U.P.B. Sci. Bull., Series C, Vol. 88, Iss. 1, 2026 ISSN 2286 – 3540

RESEARCH ON IMAGE LINE DRAWING GENERATION ALGORITHM BASED ON DIRECTION FIELD CONSISTENCY CONSTRAINTS

Yanrong LIU[1], Lijun REN[2,*], Ru CHEN[3], Yao LIU[4]

Image style transfer has been widely applied in the field of computer vision, primarily aiming to transform captured images into specific artistic styles using computational techniques. Traditional algorithms, such as those based on structural tensors and ETF (Edge Tangent Flow) or combining Kuwahara filtering with line integral convolution, rely heavily on heuristic construction by researchers' domain knowledge or experience. Most style transfer methods depend on sample-based training, resulting in high computational costs and slow processing speeds. To address these limitations, this study proposes an image line drawing generation algorithm under direction field consistency constraints. The proposed approach preprocesses images using non-local means filtering, computes the direction field through Tikhonov regularization accelerated by the Sherman-Morrison-Woodbury formula, and guides DoG (Difference of Gaussians) filtering with the regularized direction field. A perception threshold algorithm, designed based on human visual system characteristics, binarizes the filtered results to generate line drawings. Simulation experiments demonstrate that the algorithm effectively transforms input images into line drawings with smooth lines and preserves the primary information of the original images while significantly improving computational efficiency.

Keywords: Image style transfer, direction field, regularization, DoG filtering, perception threshold algorithm

1. Introduction

Image style conversion is a crucial research direction in computer vision, aiming to transform ordinary images into specific artistic styles (e.g., oil paintings, watercolor, sketches) using computational techniques. This technology not only provides new tools and inspiration for artistic creation but also holds broad application prospects in image editing, film and television production, and game development. In recent years, with the rapid development of deep learning, significant progress has been made in neural network-based image style conversion methods, which can generate realistic and artistically appealing images [1].

[1] Shaanxi Institute of International Trade & Commerce, Xi'an, Shaanxi, 712046, China
[2]* Shaanxi Institute of International Trade & Commerce, Xi'an, Shaanxi, 712046, China, corresponding author, e-mail: 20061007@csiic.edu.cn
[3] Shaanxi Institute of International Trade & Commerce, Xi'an, Shaanxi, 712046, China
[4] Shaanxi Institute of International Trade & Commerce, Xi'an, Shaanxi, 712046, China

However, these methods typically require large amounts of training data, incur high computational costs, and struggle to control stylistic details, limiting their practical applicability. As a concise yet expressive art form, line drawings effectively capture the primary features and structural information of images. Recently, line drawing-based image style conversion has garnered increasing attention. Line drawings not only highlight the main contours of subjects but also convey details and textures through variations in line thickness, density, and direction [2], offering unique advantages in artistic creation and image processing. Nevertheless, existing line drawing generation algorithms often suffer from issues such as fragmented lines, detail loss, and slow computation [3], failing to meet practical demands. Particularly when processing high-resolution images, algorithms often face trade-offs between efficiency and output quality [4-5].

Recent advancements in convolutional neural network (CNN)-based image style conversion have achieved notable success. Pan et al. proposed using pre-trained CNNs to separately process content and style in natural images [6-7], yielding impressive, stylized results. However, this method suffers from slow convergence due to excessive iterations. To address this, Johnson et al. (2016) introduced a perceptual loss function to train feedforward networks [8], significantly improving efficiency and quality. However, this approach requires training separate models for different styles, complicating implementation. To simplify model training, Zhong et al. proposed binding partial networks to specific styles [9-10], enabling multi-style conversion through minimal parameter adjustments. Despite these improvements, such methods remain constrained by limited style adaptability and necessitate retraining for novel styles. Moreover, heuristic methods—though avoiding training—often exhibit slow computation, insufficient detail preservation, and stylistic monotony. Traditional algorithms also struggle with noise and distortion when handling complex textures or detailed images, failing to meet the demands of high-quality style conversion.

To overcome these limitations, this study proposes a direction field consistency-constrained line drawing generation algorithm comprising four key steps: First, Preprocessing with Non-Local Means Filtering: Eliminate noise while preserving critical details in input images. Second, Direction Field Calculation and Regularization: Compute image direction fields using gradient information, followed by Tikhonov regularization accelerated via the Sherman-Morrison-Woodbury formula to enhance smoothness and consistency. Third, DoG Filtering Guided by Regularized Direction Fields: Enhance edges and suppress noise in preprocessed images using difference-of-Gaussians (DoG) filtering guided by the regularized direction field. Fourth, Perceptual Thresholding Based on Human Visual System (HVS): Design an adaptive thresholding algorithm inspired by HVS nonlinearities to binarize filtered results, generating high-quality line drawings. This pipeline ensures detail retention while significantly improving computational efficiency, addressing the trade-offs between speed and quality inherent in traditional methods.

2. Related Knowledge

2.1 Line Drawing

The human visual system possesses a remarkable ability to abstract observed objects into lines, a capability vividly exemplified in Chinese pictographic characters. The fundamental building block of an image lies in its grayscale values, where information is conveyed through variations in these values. An image with uniform grayscale values across all pixels contains no meaningful information, whereas regions with abrupt grayscale changes or discontinuities often correspond to the edges of semantic objects, which can be delineated through lines. Consequently, lines serve as core elements in artistic composition, making line drawing the oldest and most foundational form of visual art. From an artistic perspective, the unique visual effects of line drawings have shaped distinct modes of artistic design thinking. In modern design, whether applied to planar or three-dimensional works, realistic or abstract creations, line drawings enable artists to express emotions, demonstrate observational skills, creativity, and imagination, thus becoming a vital medium for emotional communication.

From an image processing standpoint, line drawings exhibit the following characteristics: First, they are binary images, where pixel values are restricted to two possibilities (typically black lines on a white background), inheriting all properties of binary imagery. Second, line drawings employ smooth lines to outline the semantic boundaries of objects. While more simplified than the original image, they effectively preserve key information. This aligns line drawing techniques with edge detection, as edge detection operators can also be utilized in line drawing generation algorithms. Beyond artistic applications, line drawings hold significant value in practical image processing tasks.

2.2 Edge Detection

Edge detection is a critical technique in image processing, aimed at identifying pixels where grayscale values undergo significant changes or discontinuities, typically corresponding to object boundaries or contours in an image [11]. Common edge detection operators include the Canny, Sobel, Prewitt, and Roberts operators [12-13], as detailed below. The Sobel, Prewitt, and Roberts operators share similar principles: they approximate horizontal and vertical directional derivatives through template convolution and generate binary edge images by thresholding gradient magnitudes [14]. The Sobel and Prewitt operators employ 3×3 templates to measure grayscale variations in horizontal and vertical directions, with the Sobel operator placing greater emphasis on central pixels. In contrast, the Roberts operator uses 2×2 templates to assess grayscale changes along primary and secondary diagonal directions [15]. However, these three operators lack preprocessing steps, rendering them highly sensitive to noise due to their reliance on derivative operations [16].

$$G_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad G_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \qquad \text{(Roberts operators)}$$

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad G_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \qquad \text{(Prewitt operators)}$$

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \qquad \text{(Sobel operators)}$$

In contrast to other operators, the Canny edge detection process integrates five key steps to achieve a balance between noise suppression and edge localization: it first applies Gaussian smoothing to reduce noise while preserving directional derivative robustness; subsequently computes gradient magnitude and orientation to determine edge strength and direction; employs non-maximum suppression (NMS) to retain only local maxima pixels, refining edge continuity; utilizes dual-threshold processing to classify strong and weak edges, distinguishing meaningful edges from noise; and finally performs edge linking via hysteresis thresholding to connect weak edges to strong ones, generating continuous and precise edge maps [17-18]. Grounded in rigorous theoretical foundations, the Canny operator outperforms traditional methods in edge accuracy and noise resilience, demonstrating its superiority in both academic research and industrial applications [19].

2.3 Direction Field Computation and Regularization

Direction field computation is a critical step in image processing, aiming to extract dominant directional information within local image regions, typically employed for analyzing edge orientation distributions. Initially, gradient operators are applied to compute the horizontal gradient G_x and vertical gradient G_y for each pixel in the image, followed by the derivation of gradient magnitude $\mathrm{G} = \sqrt{\mathrm{G}_x^2 + \mathrm{G}_y^2}$ and gradient orientation$\theta = \arctan\left(\frac{\mathrm{G}_y}{\mathrm{G}_x}\right)$. The gradient magnitude reflects edge strength, while the gradient orientation characterizes the local directional properties of edges. Subsequently, the image is partitioned into localized patches, and statistical metrics of gradient orientations within each patch are calculated [20]. The initial direction field computation formula 1 is formulated as follows:

$$\theta_i^0 = \frac{arctan\left(\frac{\sum G_{y,i}^2 - G_{x,i}^2}{2\sum G_{x,i} G_{y,i}}\right)}{2} \tag{1}$$

Due to image noise and local complexity, the initial direction field may exhibit inconsistencies and errors. To optimize the smoothness and consistency of the direction field, regularization methods are commonly employed to constrain and refine the optimization process. Specifically, the direction field regularization problem is formulated as an optimization task, where the objective function is typically defined as:

$$E(\theta) = \sum_i (\theta_i - \theta_i^0)^2 + \lambda \sum_{i,j} \left(\theta_i - \theta_j\right)^2 \tag{2}$$

To accelerate matrix inversion computations during regularization, the Sherman-Morrison-Woodbury formula is employed in conjunction with Tikhonov regularization, which effectively mitigates noise and inconsistencies in the direction field, thereby achieving smoother and more consistent directional field outputs.

3. Image Line Drawing Generation Algorithm Based on Direction Field Consistency Constraints

Image stylization refers to the process of generating new artistic-style images from naturally captured inputs while preserving essential original information, such as converting natural scene photographs into corresponding line drawings. Traditional image stylization algorithms suffer from excessive dependency on sample data, limiting their generalizability. This section focuses on investigating style transformation in images using traditional methodologies, with a specific emphasis on the "line drawing" style. We propose a novel algorithm based on direction field consistency constraints to address these limitations and achieve high-fidelity stylistic conversion.

3.1 Preprocessing

Denoising constitutes a critical step in preprocessing for image stylization, as noise contamination can degrade stylization outcomes. A comparative analysis of conventional filtering algorithms demonstrated that preprocessing input images with the NLM (Non-Local Means) filter yields optimal performance. The implementation methodology is detailed as follows:

Let the noisy input image be denoted as $v = \{v(i) \mid i \in \Omega\}$, the NLM-filtered output as $u = \{u(i) \mid i \in \Omega\}$, where Ω represents a specific image region and i denotes a pixel index. Under these definitions, $u(i)$ can be formally expressed as:

$$u(i) = \frac{\sum_{j \in \Omega_i} w(i,j) v(j)}{\sum_{j \in \Omega_i} w(i,j)} \tag{3}$$

Here, Ω_i denotes a square neighborhood centered at pixel i with radius t. $w(i,j)$ represents the weight assigned to $v(j)$, and $\sum_{j \in \Omega_i} w(i,j)$ serves as the normalization factor, ensuring that the sum of weights equals 1. Typically, the following is adopted:

$$w(i,j)=\exp(-\frac{d(i,j)}{h^2}) \tag{4}$$

Where in:

$$d(i,j)=\left\|N(i)-N(j)\right\|_{2,\alpha}^{2} \tag{5}$$

In formula 5, $N(i)$ and $N(j)$ denote image patches centered at pixels i and j, respectively, with a radius of f. Here, $\left\|\cdot\right\|_{2,\alpha}^{2}$ represents the Gaussian seminorm, and h denotes the filtering parameter.

In the NLM algorithm, the smoothness of filtering is governed by parameters h and t. Parameter h determines the influence of similarity measure $d(i,j)$ on weight $w(i,j)$, where increasing h reduces the impact of $d(i,j)$ on $w(i,j)$, thereby enhancing filter smoothness. Conversely, decreasing h diminishes smoothness. Parameter t controls the number of pixels involved in weighting, with larger t values further augmenting filtering smoothness.

3.2 Tikhonov Regularization of Direction Fields and Acceleration of Regularization Algorithms

(1) Direction Field Regularization

Tikhonov regularization is commonly employed to address ill-posed problems, such as image scaling, where the solution relies critically on the availability of prior models. Similarly, for direction fields, a prior model is constructed by minimizing the objective function defined in Formula (6). Let the directional vector at pixel i be denoted as $\boldsymbol{G(i)}$. For a square neighborhood i centered at t with radius Ω_i, the directional vectors of all pixels within E are represented as $\boldsymbol{G}(j),\ j\in\Omega_i$. Given the spatial stationarity of images and the prevalence of self-similar structures, $\boldsymbol{G}(i)$ can be linearly expressed as $\boldsymbol{G}(j), j\in\Omega_i$, i.e.,

$$\boldsymbol{G}(i)=\sum_{j=1}^{(2t+1)^2}\beta_j\boldsymbol{G}(j) \tag{6}$$

Here, $\beta_j, j=1,2,\cdots,L$,and $(2t+1)^2$ are coefficients, and their matrix representation is expressed as:

$$G(i)=G\cdot\boldsymbol{W} \tag{7}$$

Here, G is the matrix of $4\times(2t+1)^2$ with column vector $\boldsymbol{G}(j),\ j\in\Omega_i$, denoted as $W=\left(\beta_1,\beta_2,\cdots,L,\beta_j,\cdots,L,\beta_{(2t+1)^2}\right)^T$. Note that since pixel location i is the centroid of region Ω_i, $\boldsymbol{G}(j),\ j\in\Omega_i$ contains $\boldsymbol{G}(i)$, leading to trivial solutions in Equations (6) and (7) (achieved by setting the coefficient of $\mathrm{G}(i)$ to 1 while others to 0). To avoid trivial solutions, we employ the following Tikhonov regularization model to solve the representation coefficients.

$$\overset{\wedge}{\boldsymbol{W}} = \arg\min_{\boldsymbol{W}} \|\boldsymbol{G}(i) - G\cdot\boldsymbol{W}\|_2^2 + \lambda\|\boldsymbol{W}\|_2^2 \tag{8}$$

In the above formulation, the first term represents the fitting fidelity that minimizes the discrepancy between $G(i)$ and $G\cdot\boldsymbol{W}$. The second term corresponds to the regularization term, which enforces the L2-norm of the coefficient vector $\mathbf{W}$ in the linear representation to be minimized. Here, λ serves as the regularization parameter that balances the trade-off between the two terms. By taking the derivative of the objective function with respect to $\mathbf{W}$ and setting it to zero, we derive:

$$\begin{aligned} & -G^T\boldsymbol{G}(i) + G^TG\boldsymbol{W} + \lambda\boldsymbol{W} = 0 \\ \Leftrightarrow\ & (G^TG + \lambda i)\boldsymbol{W} = G^T\boldsymbol{G}(i) \\ \Leftrightarrow\ & \overset{\wedge}{\boldsymbol{W}} = (G^TG + \lambda I)^{-1}G^T\boldsymbol{G}(i) \end{aligned} \tag{9}$$

Let $G(i)$ denote the input variable, and $\overset{\wedge}{\boldsymbol{G}}(i) = [\hat{G}^1(i), \hat{G}^2(i), \hat{G}^3(i), \hat{G}^4(i)]^T$ represent the result of applying the regularization process to $G(i)$. so, $\overset{\wedge}{\boldsymbol{G}}(i) = G\cdot\hat{\boldsymbol{W}}$.

(2) Acceleration of Regularization Algorithms

Direction field regularization requires per-pixel computation of the inverse of an n×n matrix (Equation (9)). When n is large, this significantly increases computational overhead. For example, with n=8 in a 512×512-pixel image, 262,144 inversions of 8×8 matrices are required, causing severe computational bottlenecks. To address this, the Sherman-Morrison-Woodbury (SMW) matrix identity [21] is introduced, combined with Tikhonov regularization to accelerate matrix inversion. This approach simultaneously suppresses noise and inconsistencies in the direction field while enhancing computational efficiency, ultimately yielding a smooth and consistent directional field output.

$$(A^{-1} + B^TB)^{-1}B^T = AB^T(BAB^T + I)^{-1} \tag{10}$$

Let $A = \lambda^{-1}\boldsymbol{I}$ and B=G. Substituting these values into Formula 10, we obtain:

$$\left(G^TG + \lambda\boldsymbol{I}\right)^{-1}G^T = \lambda^{-1}\boldsymbol{I}G^T\left(\lambda^{-1}GIG^T + \boldsymbol{I}\right)^{-1} \tag{11}$$

Substituting Formula 11 into Formula 9 yields:

$$\hat{\boldsymbol{W}} = \lambda^{-1}\boldsymbol{I}G^T\left(\lambda^{-1}G\boldsymbol{I}G^T + \boldsymbol{I}\right)^{-1}\boldsymbol{G}(i) \tag{12}$$

A comparison of Formula 9 and Formula 12 reveals that while Formula 9 necessitates computing the inverse of matrix $(2t+1)^2\times(2t+1)^2$, Formula 12 only requires calculating the inverse of matrix 4×4, regardless of the value of t. Consequently, when t assumes large values, formula (12) enables significant acceleration of the algorithm.

3.3 Regularized Orientation Field-Guided Difference of Gaussians (DoG) Filtering

In natural images, the edges of semantic objects typically correspond to regions with significant intensity variations. Therefore, to accurately delineate the edges of semantic objects, it is essential to first quantify the intensity variations in the spatial domain. Theoretically, any high-pass filter can be employed to measure intensity variations. This paper adopts the classical edge detection paradigm and selects the one-dimensional Difference of Gaussians (DoG) filter to quantify intensity changes. The DoG filter employs the following kernel function:

$$K(r) = \boldsymbol{G}_{\alpha 1}(\mathrm{r}) - \boldsymbol{G}_{\alpha 2}(\mathrm{r}) \tag{13}$$

where

$$G_{\alpha}(x) = \frac{1}{\sqrt{2\pi\alpha}} e^{-\frac{x^2}{2\alpha^2}} \tag{14}$$

It can be observed that $K(r)$ represents the difference between two zero-mean Gaussian functions. As noted in [9], the DoG (Difference of Gaussians) filter effectively mimics the response of human retinal cells to intensity variations and adopts parameter $\alpha_2 = 1.35\alpha_1$ as suggested. To enhance numerical stability, the kernel function from [7] is implemented in the DoG filtering process:

$$K(r) = G_{\alpha 1}(r) - (1-\varepsilon)\cdot G_{\alpha 2}(r) \tag{15}$$

Where $\varepsilon = 0.001$.

Natural images exhibit anisotropic properties in the spatial domain, where intensity variations are larger perpendicular to edges and smaller parallel to edges. At each pixel i, the orientation vector $\hat{\boldsymbol{G}}(i) = (\hat{G}^1(i), \hat{G}^2(i), \hat{G}^3(i), \hat{G}^4(i))^T$ contains four components that quantify intensity gradients in horizontal, 45°, vertical, and 135° directions. The direction $\max(\hat{\boldsymbol{G}}(i))$ (denoted as the maximum absolute value among the four components of $\hat{\boldsymbol{G}}(i)$ corresponds to the dominant gradient direction. A one-dimensional DoG filter is applied along $\max(\hat{\boldsymbol{G}}(i))$ for preprocessing the image. During implementation, the DoG kernel function is discretized. Since the filtering direction at each pixel is determined by the orientation vectors $\hat{\boldsymbol{G}}(i)$ and $i \in \Omega$, this method is termed regularized orientation field-guided difference of gaussians (DoG) filtering. The filtered output is denoted as:

$$B(i) = DoG(\hat{\boldsymbol{G}}(i), u), i \in \Omega \tag{16}$$

3.4. Design of the Perceptual Threshold Algorithm

Thresholding the RDF-DoG filtering results can effectively extract the edges of semantic objects. However, traditional thresholding algorithms fail to capture the nonlinear characteristics of the Human Visual System (HVS). The RDF-DoG filtering output quantifies objective intensity changes (physical quantities), whereas the HVS perceives subjective changes (psychological quantities). The HVS exhibits nonlinear perception of intensity variations: below a certain threshold, perceptual changes remain undetected; above the threshold, perceived changes intensify with increasing physical variations; but beyond a critical value, perceptual changes saturate. Here, the hyperbolic tangent function is employed to model the relationship between the objective intensity change $(B(i), i \in \Omega)$ and the HVS-perceived change, defined as:

$$H(x) = \tanh(x) + 1 = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} + 1 \tag{17}$$

To delineate the edges of semantic objects, thresholding is applied to the subjective quantity (Human Visual System, HVS)-perceived changes, adopting the following thresholding function:

$$\text{threshold}(B(i)) = \begin{cases} 0 & \text{if } H(B(i)) < \tau \\ 1 & \text{else} \end{cases}, i \in \Omega \tag{18}$$

Here, τ denotes the threshold parameter with a normal range of $\tau \in [0,1]$, indicating that regions where the HVS-perceived intensity changes exceed a certain threshold are identified as semantic object edges.

4. Simulation Experiments and Results Analysis

To validate the effectiveness of the proposed algorithm, experiments were conducted on a 64-bit Windows 7 operating system with a Lenovo computer equipped with a 3.6 GHz Intel CPU and 8 GB RAM. The algorithm was implemented using MATLAB R2016a. This experimental setup sufficiently supports algorithm performance evaluation and result analysis. To investigate factors influencing the generation quality of line-drawn style images, the experimental design primarily consists of two parts: (1) a time complexity comparison experiment for the regularized acceleration algorithm, aiming to evaluate the computational efficiency of the regularization optimization strategy; (2) a threshold parameter comparison experiment for line-drawn image generation, analyzing how different threshold settings affect the quality of the final line-drawn style images. Through these experiments, the performance of each algorithmic module and their contributions to the generated results are systematically validated.

(1) Time Complexity Comparison of the Regularized Acceleration Algorithm
To validate the impact of the regularized algorithm acceleration on experimental results, this study selects six standard test images, including Building, as the experimental dataset. To ensure the accuracy of runtime measurement, each image was executed 10 times, and the average runtime was recorded as the execution time of the Tikhonov regularization algorithm. *Table 1* presents the time complexity comparison results of the six images before and after applying the Tikhonov regularization algorithm acceleration, aiming to quantitatively analyze the improvement in computational efficiency achieved by the acceleration strategy.

Table 1
Time Complexity Comparison of the Regularized Acceleration Algorithm

Image Name	Resolution	Number of Trials	Tikhonov Regularization Average Runtime	Tikhonov Regularization Accelerated Average Runtime	Speedup Ratio (Before/After)
Building	512×512	10 trials	312.34 s	15.42 s	20.10
Flower	512×512	10 trials	298.12 s	15.38 s	19.80
Blue-Car	512×512	10 trials	190.42 s	8.49 s	22.03
Player	512×512	10 trials	310.04 s	13.98 s	21.12
EV	512×512	10 trials	300.12 s	14.37 s	18.09
Female	512×512	10 trials	297.89 s	14.96 s	20.04

The experimental results demonstrate that the Tikhonov regularization acceleration strategy exhibits significant performance improvements across different images. For six images with a uniform resolution of 512×512, the post-acceleration runtime was reduced to approximately 1/20 of the original time on average. Specifically:

- The *Building* image's regularization runtime decreased from 312.34 seconds to 15.42 seconds, achieving a **20.10× speedup**.
- The *Flower* image's runtime improved from 298.12 seconds to 15.38 seconds (**19.80×speedup**).
- The *Blue-Car* image saw the highest acceleration, with a runtime reduction from 190.42 seconds to 8.49 seconds (**22.03×speedup**).
- The *Player*, *EV*, and *Female* images achieved speedups of **21.12×speedup**

(310.04s → 13.98s), **18.09×speedup** (300.12s → 14.37s), and **20.04×speedup** (297.89s → 14.96s), respectively.

These findings indicate that the Tikhonov regularization acceleration strategy substantially reduces computational overhead when processing images of identical resolution, with an average runtime reduction of ~1/20. This optimization significantly enhances experimental efficiency, particularly for high-resolution images, thereby providing robust support for subsequent image processing tasks.

(2) Comparative Analysis of Threshold Parameters' Influence on Line-Drawn Image Generation Quality

To validate the effectiveness of the proposed line-drawn image extraction algorithm, extensive experimental studies were conducted. In addition to a comprehensive analysis of the threshold parameter in formula (18), all algorithmic parameters were examined. The following focuses on the selection of the threshold parameter in Equation (18). Fig. 1. illustrates the algorithm's output under varying threshold values. Comparative analysis reveals that threshold parameter $\tau = 0.5$ significantly impacts the generation quality of line-drawn images.

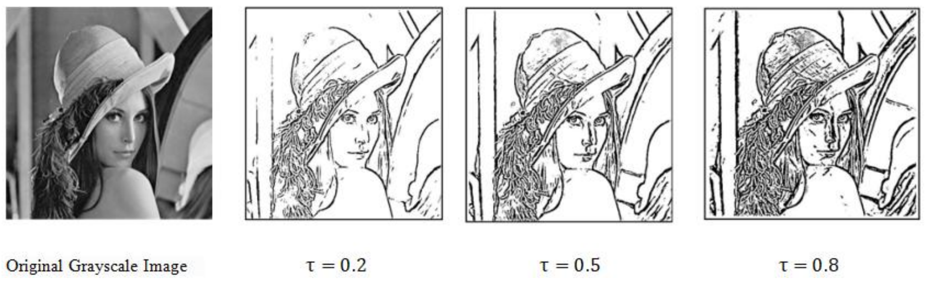

Fig. 1. Line-drawn Images of the Original Grayscale Image under Different Parameter τ Values

The experimental results demonstrate that the threshold parameter τ significantly impacts the algorithm's output. As τ increases, the edge detection process becomes more sensitive to noise, leading to excessive noise inclusion in edge information and degrading the visual quality of the output. Conversely, when A is set too low, the algorithm tends to omit partial details from the original image, resulting in incomplete generated results. Through experimental validation, this study sets τ=0.5 to balance noise suppression and detail preservation.

To further verify the algorithm's effectiveness, six natural scene images (including Building) were selected as test samples (Fig. 2.). Extensive experiments were conducted, and the results are shown in Fig. 3. The proposed algorithm generates line-drawn contours of semantic object edges using concise and smooth lines, effectively emphasizing key visual information while maintaining aesthetic appeal. By directly synthesizing line-drawn images from natural inputs, the algorithm produces realistic style outputs, highlighting its advantage in realistic expression.

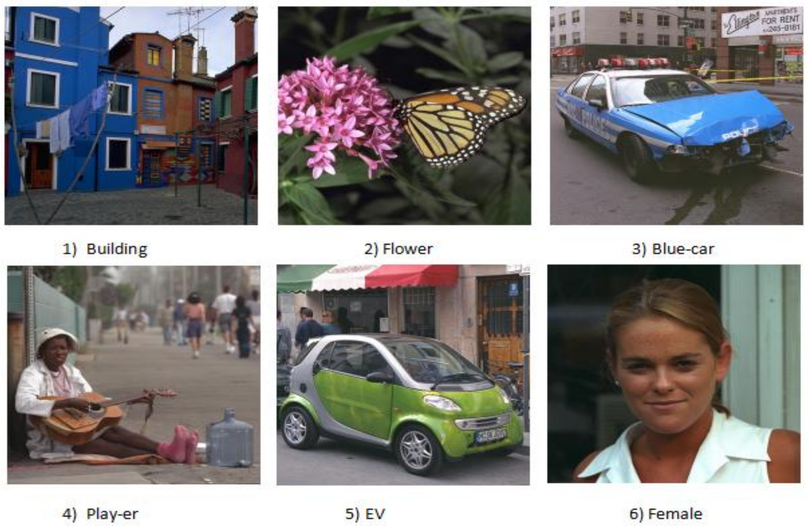

Fig. 2. Test Sample Image

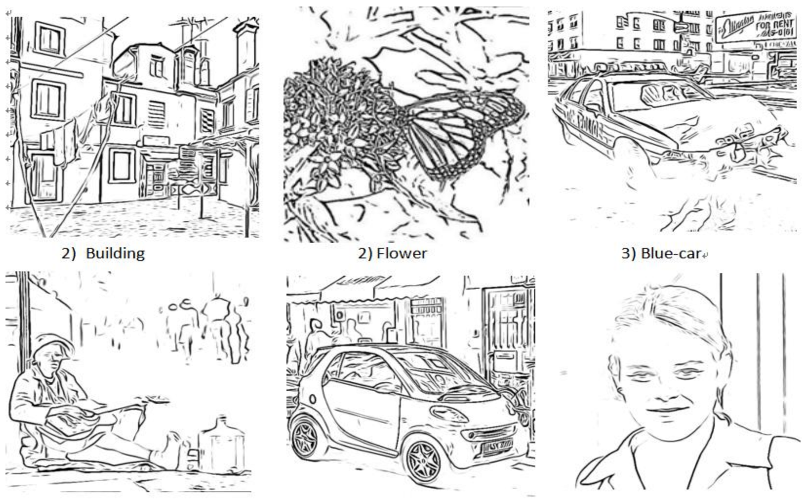

Fig. 3. Generated Images of Experimental Results

5. Conclusion

This study proposes an image line-drawn generation algorithm based on orientation field consistency constraints. Through innovative steps including non-local means filtering, Tikhonov regularization acceleration, regularized orientation field-guided Difference of Gaussians (DoG) filtering, and perceptual thresholding algorithms, the efficiency and quality of line-drawn image generation are significantly enhanced. Experimental results demonstrate that the regularization acceleration strategy reduces computational time to 1/20th of traditional methods while producing line-drawn images with smooth contours that effectively preserve key visual information. When the threshold parameter τ is set to 0.5, the algorithm achieves an optimal balance between noise suppression and detail retention, yielding outputs that exhibit both realistic texture and artistic expressiveness. However, limitations include high parameter dependency (e.g., empirical tuning of threshold τ and regularization parameter η), lack of comparison with deep learning methods to validate generalization capability, and inefficiency in handling high-resolution images. Future research will focus on integrating adaptive parameter optimization mechanisms and deep learning techniques to improve adaptability to complex images. Additionally, cross-domain applications and real-time performance optimization will be explored to facilitate broader adoption of line-drawn generation technology in artistic creation and industrial inspection.

R E F E R E N C E S

[1]. L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2016, pp. 2414–2423.
[2]. J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in Proc. Eur. Conf. Comput. Vis., vol. 9906, 2016, pp. 694–711.
[3]. Y. Li, C. Fang, J. Yang, et al, "Universal style transfer via feature transforms," Adv. Neural Inf. Process. Syst., vol. 34, pp. 386–396, 2021.
[4]. X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in Proc. IEEE Int. Conf. Comput. Vis., 2021, pp. 1501–1510.
[5]. A. Sanakoyeu, D. Kotovenko, S. Lang, et al, "A style-aware content loss for real-time HD style transfer," ACM Trans. Graph., vol. 41, no. 4, pp. 1–15, 2022.
[6]. H. Zhang, Y. Li, and J. Sun, "Dynamic neural style transfer with temporal consistency for video," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2023, pp. 2345–2355.
[7]. D. Y. Park and K. H. Lee, "Arbitrary style transfer with style-attentional networks," IEEE Trans. Pattern Anal. Mach. Intell., vol. 45, no. 7, pp. 8230–8243, 2023.
[8]. M. Chen, L. Wang, and D. Zhang, "Real-time neural style transfer for high-resolution images via lightweight multi-scale architecture," IEEE Trans. Image Process., vol. 33, no. 5, pp. 2100–2112, 2024.
[9]. E. Risser, P. Wilmot, and C. Barnes, "Stable and controllable neural texture synthesis and style transfer," Comput. Vis. Image Underst., vol. 227, p. 103595, Jun. 2023.
[10]. R. Gao and T. Funkhouser, "Physics-informed neural style transfer for material appearance synthesis," ACM Trans. Graph., vol. 42, no. 3, pp. 1–10, 2023.
[11]. S. Liu, C. Yang, and J. Hays, "Universal style transfer via latent space disentanglement," ACM Trans. Graph., vol. 43, no. 2, pp. 1–12, Jan. 2024.
[12]. N. Kolkin, J. Salavon, and G. Shakhnarovich, "Style transfer by relaxed optimal transport and self-similarity," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2022, pp. 10051–10060.
[13]. Y. Jing, Y. Yang, Z. Feng, *et al*, "Neural style transfer: A review," IEEE Trans. Vis. Comput. Graph., vol. 30, no. 3, pp. 1255–1275, Mar. 2022.
[14]. T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 12, pp. 4217–4228, Dec. 2021.
[15]. P. Wang, H. Li, and C. Chen, "Non-photorealistic image generation method based on convolutional neural networks," J. Comput.-Aided Des. Comput. Graph., vol. 32, no. 9, pp. 1789–1796, 2020.

[16]. X. Wang, Y. Liu, and C. Chen, "Research on image style transfer algorithm based on generative adversarial networks," J. Softw., vol. 31, no. 8, pp. 2456–2468, 2020.
[17]. H. Li, Q. Zhao, and W. Sun, "Real-time image stylization method based on convolutional neural networks," J. Comput. Appl. Res., vol. 39, no. 3, pp. 789–796, 2022.
[18]. M. Ruder, A. Dosovitskiy, and T. Brox, "Artistic style transfer for videos using temporal constraints," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2023, pp. 10234–10243.
[19]. E. Richardson, G. Metger, and D. Cohen-Or, "Neural 3D style transfer for volumetric shapes," ACM Trans. Graph., vol. 42, no. 6, pp. 1–14, 2023.
[20]. L. Karazija, A. Holynski, and R. Zhang, "CLIPstyler: Multimodal style transfer with vision-language models," Adv. Neural Inf. Process. Syst., vol. 36, pp. 12389–12402, 2023.
[21]. Horn R A and Johnson C R. Matrix Analysis. New York: Cambridge University Press, 2012.