Two-stage Gradient Vectorization via SAM Guidance and Differentiable Rendering

Anshu Hu, Guixiang Nie*, Qing Xie, Jiachen Li, Jinyu Xu
Wuhan University of Technology, Wuhan, Hubei, China
Engineering Research Center of Intelligent Service Technology for Digital Publishing, Ministry of Education
Emails: anshuhu@whut.edu.cn, nieguixiang@whut.edu.cn, felixxq@whut.edu.cn, lijiachen@whut.edu.cn, jinyxu@whut.edu.cn
Yanchun Ma
Wuhan Vocational College of Software and Engineering, Wuhan, Hubei, China
Hubei Engineering Research Center for Intelligent Detection and Identification of Complex Parts
Email: mayanchun@whvcse.edu.cn

***Abstract*—Image vectorization aims to convert raster images into editable and resolution-independent vector representations. While recent learning-based methods based on differentiable rendering achieve high automation, most rely on solid-color primitives and struggle to model smooth color gradients efficiently. Existing gradient-aware approaches improve gradient fitting but often suffer from limited structural robustness and high computational cost on complex images.**

In this paper, we propose TG-Vec, a two-stage gradient image vectorization framework guided by a large-scale pretrained segmentation model (SAM) and differentiable rendering. The first stage constructs a semantically coherent global vector structure using SAM-guided segmentation and radial gradient primitives. The second stage selectively complements missing details from residual regions, enabling efficient refinement without redundant optimization.

Extensive experiments on emoji and natural image datasets demonstrate that TG-Vec achieves improved visual fidelity and more structured vector representations compared with state-of-the-art methods, while reducing computational cost.

***Index Terms*—Image Vectorization, SAM, Differentiable Rendering**

I. Introduction

Vector graphics, commonly represented in the SVG [1] format, store images using lightweight code. Owing to their resolution independence and high editability, they are widely used in publishing and graphic design.

Early traditional approaches based on region extraction, diffusion curves, and gradient meshes [2]–[6] explicitly model color transitions using gradient primitives, enabling accurate representation of smooth shading and illumination variations in raster images. These methods demonstrate strong expressive power for gradient regions and can produce visually appealing vector results when carefully configured. However, such traditional techniques typically rely on hand-crafted heuristics, manually designed pipelines, or extensive parameter tuning, which makes them sensitive to image content and difficult to generalize across diverse scenes. As image complexity increases, their performance degrades noticeably, limiting scalability and practical applicability in large-scale or fully automatic scenarios [7], [8].

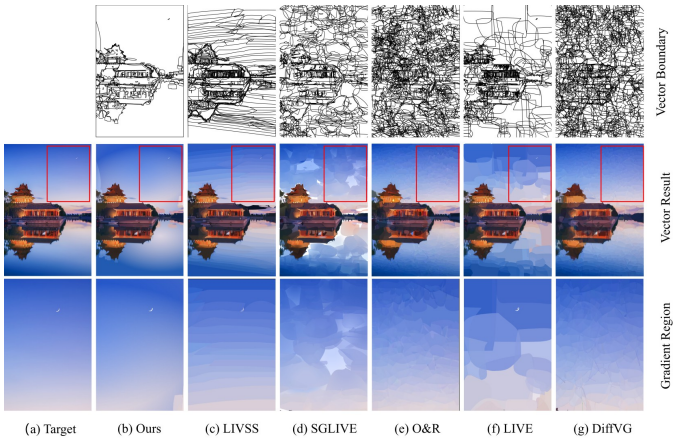

Fig. 1: Comparison of Various Vectorization Result in Natural Image. We vectorized a natural image with gradient regions using our method, LIVSS [15], SGLIVE [16], O&R [12], LIVE [11], and DiffVG [9].

Driven by advances in differentiable rendering [9] and deep learning, a growing body of learning-based image vectorization methods has emerged [10]. These approaches formulate vectorization as an optimization problem over vector primitive parameters, minimizing the discrepancy between the target raster image and the rasterized output of the current vector representation. Compared with traditional methods, learning-based techniques offer higher automation, better extensibility, and improved adaptability to complex image content [11]–[15]. Despite these advantages, most existing learning-based methods primarily rely on solid-color primitives, which severely restricts their ability to model regions with smooth color transitions and continuous gradients. As a result, they often require a large number of primitives to approximate gradient regions, leading to compromised reconstruction fidelity and reduced editability. To address this limitation, SGLIVE [16] extends learning-based vectorization frameworks by introducing radial gradient primitives, enabling automatic gradient-aware vectorization within a differentiable optimization paradigm. While effective on relatively simple

images, SGLIVE adopts coarse segmentation strategies and an inefficient layer-wise optimization scheme [11], which requires multiple sequential optimization stages and leads to high computational cost. These limitations hamper both structural robustness and optimization efficiency when applied to complex natural images. As illustrated in Figure 1, for real-world scenes, the gradient-based results produced by SGLIVE often fail to preserve structural integrity and visual fidelity, and can even be inferior to segmentation-driven solid-color vectorization approaches [15], [17].

To address the above challenges, we propose **TG-Vec**, a **T**wo-stage **G**radient **Vec**torization via SAM [17] Guidance and Differentiable Rendering [9]. Our main contributions are summarized as follows:

- We leverage a large-scale image segmentation model, Segment Anything (SAM), to guide the initialization of vector primitives with radial gradient colors, enabling structure-aware and gradient-aware vector representations for complex images.
- We propose an efficient two-stage vector initialization and optimization strategy, which refines vector representations from coarse structures to fine details, significantly reducing time cost compared with prior gradient-based methods.
- Extensive experiments demonstrate that the proposed method achieves high visual fidelity and strong editability on both solid-color and gradient images, across simple and complex scenes, while maintaining improved computational efficiency.

II. Related Work

A. Traditional Image Vectorization

Traditional image vectorization methods can be broadly categorized into solid-color and gradient-based approaches [7]. Solid-color methods typically follow a pipeline of color quantization [18], [19], region segmentation [20], contour extraction, and curve fitting. Representative systems such as Potrace [21] and Adobe Image Trace [22] approximate region boundaries using polygons or Bézier curves [23], [24]. While effective for flat-color images, these methods struggle to represent smooth color transitions commonly found in natural images and illustrations.

To overcome this limitation, gradient-based methods explicitly introduce continuous color models. Diffusion curves [4], [25] propagate colors defined along curve boundaries, while mesh-based approaches [2], [3] interpolate colors within triangular or irregular meshes. However, many structure-preserving gradient techniques [5], [6], [26], [27] rely on complex algorithms or user intervention, limiting their efficiency and scalability.

B. Learning-Based Image Vectorization

Learning-based image vectorization methods [10] formulate vectorization as an optimization problem under differentiable rendering frameworks such as DiffVG [9], where solid-color primitives are iteratively optimized to minimize the discrepancy between the target image and the rendered output. But without explicit structural guidance, these approaches often require densely stacked primitives to approximate complex appearances, resulting in cluttered vector structures and limited editability. Methods such as LIVE [11] and Optimize-and-Reduce (O&R) [12] improve reconstruction quality through increasingly sophisticated optimization strategies, at the cost of substantial computational overhead. To enhance structural quality, segmentation-guided methods [13]–[15], [28], [29] leverage image segmentation results [17] to guide vector initialization and optimization, producing clearer vector layouts and improved editability.

Recently, gradient-aware learning-based methods have been explored to better model smooth color transitions. SGLIVE [16] extends differentiable vectorization frameworks by introducing radial gradient primitives, enabling automatic gradient vectorization for simple images such as stickers and emojis. Its progressive segmentation-and-optimization strategy improves gradient fitting but incurs multiple optimization stages, leading to limited efficiency and reduced robustness when applied to complex natural images.

III. Method

A. Overview

We propose a two-stage gradient image vectorization framework guided by SAM and differentiable rendering, as illustrated in Fig. 2. The framework is designed to address three key challenges in gradient vectorization: structural robustness, visual fidelity, and efficiency. The input image is represented using radial gradient primitives and vectorized through two stages: Structure Construction and Detail Complement, both of which consist of vector initialization followed by differentiable optimization, forming a coarse-to-fine strategy. In the Structure Construction stage, SAM is employed to guide the initialization to capture global structure and consistent region layouts for reliable gradient configurations, while in the Detail Complement stage, the difference image between the

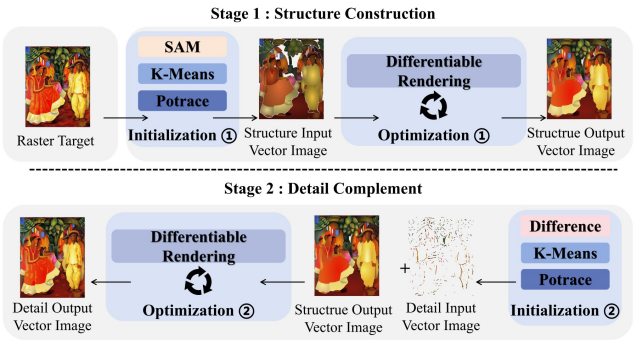

Fig. 2: Overview of the proposed two-stage gradient image vectorization framework. Each stage involves vector initialization and differentiable optimization. We use K-Means [30] to initialize vector colors and Potrace [21] to initialize vector shapes from different sources.

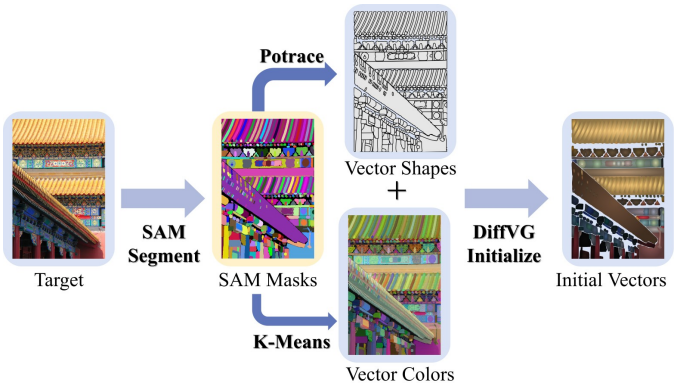

Fig. 3: Structure Construction Initialization: SAM masks guide vectorization, with contours from Potrace and colors initialized by K-means.

target raster image and the first-stage result is used to refine primitives only in residual regions, thereby supplementing fine details without compromising efficiency.

B. Initialization

Both stages involve vector initialization to provide suitable parameters for subsequent differentiable optimization. Each initialization includes vector shape and color construction. The two stages mainly differ in shape initialization, while color initialization follows the same procedure.

As illustrated in Fig. 3, the initialization of structure construction stage is guided by segmentation masks produced by the Segment Anything Model (SAM) [17]. We apply global segmentation to the target raster image I_{target} to obtain a set of binary masks M_{struct}, which can be formulated as:

$$M_{struct} = SAM(I_{target}), \tag{1}$$

where $M_{struct} = \{m^i_{struct}\}_1^{n_{struct}}$, n_{struct}, and $SAM(\cdot)$ represents the segmentation process. If the number of masks n_{struct} is smaller than the total number of primitives n, detail complement stage will perform.

The initialization of detail complement stage is shown in Fig. 4. Specifically, we extract connected components from

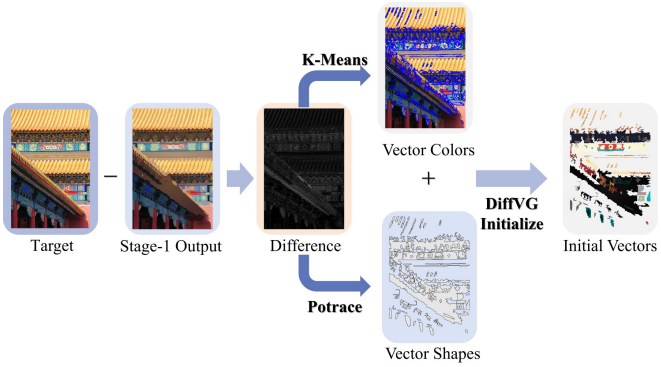

Fig. 4: Detail Complement Initialization: additional primitives are generated from the difference between the target image and the rasterized structure-stage result.

the difference image I_{diff} between the optimized result of the structure construction stage and the target image. This process can be formulated as

$$M_{diff} = Connect(I_{diff}), \tag{2}$$

where $M_{diff} = \{m^i_{diff}\}_{i=1}^{n_{diff}}$, n_{diff} denotes the number of extracted masks, and $Connect(\cdot)$ denotes connected-component extraction.

If the number of extracted primitives exceeds n, the masks are ranked according to their area and contour perimeter, and the top n_{detail} masks are selected to form the final detail mask set:

$$M_{detail} = Sort(M_{diff}, n_{detail}), \tag{3}$$

where $n_{detail} = n - n_{struct}$, $n_{detail} > n_{diff} > 0$, and $Sort(\cdot)$ returns the top n_{detail} masks after ranking.

To obtain the final vector shapes, we apply Potrace [21] to vectorize the above mask regions. For each stage $i \in struct, detail$, we obtain a set of vector shapes P_i, defined as

$$P_i = Potrace(M_i), \tag{4}$$

where $P_i = \{P^j_i\}_{j=1}^{n_i}$, and each P^j_i consists of a set of Bézier control points whose number is adaptively determined by Potrace.

After obtaining the vector shape parameters, we further perform color initialization on the corresponding image masks to obtain the color parameters. For the radial gradient primitive adopted in this work, the color parameters are represented by a set of variables, including the gradient colors c, the gradient radius r, the gradient center coordinates (c_x, c_y), and a vector of gradient stop positions. The gradient stop positions are fixed to [0,1], which means each primitive contains two gradient colors, start color c_0 and end color c_1.

For each stage $i \in struct, detail$, we initialize the color parameter sets C_i by applying K-means clustering (k=2) to the pixel colors of the target image within each mask region:

$$C_i = KMeans(M_i, I_{target}, k), \tag{5}$$

where $C_i = \{C^j_i\}_{j=1}^{n_i}, C^j_i = \{c^j_{i0}, c^j_{i1}\}$, and each color is represented as a four-dimensional vector $c = (R, G, B, \alpha)$. The gradient radius parameters R_i and the gradient center coordinates $(C_x, C_y)_i$ are initialized from the bounding box of each mask region:

$$R_i, (C_x, C_y)_i = Bound(M_i), \tag{6}$$

where $Bound(\cdot)$ denotes the operation that computes the center and radius of the bounding box enclosing the mask region.

C. Optimization

In the vector optimization stage, we adopt a deliberately simple optimization strategy: both the structure construction stage and the detail complement stage perform only a single round of vector parameter optimization. Compared with methods such as LIVSS [15] and O&R [12], which iteratively add

or remove vector primitives across multiple layers, our strategy significantly simplifies the optimization pipeline and improves computational efficiency.

During optimization, we collect all initial vector parameters into a parameter set V and optimize them using the Adam optimizer. The parameter set includes Bézier curve control points p, color parameter c, radial gradient radius r, and gradient center coordinates (c_x, c_y).

Vector optimization is performed in two stages. For the structure construction stage, we optimize the initialized vectors:

$$V_{out}^{struct} = Opt(V_{in}^{struct}). \tag{7}$$

Then, the optimized structural vectors are jointly optimized with the initialized detail vectors:

$$V_{out}^{detail} = Opt(V_{out}^{struct} \cup V_{in}^{detail}), \tag{8}$$

and V_{out}^{detail} is used to generate the final vectorized result. Both stages adopt a simple mean squared error loss:

$$L = MSE(I_{render}, I_{target}), \tag{9}$$

where I_{render} is obtained via differentiable rendering.

IV. Experiment

A. Experimental Setup

Datasets. We evaluate our method on three self-defined datasets covering both synthetic and real-world scenarios: **PureEmo** and **GradientEmo** are emoji datasets from Kaggle [31] with 500 images each, representing pure-color and smooth-gradient cases, respectively; **NaturalPic** contains 200 natural images with complex structures and illumination variations original from Kaggle and Baidu [32], [33]. Although the datasets are relatively small, they cover both synthetic and complex natural scenarios, providing a representative evaluation of gradient vectorization tasks.

Compared Methods. We compare our approach with representative image vectorization methods, including LIVSS [15], SGLIVE [16], O&R [12], LIVE [11], and DiffVG [9]. Among these methods, LIVSS leverages SAM for segmentation guidance, while SGLIVE is the only learning-based approach that explicitly employs gradient primitives. The remaining methods rely on solid-color primitives and differentiable rendering without external segmentation models.

Implementation Details. All methods use the same number of vector primitives and default parameter settings provided in their original implementations. We set the number of primitives to $n = 16$ for the PureEmo and GradientEmo datasets, and to $n = 512$ for the NaturalPic dataset to account for the increased structural complexity of natural images. All experiments are conducted on a workstation equipped with an Intel Xeon Silver 4214R (2.40 GHz) CPU and an NVIDIA RTX 3090 GPU, running Ubuntu 20.04.4.

Evaluation Metrics. We evaluate vectorization performance using the following metrics: (1) Mean Squared Error (MSE); (2) Peak Signal-to-Noise Ratio (PSNR); (3) Structural Similarity Index Measure (SSIM) [34]; (4) Learned Perceptual Image Patch Similarity (LPIPS) [35]. In addition, we report efficiency-related metrics, including the total vectorization time (Time), primitive optimization time (OT), and non-optimization processing time (PT).

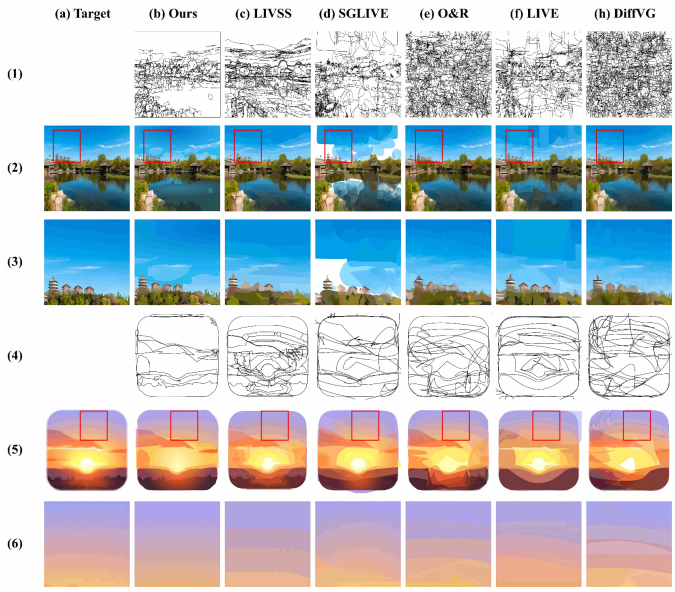

Fig. 5: Gallery of examples vectorized by six methods. Images in (1), (4) are vector boundaries of vectorization results. Images in (2), (5) are vectorization result, and (3), (6) their gradient regions.

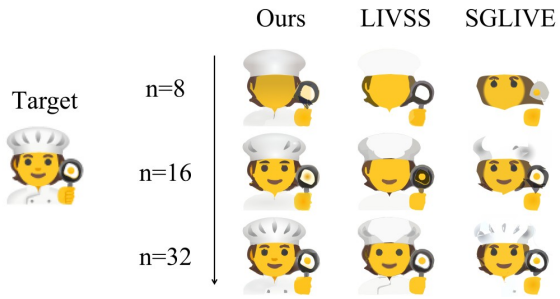

Fig. 6: Comparison of LIVSS, SGLIVE, and our method under different primitive numbers ($n = 8, 16, 32$) on emoji images. Our method effectively balances gradient modeling and structural preservation.

B. Results

Qualitative Evaluation. Fig. 5 presents qualitative comparisons on both emoji from GradientEmo and natural images from NaturalPic. Our method achieves structural quality comparable to LIVSS while recovering more fine details. Compared with SGLIVE and other solid-color methods, our approach more effectively models large gradient regions and captures more details, resulting in more accurate and compact vector structures.

We further compare our method with the state-of-the-art approaches SGLIVE and LIVSS under different primitive number n. For emoji images, as shown in Fig. 6, our method

Fig. 7: Comparison of LIVSS, SGLIVE, and our method under different primitive numbers ($n = 256, 512, 1024$) on natural images. We choose a natural image with light bulbs as input to do comparison. Our method maintains realistic structures while better capturing gradient effects such as reflections and shadows.

TABLE I: Quantitative comparison between the state-of-the-art methods and Ours. **Bold red** for the best result and **bold blue** for the second.

Datasets	Method	Time (s)↓	OT (s)↓	PT (s)↓	MSE↓	PSNR↑	SSIM↑	LPIPS↓
PureEmo	Ours	**13.60**	**5.832**	7.774	**0.0045**	**26.28**	**0.9123**	**0.0298**
	LIVSS	111.1	31.05	80.14	**0.0033**	**27.12**	**0.9494**	**0.0282**
	SGLIVE	89.39	89.37	**0.022**	0.0183	22.04	0.8317	0.0889
	OR	102.0	99.66	2.353	0.0071	24.25	0.9025	0.0603
	LIVE	69.48	69.44	0.047	0.0054	24.59	0.8869	0.0445
	DiffVG	**13.87**	**13.83**	**0.041**	0.0086	23.05	0.8674	0.0771
GradientEmo	Ours	**20.96**	**8.735**	12.22	**0.0039**	**25.59**	**0.8584**	**0.0420**
	LIVSS	106.8	32.30	74.53	0.0052	**25.33**	**0.8631**	0.0674
	SGLIVE	69.30	69.28	**0.0185**	0.0472	23.62	0.8053	0.0862
	OR	88.98	86.59	2.394	0.0054	23.44	0.8488	0.0754
	LIVE	61.59	61.55	**0.0397**	**0.0048**	24.80	0.8168	**0.0655**
	DiffVG	**13.70**	**13.66**	0.0409	0.0058	23.94	0.8159	0.0896
NaturalPic	Ours	**325.0**	**94.03**	231.0	**0.0085**	**21.75**	**0.5630**	**0.4706**
	LIVSS	1188.4	295.2	893.2	**0.0094**	**21.57**	**0.5128**	**0.5285**
	SGLIVE	1136.7	1132.2	**4.579**	0.0746	12.97	0.4293	0.6371
	OR	916.8	270.0	646.8	0.0120	20.22	0.4617	0.5754
	LIVE	1688.6	1681.6	6.997	0.0098	21.32	0.5031	0.5544
	DiffVG	**314.4**	**314.1**	**0.364**	0.0096	21.27	0.4890	0.5660

exhibits more robust performance across varying primitive counts. For complex natural images, as illustrated in Fig. 7, under different primitive settings, our approach not only better preserves the overall image structure but also provides more realistic representations of reflections, shadows, and other fine details.

Quantitative Comparison. The quantitative results of all methods are summarized in Table I. In terms of visual fidelity metrics, our method achieves the best performance on emoji images with gradient regions and on natural images, while slightly underperforming LIVSS on solid-color emojis. Regarding efficiency, compared with segmentation-based methods such as LIVSS, our approach significantly reduces non-optimization overhead and achieves the lowest optimization time. As a result, the overall runtime is second only to the simplest DiffVG baseline.

Ablation Study. To investigate the effects of the detail complement stage and gradient primitives, we conduct four ablation studies on both PureEmo and GradientEmo datasets, considering two factors: whether to use solid-color or gradient primitives, and whether to include the detail complement stage. As reported in Table II, both the detail complement stage and the use of radial gradient primitives consistently improve the vectorization performance.

TABLE II: Quantitative Results of Ablation Studies: **Bold red** for the best result.

Gradient Fill	Detail Complement	MSE↓	PSNR↑	SSIM↑	LPIPS↓
×	×	0.0188	21.51	0.7854	0.0998
×	√	0.0119	22.89	0.8166	0.0751
√	×	0.0115	23.01	0.8199	0.0782
√	√	**0.0042**	**25.94**	**0.8854**	**0.0359**

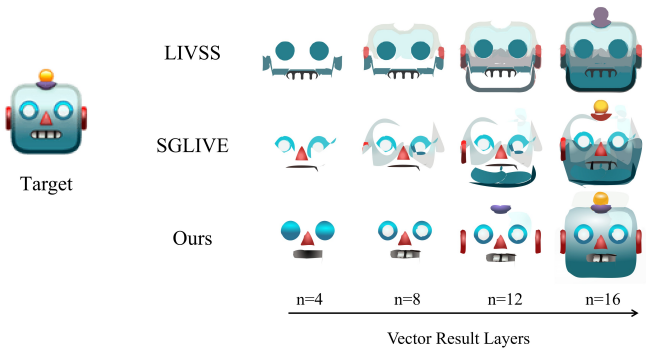

Fig. 8: Layer-wise visualization of emoji vectorization results produced by our method, LIVSS, and SGLIVE. The vector representations are progressively decomposed with the number of primitives increasing from $n = 4$ to $n = 16$.

Layer Representation. We further present a layer-wise visualization of the vectorization results produced by SGLIVE, LIVSS, and our method. As shown in Fig. 8, the progressive decomposition of vector layers indicates that our method gen-

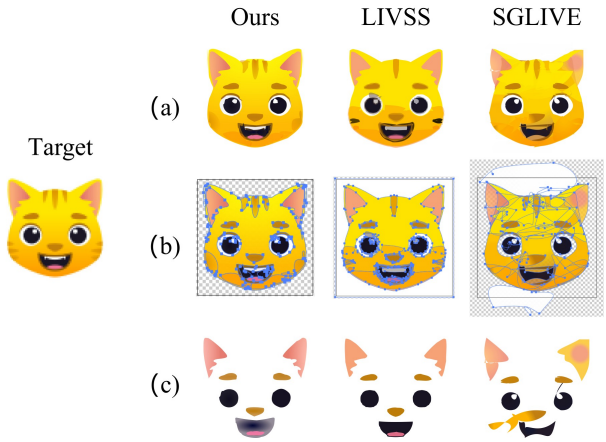

Fig. 9: Visualization of target selection for the vectorized results of our method, LIVSS, and SGLIVE. (a) Vectorization outputs. (b) Vector representations shown in a vector editing software, with blue points denoting Bézier control points. (c) Extracted facial components.

erates more natural and well-organized vector layers with more regular primitive shapes. We additionally open the vectorization results using a standard vector editing tool and perform object-level extraction. As illustrated in Fig. 9, compared with SGLIVE, our method produces vector results with clearer layer organization and more coherent shapes, while compared with LIVSS, it provides more accurate representations of gradient regions.

V. Conclusion

TG-Vec integrates segmentation guidance, radial gradients, and two-stage optimization for progressive refinement from global structure to fine details. Experiments show improved visual quality, compact representations, and efficiency. However, performance depends on segmentation quality and SAM, which may cause fragmented structures and limit the number of primitives. Future work will focus on improving segmentation robustness and primitive flexibility.

Acknowledgment

This research is partially supported by National Natural Science Foundation of China (Grant No. 62271360) and Natural Science Foundation of Hubei Province(Grant No.2025AFB950).

References

[1] J. Ferraiolo, F. Jun, and D. Jackson, *Scalable vector graphics (SVG) 1.0 specification*. iuniverse Bloomington, 2000.

[2] J. Sun, L. Liang, F. Wen, and H.-Y. Shum, "Image vectorization using optimized gradient meshes," *ACM Transactions on Graphics (TOG)*, vol. 26, no. 3, pp. 11–es, 2007.

[3] Y.-K. Lai, S.-M. Hu, and R. R. Martin, "Automatic and topology-preserving gradient mesh generation for image vectorization," *ACM Transactions on Graphics (TOG)*, vol. 28, no. 3, pp. 1–8, 2009.

[4] A. Orzan, A. Bousseau, H. Winnemöller, P. Barla, J. Thollot, and D. Salesin, "Diffusion curves: a vector representation for smooth-shaded images," *ACM Transactions on Graphics (TOG)*, vol. 27, no. 3, pp. 1–8, 2008.

[5] J.-D. Favreau, F. Lafarge, and A. Bousseau, "Photo2clipart: Image abstraction and vectorization using layered linear gradients," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 6, pp. 1–11, 2017.

[6] Z.-J. Du, L.-F. Kang, J. Tan, Y. Gingold, and K. Xu, "Image vectorization and editing via linear gradient layer decomposition," *ACM Transactions on Graphics (TOG)*, vol. 42, no. 4, pp. 1–13, 2023.

[7] X. Tian and T. Günther, "A survey of smooth vector graphics: Recent advances in representation, creation, rasterization, and image vectorization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 30, no. 3, pp. 1652–1671, 2022.

[8] A. Hu, Y. Sun, J. Li, Y. Ma, Q. Xie, and Y. Liu, "Tovect: Topology-optimized vectorization for intangible cultural heritage thangka element line art," in *Proceedings of the 7th ACM International Conference on Multimedia in Asia*, 2025, pp. 1–7.

[9] T.-M. Li, M. Lukáč, M. Gharbi, and J. Ragan-Kelley, "Differentiable vector graphics rasterization for editing and learning," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 6, pp. 1–15, 2020.

[10] M. Dziuba, I. Jarsky, V. Efimova, and A. Filchenkov, "Image vectorization: a review," *Journal of Mathematical Sciences*, vol. 285, no. 2, pp. 155–168, 2024.

[11] X. Ma, Y. Zhou, X. Xu, B. Sun, V. Filev, N. Orlov, Y. Fu, and H. Shi, "Towards layer-wise image vectorization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 314–16 323.

[12] O. Hirschorn, A. Jevnisek, and S. Avidan, "Optimize & reduce: a top-down approach for image vectorization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 3, 2024, pp. 2148–2156.

[13] H. Zhu, J. I. Chong, T. Hu, R. Yi, Y.-K. Lai, and P. L. Rosin, "Samvg: A multi-stage image vectorization model with the segment-anything model," in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 4350–4354.

[14] K. Zhao, L. Bao, Y. Li, X. Su, K. Zhang, and X. Qiao, "Less is more: Efficient image vectorization with adaptive parameterization," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 18 166–18 175.

[15] Z. Wang, J. Huang, Z. Sun, Y. Gong, D. Cohen-Or, and M. Lu, "Layered image vectorization via semantic simplification," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 7728–7738.

[16] H. Zhou, H. Zhang, and B. Wang, "Segmentation-guided layer-wise image vectorization with gradient fills," in *European Conference on Computer Vision*. Springer, 2024, pp. 165–180.

[17] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *et al.*, "Segment anything," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 4015–4026.

[18] P. Heckbert, "Color image quantization for frame buffer display," *ACM Siggraph Computer Graphics*, vol. 16, no. 3, pp. 297–307, 1982.

[19] M. T. Orchard, C. A. Bouman *et al.*, "Color quantization of images," *IEEE transactions on signal processing*, vol. 39, no. 12, pp. 2677–2690, 1991.

[20] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 11, pp. 2274–2282, 2012.

[21] P. Selinger, "Potrace: a polygon-based tracing algorithm," 2003.

[22] A. Illustrator, "Adobe illustrator," 2026, accessed: 2026.01.01. [Online]. Available: https://www.adobe.com

[23] E. A. Dominici, N. Schertler, J. Griffin, S. Hoshyari, L. Sigal, and A. Sheffer, "Polyfit: Perception-aligned vectorization of raster clip-art via intermediate polygonal fitting," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 77–1, 2020.

[24] J. R. Diebel, *Bayesian Image Vectorization: the probabilistic inversion of vector image rasterization*. Stanford University, 2008.

[25] G. Xie, X. Sun, X. Tong, and D. Nowrouzezahrai, "Hierarchical diffusion curves for accurate automatic image vectorization," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 6, pp. 1–11, 2014.

[26] Z. Liao, H. Hoppe, D. Forsyth, and Y. Yu, "A subdivision-based representation for vector image editing," *IEEE transactions on visualization and computer graphics*, vol. 18, no. 11, pp. 1858–1867, 2012.

[27] S. Chakraborty, V. Batra, A. Phogat, V. Jain, J. S. Ranawat, S. Dhingra, K. Wampler, M. Fisher, and M. Lukáč, "Image vectorization via gradient reconstruction," in *Computer Graphics Forum*. Wiley Online Library, 2025, p. e70055.

[28] T. Hu, R. Yi, B. Qian, J. Zhang, P. L. Rosin, and Y.-K. Lai, "Supersvg: Superpixel-based scalable vector graphics synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 24 892–24 901.

[29] Q. Xie, G. Nie, A. Hu, Y. Ma, J. Xu, and J. Li, "Enhancing image vectorization fidelity and editability through adaptive layered techniques," *The Visual Computer*, vol. 42, no. 4, p. 201, 2026.

[30] J. MacQueen, "Multivariate observations," in *Proceedings ofthe 5th Berkeley Symposium on Mathematical Statisticsand Probability*, vol. 1, 1967, pp. 281–297.

[31] Subinium, "Full emoji image dataset," 2021. [Online]. Available: https://www.kaggle.com/datasets/subinium/emojiimage-dataset

[32] ikarus777, "Best artworks of all time," 2019. [Online]. Available: https://www.kaggle.com/datasets/ikarus777/best-artworks-of-all-time

[33] Baidu AI Studio, "Building and scene photo dataset," 2025. [Online]. Available: https://aistudio.baidu.com/datasetdetail/163876

[34] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[35] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 586–595.