Data-Driven Sketch Beautification with Neural Feature Representation

I-Chao Shen
The University of Tokyo

***Abstract*—This paper presents a data-driven approach for beautifying freehand sketches. Our key premise is that the artist-drawn vector can be used to sketch visually appealing shapes, such as local shapes with a clean appearance and better global visual properties (*e.g.*, symmetry). However, these merits may not apply to all object categories. In this paper, we use a neural network to represent local and global merits across different object categories, to design our beautification method. First, we match sample points between input sketches and the collected vector shapes using the extracted feature representations. Then, we then design an optimization problem to ensure resemblance between the deformed sketch and vector shape in the representation space, while preserving the semantic meaning and style of the original sketch. Finally, we demonstrate our method on sketches across different shape categories.**

Introduction

Sketching has been a major content creation and communication technique since prehistoric times. Even today, sketching may be the only rendering technique readily available to all humans. Several previous works have investigated how nonexperts sketch everyday objects [1], [2], by analyzing freehand sketch databases across different object categories. These works intend to identify *shared* and *iconic* representations of objects. Inspired by the availability of large freehand sketch datasets, Ha *et al.* [3] proposed *sketch-rnn*, an automatic sketching method based on a recurrent neural network (RNN). However, most of the sketches drawn by human and automatic methods [3] are not visually appealing.

Vector drawing is another popular approach to render abstract shapes. Professional artists usually create these vector shapes with dedicated tools such as Adobe Illustrator [1] or Inkscape[2]. Vector drawing has the following visual merits: (i) a clean and sharp visual appearance owing to the low dimensionality of the use of parametric curves, and (ii) better global relationships between different parts of the shape, such as symmetric and visual continuity [4]. Given the wide availability of vector drawings on the inter-

[1] https://www.adobe.com/products/illustrator.html
[2] https://inkscape.org/en/

 Published by the IEEE Computer Society

net, we aimed to determine if the visual merits could be transferred to freehand sketches? To answer this question, first, we need to identify correspondences between the shape of sketches and vector drawings, to transfer their properties between them. However, accurately establishing correspondences between two abstraction domains is usally difficult. Second, neither the local nor global merits of vector shapes have been confirmed to be applicable to different shape categories.

To address these issues, we propose a novel data-driven method to automatically beautify the freehand sketches. We utilize a point-based neural network [5] to extract features, as local and global characterizations of sample points in 2D space. To transfer the clean appearance and superior inter-part relationships of vector shapes, we have gathered vector shapes in several categories, such as cat, chair, and airplane from an online repository[3]. We ignore the colors of these vector shapes and focus only on the shape outlines. We extract the features of sample points from the collected vector shapes and store them in a reference database.

Our method considers a freehand sketch as a set of polylines and beautifies it via a series of steps. We extract the features of the sample points in the freehand sketch and establish their correspondences in the reference database. With the established correspondences thus established, we formulate an optimization problem to deform the sketch, with two separate goals: appearance transfer and shape preservation. We express the appearance transfer as the similarity between features and preserve the local shape of the sketch to maintain its original semantic meaning and drawing style.

Related works

Sketching is one of the most fundamental ways of expressing thoughts. To find the *shared* and *iconic* representations of objects, previous studies collected numerous sketches into databases of everyday objects [1], [2]. The Sketchy database [2] was compiled by asking crowd workers to sketch particular photographic objects sampled in 125 categories. Paired photograph and sketch databases serves as a benchmark for sketch image retrieval [2] and sketch-based image synthesis [6]. Given their simplicity, sketches are widely adopted in various computer graphics applications, *e.g.*, 3D shape modeling [7] and image synthesis [8].

Unlike other methods that use sketches as a proxy for other content domains, Ha and Eck [3] proposed *sketch-rnn*, which focuses on generating sketches automatically. They trained the RNN-based Seq2Seq Variational Autoencoder model on the *QuickDraw*[4] dataset. The trained model could generate unlimited sketches by assigning them to desired categories or providing sketch examples. However, the generated sketches were usually not visually appealing, such that there was a need for sketch beautification.

Many works have explored methods for improving freehand sketches and writing. These methods can roughly be categorized into rule-based and example-based methods. Regarding the rule-based method, the works of [9], [10] explore automatic techniques for beautifying geometric drawings by enforcing various relations, and provide several options for beautification. For the example-based method, freehand drawings and sketches can be beautified by converging strokes to the average form of similar strokes [11], or clustered and reordered strokes [12]. Zitnick [11] beautified simple freehand sketch drawings by converging strokes towards an average of similar strokes. Limpaecher *et al.* [13] built an artistic consensus model based on crowdsourced data and used it to interactively correct sketch drawings. Lu *et al.* [14] transferred styles from different strokes to auxiliary stylus trajectory data. Moreover, Baran *et al.* [15] smoothed the sketches using various curves, such as clothoid curves, while maintaining the user's intended meaning.

Our method is inspired by research on neural network interpretability and understanding [16], [17]. Mahendran *et al.* [16] aimed to reconstruct an image using feature maps in a convolutional neural network. They optimized a function by comparing the image representation with a natural image prior. Unlike previous works, our method uses the feature representation of vector shapes as a reference, and employs a similar optimization function to improve the input sketch appearance.

[3]https://www.flaticon.com

[4]http:\https://quickdraw.withgoogle.com

Method

Figure 1 illustrates the overview of our method. We first focus on a specific type of object, *e.g.*, a cat, dog, or airplane. The input sketch $\mathbf{S}$ is represented as a set of polylines, *i.e.*, $\mathbf{S} = \{L_0, L_1, ..., L_n\}$. Meanwhile, we collect a set of vector graphical shapes $\mathbf{VG} = \{\mathbf{G}_0, \mathbf{G}_1, ..., \mathbf{G}_m\}$ that belong to the same object category. The shape in $\mathbf{VG}$ is composed of continuous curves and formatted as SVG 1.1. We ignore all of the properties of the shape, such as color and curve thickness. We preprocess each of the shapes $\mathbf{G} \in \mathbf{VG}$ by subsampling the curves and convert them also into a set of unordered 2D points, *i.e.*, $\mathbf{G} = \{q_0, q_1, ..., q_n\}$.

We formulate the problem for beautifying the input sketch $\mathbf{S}$ as a sketch deformation optimization in the representation space (Figure 1). Our method leverages the neural network architecture PointNet [5] to obtain the neural feature representations of each sample point. We designed our method as a two-step approach. First, we match each sample point in $\mathbf{S}$ with the sample point in $\mathbf{VG}$ using the extracted feature representations. Then, we design an optimization problem to deform the original sketch, so that it resembles the neural representation in $\mathbf{VG}$.

Point feature and correspondence

In this section, we discuss the neural feature representation used for each sample point, and how to use them to match sample points. With the recent rapid developments in 3D point-cloud classification and segmentation [5], deep learning based architectures have proved to be effective for describing both the local and global properties of unstructured data. Among the architectures, we choose PointNet [5] because of its simplicity and powerful feature representation, which can encode both local and global structures.

PointNet The PointNet architecture for feature representation extraction is illustrated in Figure 2. PointNet is designed to consider unordered point sets as input data, and performs object classification and semantic segmentation tasks. Its key elements are as follows : a max-pooling layer, used as a symmetric function to aggregate information from all input points; a local and global information-based structure; and two alignment networks that align both input points and point representations. In our work, we leverage the benefit of local and global combination structures to extract a meaningful sample point feature. To obtain a useful feature representation that captures both the local and global properties of a sketch, we train the network to perform a sketch classification task. We use all of the sketches in the Sketchy dataset [2], across 125 categories, as training data. After training, we discard the final prediction layer in the network and use the network as a feature representation extractor (Figure 2).

We extract the features for both the sketch $\mathbf{S}$ and the vector shapes $\mathbf{G} \in \mathbf{VG}$. We denote the feature representation extractor as $\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^d$ ($d = 1088$ as shown in Figure 2). The extract representation of $\mathbf{S}$ is $\Phi(\mathbf{S}) = \{\Phi(p) : p \in \mathbf{S} = \{L_0, L_1, ..., L_n\}\}$, where p is the sample point on each of the polylines in sketch $\mathbf{S}$. Each vector shape $\mathbf{G} \in \mathbf{VG}$ is similarly defined as $\Phi(\mathbf{G}) = \{\Phi(q) : q \in \mathbf{G}\}$. We add the features of the entire vector shape in this category to one database $\mathcal{D}_{\mathbf{VG}} = \{\Phi(\mathbf{G}_0), \Phi(\mathbf{G}_1), ..., \Phi(\mathbf{G}_n)\}$.

Correspondence optimization To match correspondences for a sample point on $\mathbf{S}$, we first find the nearest neighbor of $\Phi(p) : p \in (\mathbf{S})$ in $\mathcal{D}_{\mathbf{VG}}$ (the corresponding point might belong to a different vector shape). However, if we use this nearest neighbor as the corresponding point, inconsistencies may arise among adjacent sample points, causing unsatisfactory visual effects. Thus, we design a correspondence optimization problem to further regularize the correspondence field further. We compute the assignment function f that assigns labels to each sample point p, where $p \in S$, such that the labeling f minimizes the following energy $E(f)$:

$$E(f) = w_{data} * \sum_{p \in S} D(p, f_p) \tag{1}$$

$$+ w_{smoothness} * \sum_{p,q \in \mathcal{N}} S(p, q, f_p, f_q), \tag{2}$$

where f_p and f_q are labels assigned to sample points p and q, respectively, and $\mathcal{N}$ is the entire set of neighboring sample points. We optimize this function by using the multilabel graph-cut algorithm proposed by Boykov [18]. We use the

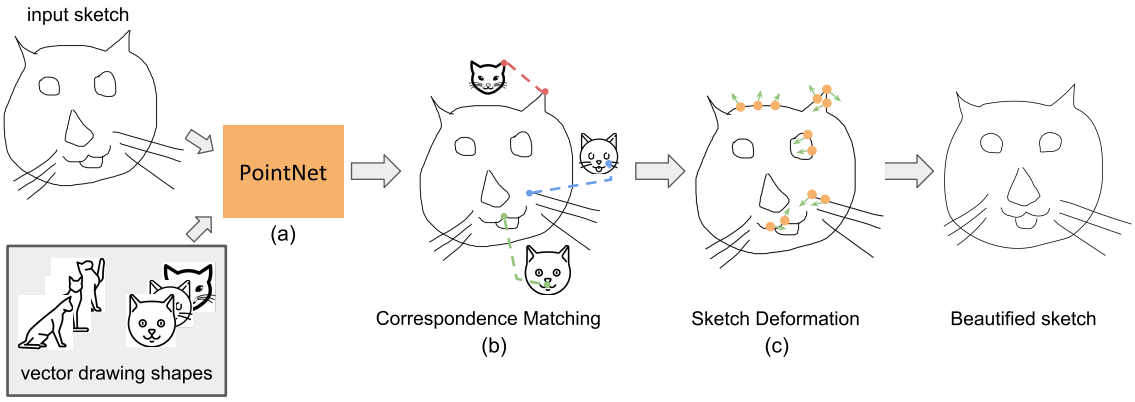

Figure 1: Overview of our method on the example of beautifying the cat sketch. For preprocessing, we collected a set of vector drawings (shapes) and extract their features using a PointNet as the reference database. We (a) extract the features of the sample points of an input sketch. Then, we (b) matched the sample points of the sketch to those in the vector shape reference database and obtained correspondence. Finally, we (c) performed a deformation optimization with the established correspondences, to transfer the visual merits from vector shapes to the input sketch. The orange dots are the sample points of the input sketch, and the green arrows indicate the deformation direction (we only show some of the sample points).

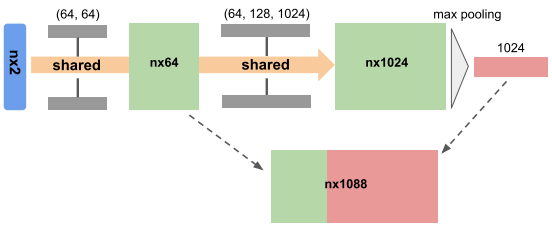

Figure 2: PointNet architecture for extracting point features. The global feature representation is obtained by max pooling. For each data point, the final feature representation is obtained by concatenating global and local features ($d = 1088$).

entire collection of all nearest sample points as the complete possible label set L. The $E(f)$ comprises two terms: data cost and smoothness.

Data cost. We measure the distance between a sample point p and its nearest neighbor in the feature space, as the data cost.

$$D(p, f_p) = \log(d(p, \Phi(p))), \tag{3}$$

Smoothness cost. This term measures the spatial consistency of neighboring elements.

$$S(p, q, l_p, l_q) = \begin{cases} 0, & \text{if } l_p = l_q, \\ d(\Phi(p), \Phi(q))^{-1}, & \text{otherwise} \end{cases} \tag{4}$$

where we define the smoothness term by the inverse Euclidean distance between p and q in the feature space. With the smoothness term, two adjacent sample points are likely to have corresponding consistent labels. We use $w_{\text{data}} = 1$ and $w_{\text{smoothness}} = 5$ for all of the results shown in this paper.

Sketch deformation

Using the matched points, we aim to deform the sketch $\mathbf{S}$ into $\mathbf{S}^*$ with the following two objectives: *appearance transfer* : transferring the appealing appearance of the vector shape to the sketch, and *maintain shape style*: preserve the original shape of $\mathbf{S}$. Moreover, we formulate the sketch deformation optimization problem as follows:

$$E = E_f + \lambda \cdot E_s, \tag{5}$$

where E_f measures the difference between the feature representations of the sketch sample points and their correspondence points, and E_s measures the local shape deviation of the deformed sketch relative to the original sketch. We use $\lambda = 10.0$ to obtain the results discussed in this paper. Our optimization focuses only on updating the sample point position in a sketch;

we do not introduce new points in the deformed sketch $\mathbf{S}^*$.

Appearance transfer The appearance transfer goal is formulated as the problem of finding the sketch $\mathbf{S}^*$, where the representations of all sample points ($p^* \in \mathbf{S}^*$) best match the corresponding points q in $\mathcal{D}_{\mathbf{VG}}$. Given a correspondence pair $\mathcal{K} = (p, q)$ and the representation function Φ, we want to obtain a new p^* such that the distance between $\Phi(p^*)$ and $\Phi(q)$ is minimized (*i.e.*, $\Phi(p^*) \approx \Phi(q)$). The appearance transfer term is then defined as:

$$E_f = \sum_{(p^*,q)\in\mathcal{K}} \|\Phi(p^* - \Phi(q)\|_2^2, \qquad (6)$$

where we compare the Euclidean distance between the point representation $\Phi(p)$ and the corresponding representation $\Phi(q)$.

Shape maintenance The deformation toward the representation of the above-described correspondence leads to variable results and is likely to change the entire sketch. However, our goal in sketch beautification is to improve the current sketch, not to create a new one. To maintain the original shape, we encourage the sample point p_i^* on the deformed sketch $\mathbf{S}^*$ maintains its shape relative to its neighbors. We use the Laplacian coordinate [19] to describe the local shapes of the sketch. For each sample point p_i, we compute the Laplacian coordinate as:

$$\delta(p_i) = \sum_{p_i\in\mathbf{S}} \frac{1}{2}(p_{i-1} - p_i) + \frac{1}{2}(p_{i+1} - p_i), \qquad (7)$$

We formulate this problem usingthe following term:

$$E_s = (\delta(p_i^*) \cdot \delta(p_i))^2 \qquad (8)$$

which essentially measures the deviation of the 1D Laplacian coordinate [19] before and after $\mathbf{S}$ is deformed.

Because the feature extractor Φ is differentiable, we can iteratively optimize Eq. 5 iteratively using the gradient descent method.

Result

We tested our method on various sketch categories, including cat, dog, and airplane, collected from the Sketchy dataset [2]. We collected 20-50 vector graphics shapes for each category, from websites such as https://www.flaticon.com. To validate our method, we compared our results with those of some existing alternative algorithms, such as pure Laplacian smoothing [19] and Cornucopia [15]. We generated results for both methods by separately processing each polyline in the input sketch.

Variant of the proposed method We compared our results to those obtained using a variant of our method. Zitnick [11] proposed that the appearance of the average of multiple instances of the same written word or shape is usually better than that of individual instances. Inspired by this proposition, we designed a variant of our method (Our_{avg}). First, we gathered all of the sketches within a given category and built a database of their features. For each sample point p on the sketch to be beautified, we identified the most k similar points $N(p)$ in the database by computing the cosine similarity between the feature representations. We then used the average of these k feature representations as the "transferring source". More specifically, we changed the target feature representation $\Phi(q)$ in Eq. (6) into :

$$\Phi(q') = \frac{1}{k} \sum_{\tilde{p}\in N(p)} \Phi(\tilde{p}) \qquad (9)$$

We show our example side-by-side with the comparison results for cat (Figure 4), airplane (Figure 5), and chair (Figure 6). In the figures, we highlight the region where our method beautifies the input sketch most effecitvely, in terms of cleaner and sharper curves and better global relationships. Across these categories, our method and the variant method (Our_{avg}) both improved local and global visual properties. As Zitnick [11] suggested, the average representation of a sketch can serve as a good reference to beautify it. However, we designed a more straightforward optimization problem based on informative neural feature representations. By introducing visual appealing vector shapes, our method could further beautify the input sketch, especially in terms of global merits such as symmetry.

Crowdsourced evaluation

We performed a crowd-sourced comparison study to evaluate the visual quality of the sketches produced by our method. We showed the crowd workers an original sketch for each query, the result obtained using our method, and the result obtained using another method. We then asked them to indicate the result that most effectively beautified the original sketch. We designed a survey composed of 16 queries. We used the Amazon Mechanical Turk interface to conduct the survey: each participant was shown the 16 queries (twice each and the order of presentation was varied). We excluded inconsistent answers by crowd workers from the analysis, for example when a duplicated query was answered differently or when over 25% of participants' answers were inconsistently. We report the result of 50 crowd workers who passed the consistency check.

Specifically, we showed the participants the original sketch, which was marked "A" and indicated the sketch category. The two beautified sketches were presented below the original and marked "B" and "C". We then asked: "Which of the two sketches at the bottom, "B" or "C", beautifies the sketch "A" the most? The answer options were "B" and "C". For each category, we randomized the assignment of the results of our method and the other method to "B" and "C". The results are shown in Figure 3.

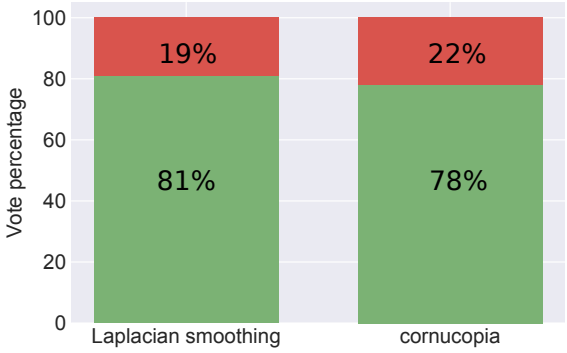

Figure 3: The crowd workers' preferences for our method (green) and the other method (red) across all queries.

However, there were also some limitations: if there are no similar vector shapes in sketches drawn from similar viewpoints, the perspective of the resulted sketches will not be correct (see the result for chair2 obtained using our method in the second row of Figure 6). Because the variant method (Our_{avg}) considers sketches drawn from similar viewpoints, it achieved better results for this case.

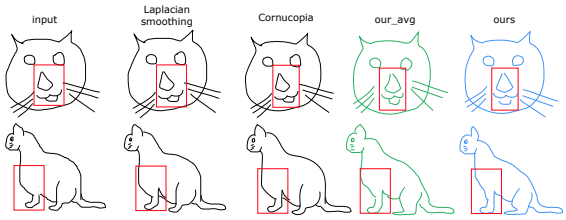

Figure 4: Beautification of *cat* sketches. Comparison of our results with those of Laplacian smoothing [19], smooth polyline fitting [15], and the variant of our own method (Our_{avg}).

Conclusions and Future Work

Our method can beautify sketches in a neural feature space. The feature representation captures both the local and global merits of vector drawing shapes. Our novel optimization enables deformations of the sketch toward visually appealing vector drawings. However, the representation function is highly nonlinear owing to the complex neural network layer stacks, complicating the optimization of our objective function. Moreover, because our method considers the global relationship between different parts of the sketch, the entire sketch is needed to extract a meaningful features. Thus, a featurerepresentation function that is more robust in describing unfinished shapes is needed. Finally, with the rapid development of deep learning methods for 3D point-cloud processing, our method is compatible with the latest feature extractor, *e.g.*, PointNet++ [5] and DGCNN [20].

ACKNOWLEDGMENT

I thank the anonymous reviewers who helped me improve the paper. This study was supported by the MediaTek Fellowship and JSPS KAKENHI Grant Number JP21F20075.https://www.overleaf.com/project/60ab3a86fa398b

REFERENCES

1. M. Eitz, J. Hays, and M. Alexa, "How do humans sketch objects?" *ACM Trans. Graph. (Proc. SIGGRAPH)*, vol. 31, no. 4, pp. 44:1–44:10, 2012.

Result name	Point count	Matching	Deformation
cat1 (Figure 4 top)	956	0.162 s	7.32 s
cat2 (Figure 4 bottom)	572	0.1 s	4.98 s
airplane1 (Figure 5 top)	1007	0.168 s	9.12 s
airplane2 (Figure 5 bottom)	797	0.139 s	6.82 s
chair1 (Figure 6 top)	792	0.134 s	7.31 s
chair2 (Figure 6 bottom)	597	0.105 s	5.2 s
bell ((Figure 7)	488	0.08 s	4.3 s
dolphin (Figure 7)	501	0.09 s	4.65 s
violin ((Figure 7)	416	0.07 s	4.13 s

Table 1: The number of sample points and running times for the matching and deformation steps for various sketches.

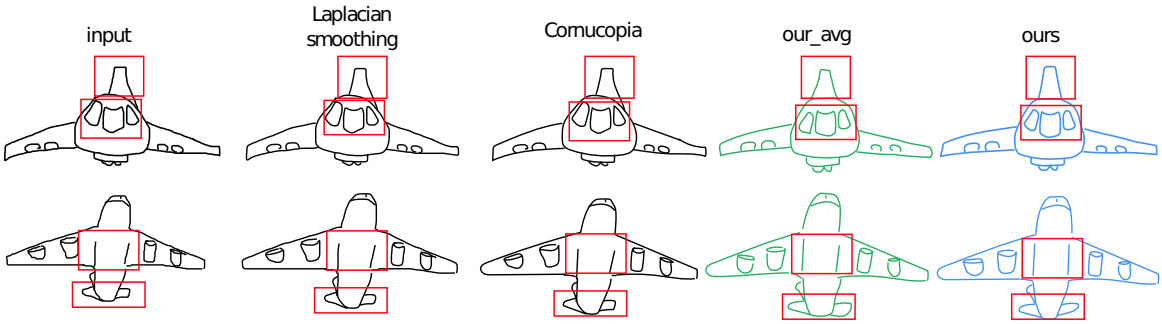

Figure 5: Beautification of *airplane* sketches. Comparison of our results with those of Laplacian smoothing [19], smooth polyline fitting [15], and the variant of our own method (Our_{avg}).

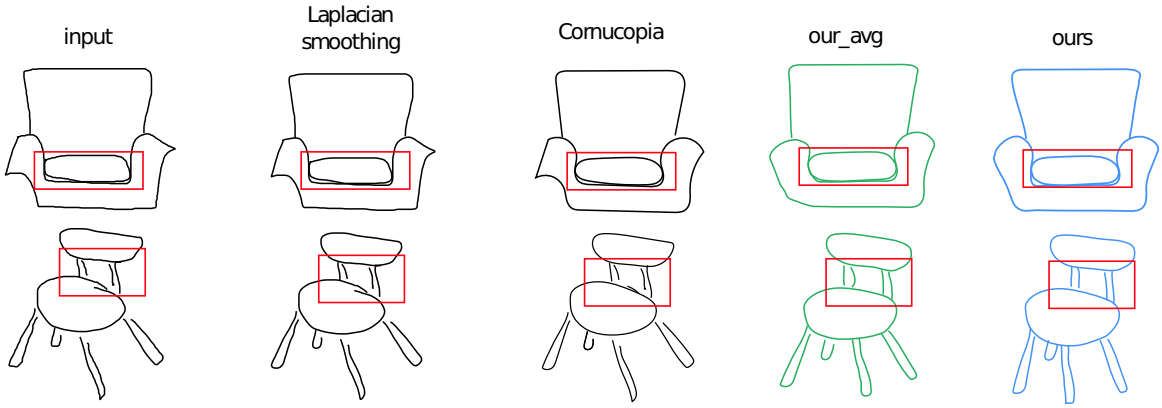

Figure 6: Beautification of *chair* sketches. Comparison of our results with those of Laplacian smoothing [19], smooth polyline fitting [15], and the variant of our own method (Our_{avg}).

2. P. Sangkloy, N. Burnell, C. Ham, and J. Hays, "The sketchy database: Learning to retrieve badly drawn bunnies," *ACM Transactions on Graphics (proceedings of SIGGRAPH)*, 2016.

3. D. Ha and D. Eck, "A neural representation of sketch drawings," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=Hy6GHpkCW

4. J. Wagemans, J. H. Elder, M. Kubovy, S. E. Palmer, M. A. Peterson, M. Singh, and R. von der Heydt, "A century of gestalt psychology in visual perception i. perceptual grouping and figure-ground organization," *Psychological Bulletin*, vol. 138, no. 6, pp. 1172–1217, 2012.

5. C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.

6. W. Chen and J. Hays, "Sketchygan: Towards diverse and realistic sketch to image synthesis," in *Proceedings of the IEEE Conference on Computer Vision and Pat-*

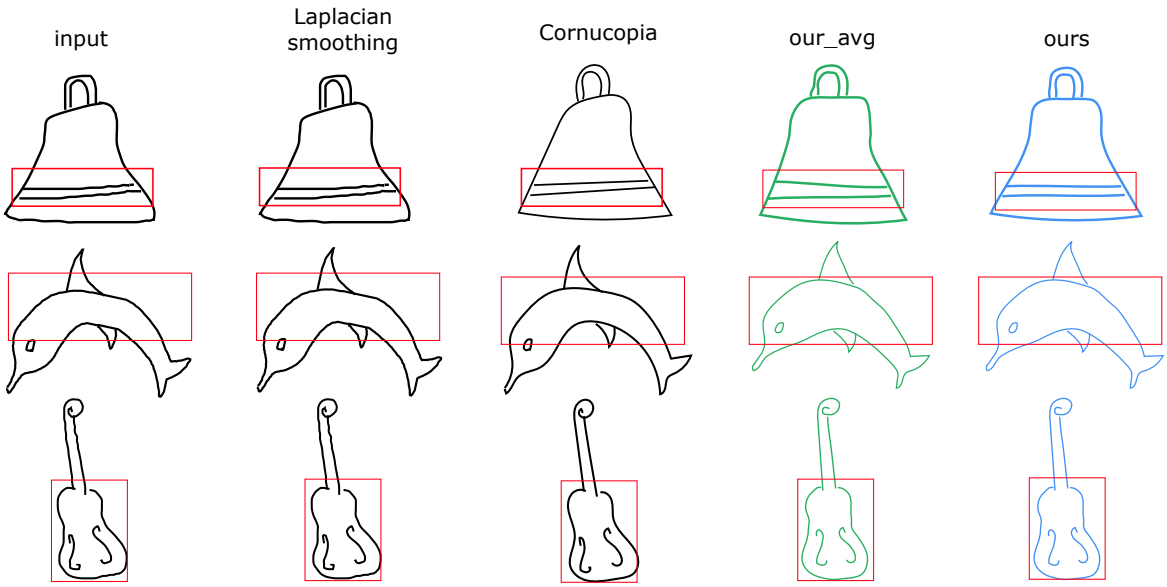

Figure 7: Beautification of *bell*, *dolphin*, and *violin* sketches. Comparison of our results with those of Laplacian smoothing [19], smooth polyline fitting [15], and the variant of our own method (Our_{avg}).

tern Recognition, 2018, pp. 9416–9425.

7. T. Igarashi, S. Matsuoka, and H. Tanaka, "Teddy: a sketching interface for 3d freeform design," in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 1999, pp. 409–416.
8. P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, 2017.
9. T. Igarashi, S. Matsuoka, S. Kawachiya, and H. Tanaka, "Interactive beautification: A technique for rapid geometric design," in *Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '97. New York, NY, USA: ACM, 1997, pp. 105–114. [Online]. Available: http://doi.acm.org/10.1145/263407.263525
10. J. Fišer, P. Asente, S. Schiller, and D. Sỳkora, "Advanced drawing beautification with shipshape," *Computers & Graphics*, vol. 56, pp. 46–58, 2016.
11. C. L. Zitnick, "Handwriting beautification using token means," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 53:1–53:8, Jul. 2013. [Online]. Available: http://doi.acm.org/10.1145/2461912.2461985
12. G. Orbay and L. B. Kara, "Beautification of design sketches using trainable stroke clustering and curve fitting," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 5, pp. 694–708, 2011.
13. A. Limpaecher, N. Feltman, A. Treuille, and M. Cohen, "Real-time drawing assistance through crowdsourcing," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 54:1–54:8, Jul. 2013. [Online]. Available: http://doi.acm.org/10.1145/2461912.2462016
14. J. Lu, F. Yu, A. Finkelstein, and S. DiVerdi, "HelpingHand: Example-based stroke stylization," in *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 31, no. 4, Aug. 2012, pp. 46:1–46:10.
15. I. Baran, J. Lehtinen, and J. Popović, "Sketching clothoid splines using shortest paths," in *Computer Graphics Forum*, vol. 29, no. 2. Wiley Online Library, 2010, pp. 655–664.
16. A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 5188–5196.
17. A. Dosovitskiy and T. Brox, "Inverting visual representations with convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4829–4837.
18. Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, pp. 1124–1137, September 2004. [Online]. Available: http://www.csd.uwo.ca/~yuri/Abstracts/pami04-abs.shtml
19. O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel, "Laplacian surface editing," in *Proceedings of the EUROGRAPHICS/ACM SIGGRAPH*

Symposium on Geometry Processing. ACM Press, 2004, pp. 179–188.

20. Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *Acm Transactions On Graphics (tog)*, vol. 38, no. 5, pp. 1–12, 2019.

I-Chao Shen is a JSPS postdoctoral researcher at the Graduate School of Information Science and Technology at the University of Tokyo, working with Takeo Igarashi. He did his Ph.D. in the computer graphics group at National Taiwan University, advised by Robin Bing-Yu Chen. He received B.B.A and M.B.A degrees in information management from National Taiwan University, in 2009 and 2011, respectively. He was a research assistant in the Imager lab of The University of British Columbia, a research internship in the Imagination lab of Adobe Corporation, and visiting researcher at The University of Tokyo. Please contact him at ichao.shen@ui.is.s.u-tokyo.ac.jp.