaller the average descriptor distance, the more geometrically similar the two images are, indicating better stroke extraction accuracy.

As shown in Figure 7(b), the average descriptor distance increases with stroke count for all methods, reflecting the greater difficulty of stroke extraction in more complex characters.

Our method consistently achieved the smallest distances, demonstrating the highest geometric fidelity. In contrast, *w/o fitting* produced the largest errors overall, with distances



(a) succeeded (b) failed

Figure 9: Example of stroke extraction results

Figure 10: Examples of initial SDNet alignments that resulted in extraction failures.

growing rapidly as the stroke count increased (up to 615.17 at 20 strokes), highlighting the essential role of curve fitting in preserving structural accuracy. *w/o SDNet* performed better than SDNet-only initialization, thanks to the optimization step, but still lagged behind our method across all stroke counts. This indicates that high-quality initialization from SDNet is crucial for achieving accurate feature alignment.

5.3 Qualitative Evaluation

To complement the quantitative evaluation, we visually assessed the quality of extracted strokes. Figure9(a) shows some examples of successful extraction results, while Figure9(b) presents examples of failed extractions.

As noted in the previous section, characters with fewer strokes generally yield better extraction results. However, we also observed cases where the method performs well even for moderately complex characters, such shown in the bottom row of Figure9(a). In contrast, extraction failures were primarily caused by significant misalignments introduced during the initial image registration step using SDNet. Figure 10 shows examples in which incorrect

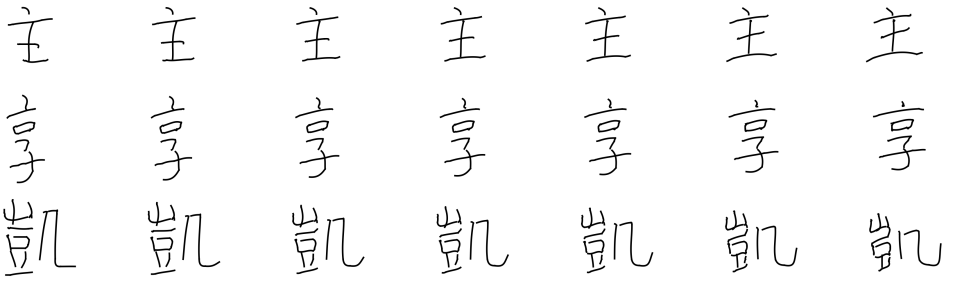


Figure 11: Linear interpolation between two real handwritten samples using extracted Bézier curves. Each row shows one example. From left to right: real sample 1, interpolated samples at interpolation weights 0.2, 0.4, 0.5, 0.6, 0.8, and real sample 2.

initial alignment leads to poor extraction outcomes.

In addition to evaluating the stroke extraction quality, we also tested the applicability of the extracted Bézier curves for character synthesis. Specifically, we conducted linear interpolation between two real handwritten samples of the same character by interpolating the control points of corresponding strokes. As shown in Figure 11, this process generates smooth and realistic intermediate character variants that preserve stroke continuity and structure. These results demonstrate the generative potential of our method and its suitability for producing diverse samples for data augmentation.

6 Conclusion and Future Work

In this study, we proposed a method to extract strokes represented by cubic Bézier curves from raster images of handwritten characters, and evaluated its effectiveness. The OCR accuracy on characters reconstructed from the extracted curves was over 95%, which suggests our extraction results capture the overall shapes of characters correctly.

Our results however indicate that the OCR accuracy tends to decline as the number of strokes increases, suggesting that stroke extraction for complex characters remains a challenge. To address this issue, we believe that incorporating character feature points as additional constraints in the initial alignment will be effective. Feature points can help guide the position and shape of strokes—particularly at endpoints and curved regions—and are expected to improve extraction accuracy in the presence of complex structures or stroke intersections.

Furthermore, the extracted Bézier representations support stroke-level operations such as interpolation, enabling the synthesis of diverse handwritten character variants for potential use in OCR training and data augmentation.

In future work, we aim to build on this framework to develop a high-quality handwritten character generation model that leverages the extracted stroke representations to produce diverse, natural-looking samples with controllable style variation.

Acknowledgement This work was supported by JSPS KAKENHI Grant Number 21H04416.

References

[1] URL http://etlcdb.db.aist.go.jp/?lang=ja.

[2] URL https://kanjivg.tagaini.net.

[3] Cyrille Berger. Strokes detection for skeletonisation of characters shapes. In *Advances in Visual Computing*, pages 510–520, Cham, 2014. Springer International Publishing. ISBN 978-3-319-14364-4.

[4] Ayan Das, Yongxin Yang, Timothy Hospedales, Tao Xiang, and Yi-Zhe Song. BézierSketch: A generative model for scalable vector sketches. In *Computer Vision – ECCV 2020*, pages 632–647, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58574-7.

[5] Kuo-Chin Fan, Den-Fong Chen, and Ming-Gang Wen. Skeletonization of binary images with nonuniform width via block decomposition and contour vector matching. *Pattern Recognition*, 31(7):823–838, 1998. ISSN 0031-3203.

[6] Andrew S. Glassner, editor. *Graphics gems*. Academic Press Professional, Inc., USA, 1990. ISBN 0122861695.

[7] Antonin Guttman. R-trees: a dynamic index structure for spatial searching. In *Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data*, pages 47–57, 1984. ISBN 0897911288. doi: 10.1145/602259.602266.

[8] F. Le Bourgeois and H. Emptoz. Skeletonization by gradient regularization and diffusion. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 2, pages 1118–1122, 2007. doi: 10.1109/ICDAR.2007.4377089.

[9] Meng Li, Yahan Yu, Yi Yang, Guanghao Ren, and Jian Wang. Stroke extraction of chinese character based on deep structure deformable image registration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 1360–1367, 2023.

[10] Blagoj Mitrevski, Arina Rak, Julian Schnitzler, Chengkun Li, Andrii Maksai, Jesse Berent, and Claudiu Cristian Musat. Inksight: Offline-to-online handwriting conversion by teaching vision-language models to read and write. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=pSyUfV5BqA.

[11] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66, 1979. doi: 10.1109/TSMC.1979.4310076.

[12] Adrien Bartoli Pablo Alcantarilla, Jesus Nuevo. Fast explicit diffusion for accelerated features in nonlinear scale spaces. In *Proceedings of the British Machine Vision Conference*, pages 13.1–13.11. BMVA Press, 2013.

[13] James Parker. *Algorithms for Image Processing and Computer Vision*. Wiley Publishing, Inc., 2nd edition, 2010.

[14] T. Y. Zhang and C. Y. Suen. A fast parallel algorithm for thinning digital patterns. *Commun. ACM*, 27(3):236–239, March 1984. ISSN 0001-0782. doi: 10.1145/357994.358023. URL https://doi.org/10.1145/357994.358023.