DOI: 10.1111/cgf.14485
EUROGRAPHICS 2022 / R. Chaine and M. H. Kim
(Guest Editors)
Volume 41 (2022), Number 2

Vectorizing Line Drawings of Arbitrary Thickness via Boundary-based Topology Reconstruction

Zibo Zhang[1], Xueting Liu[†2], Chengze Li[2], Huisi Wu[†1] and Zhenkun Wen[1]

[1]Shenzhen University, China
[2]Caritas Institute of Higher Education, Hong Kong SAR, China

Abstract

Vectorization is a commonly used technique for converting raster images to vector format and has long been a research focus in computer graphics and vision. While a number of attempts have been made to extract the topology of line drawings and further convert them to vector representations, the existing methods commonly focused on resolving junctions composed of thin lines. They usually fail for line drawings composed of thick lines, especially at junctions. In this paper, we propose an automatic line drawing vectorization method that can reconstruct the topology of line drawings of arbitrary thickness. Our key observation is that no matter the lines are thin or thick, the boundaries of the lines always provide reliable hints for reconstructing the topology. For example, the boundaries of two continuous line segments at a junction are usually smoothly connected. By analyzing the continuity of boundaries, we can better analyze the topology at junctions. In particular, we first extract the skeleton of the input line drawing via thinning. Then we analyze the reliability of the skeleton points based on boundaries. Reliable skeleton points are preserved while unreliable skeleton points are reconstructed based on boundaries again. Finally, the skeleton after reconstruction is vectorized as the output. We apply our method on line drawings of various contents and styles. Satisfying results are obtained. Our method significantly outperforms existing methods for line drawings composed of thick lines.

CCS Concepts

• ***Applied computing*** → ***Fine arts;***

1. Introduction

Vectorization is a commonly used technique for converting raster images to vector format so that the image can be rescaled without any quality loss. Comparing to raster images, vector graphics are of relatively small file size which is suitable to be used in various applications, such as web-compatible images and illustrations. Besides, with the vector representation, one can easily edit the line drawing, such as changing the color and thickness of each individual line. The vector representation also benefits various applications of line drawings, such as retargeting and stereoscopization. While vector graphics can be directly created with vector graphics software (e.g. Adobe Illustrator, CorelDRAW, etc.), drawing in raster format (either digitally or on paper) is still more natural to humans. Moreover, even if a drawing is created in vector format, it may be rasterized when distributed, since most of the current social communication software only supports the exchange of raster images. Therefore, how to vectorize raster images has long been a research interest and attracts research attempts from many researchers.

During vectorization, one key challenge is in vectorizing lines with arbitrary thickness and shapes. Various methods have been proposed to vectorize line drawings based on different topology extraction methods. A simple but commonly adopted method is to first thin the lines to one-pixel thickness [ZS84, LLS92] and then connect the pixels based on local connectivity. While this method works for arbitrary-thickness line drawings, the reconstructed topology usually fails at junctions and corners (Figure 1(b)). One idea for improvement is to take advantage of global information [FLB16], but it may lead to oversimplified results (Figure 1(c)). Traditional vectorization methods estimate the topology of the lines by fitting the lines to some pre-defined shapes [CY98, DCP17, JV97, HT06], but these methods generally fail to resolve lines of arbitrary thickness and shapes, especially at junctions, due to the overlapping of lines. To reconstruct the topology at junctions, methods have been proposed to resolve the ambiguity of junctions based on tangential fields [NHS*13, NS19, BS19]. However, these methods are usually tailored for resolving junctions composed of thin lines. They usually fail to reconstruct the topology of line drawings composed of thick lines (Figure 1(d)). Recently, several learning-based methods have been proposed to reconstruct the topology of line drawings [KWÖG18, SII18, GZH*19], but these methods are unable to solve line drawings of arbitrary thickness

† Corresponding author

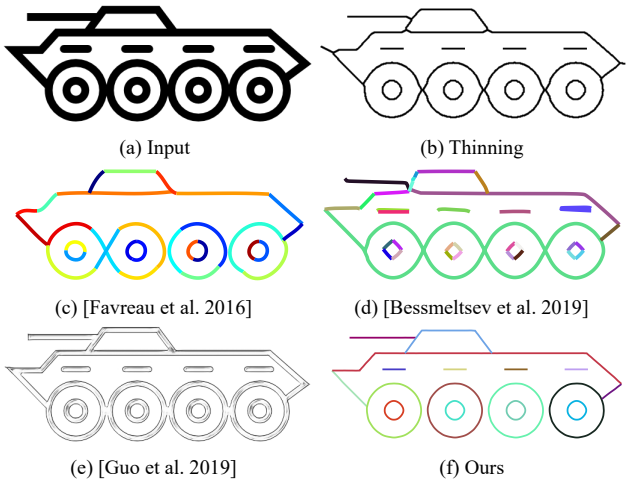


Figure 1: *Comparisons with existing methods.*

due to the limited receptive field (Figure 1(e)). In contrast, in this paper, we propose a method that can reconstruct the topology of line drawings of arbitrary thickness, as shown in Figure 1(f).

To analyze the topology of junctions composed of thick lines, we observe that, the boundaries of the lines can always provide useful hints in making the decisions. Figure 2 (a) and (b) show two similar line drawings where (a) is composed of two curves and (b) is composed of three curves. While it is not easy to differentiate the two cases using either flow-based or learning-based methods, the top boundary curve actually shows clear hints on the topology of this junction. In (a), the top boundary curve is a smooth curve which indicates that the horizontal curve should also be composed of only one curve. In (b), the top boundary curve has a sharp turn which indicates that the horizontal curve should be composed of more than one curve. Figure 2 (c) and (d) shown another pair of similar line drawings where the only difference is that there contains a very short spike curve in (d). While these two cases are also challenging for flow-based and learning-based methods, the topology is clearly indicated by the boundaries that (d) has an additional spike, while (c) doesn't.

In this paper, we propose a boundary-based topology reconstruction and vectorization method that converts a line drawing of arbitrary thickness to the vector representation. We first extract a rough skeleton of the input line drawing via a classic thinning method. The advantage of a thinning method is that it can deal with lines of arbitrary thickness. However, the extracted skeletons are not precise, especially in junctions and corners. Therefore, we propose to first identify whether a skeleton pixel is precise or not by analyzing the reliability of the skeleton pixels. Then we preserve the topology of reliable skeleton points and reconstruct the topology of unreliable skeleton points. We rely on the hints suggested by the boundaries to make the reconstruction. Through analyzing the mapping between the skeleton and the boundary, we may reconstruct the topology of junctions composed of arbitrary-thickness lines. We apply our method on line drawings of various shapes and styles. Convincing results are obtained.

2. Related Works

Vectorization methods can be classified into two types: region-based vectorization and stroke-based vectorization. The region-based vectorization methods target for mimicking the color regions in the raster images using parametric representations, such as gradient-filled cubic splines [LL06] and diffusion curves [OBW*08]. The stroke-based vectorization methods target for representing the raster image (usually line drawings or sketches) using vector strokes, such as lines, arcs, and Bézier curves. Our method belongs to the latter type.

To vectorize the strokes in a raster image, one straight-forward and classic approach is to first extract the skeleton of the strokes using the thinning method, and then fit the skeleton with parametric curves. The most well-known and classic method is the Zhang-Suen thinning algorithm [ZS84], further improved by much later research. One of the improved versions [LLS92] is still the built-in function in Matlab. An in-depth survey of the thinning algorithms can be found in [GV16]. Donati et al. [DCP19] proposed to change erosion strategies at concave angles to improve refinement performance, but it has limited applicability. The thinning methods usually do not focus on reconstructing the topology of the lines and only estimate a rough topology of the input line drawing. So, these methods generally fail to extract precise skeletons at junctions and corners, as shown in the second row of Figure 2.

To better estimate the topology of the line drawings, traditional vectorization methods proposed to fit the black pixels in the line drawings with pre-defined parametric primitives, such as lines [JV97], arcs [HT06], and cubic Bézier curves [CY98, FLB16, DCP17], via optimizations. However, these methods generally fail to reconstruct the topology of line drawings of arbitrary thickness, especially at junctions and corners, due to the uncertain thickness of lines. To better reconstruct the topology at junctions, de Goes et al. [dGCAD11] proposed to first construct a triangulation of the black pixels as the input point sample set using Delaunay triangulation and then reconstruct the topology of lines based on the optimal transport measurement between the reconstruction and the input point set. Noris et al. [NHS*13] proposed to reconstruct the topology of junctions in clean line drawings by searching for the best topology in all potential topology settings via reverse drawing. Chen et al. [CLMP15] extended this method for sketchy input. Recently, methods have been proposed to reconstruct the topology and vectorize line drawings based on frame fields [BS19], multi-scale edge detection [NS19], and global drawing-aligned integer grids [SBBB20]. However, these methods are usually tailored for resolving junctions composed of thin lines, and they usually fail to reconstruct the topology of line drawings composed of thick lines.

Lately, several learning-based methods have also been proposed to reconstruct the topology of line drawings [KWÖG18, SII18, GZH*19]. Kim et al. [KWÖG18] proposed to segment line drawings into semantic line segments. Simo-Serra et al. [SII18] proposed to convert sketches into clean line drawings in an interactive way. Guo et al. [GZH*19] proposed to reconstruct the topology of line drawings in a junction-tailored two-step approach. Mo et al. [MSSG*21] proposed to use a dynamic window around a virtual pen to draw lines. However, the major problem of these learning-based methods is either classification (with BCE loss) or

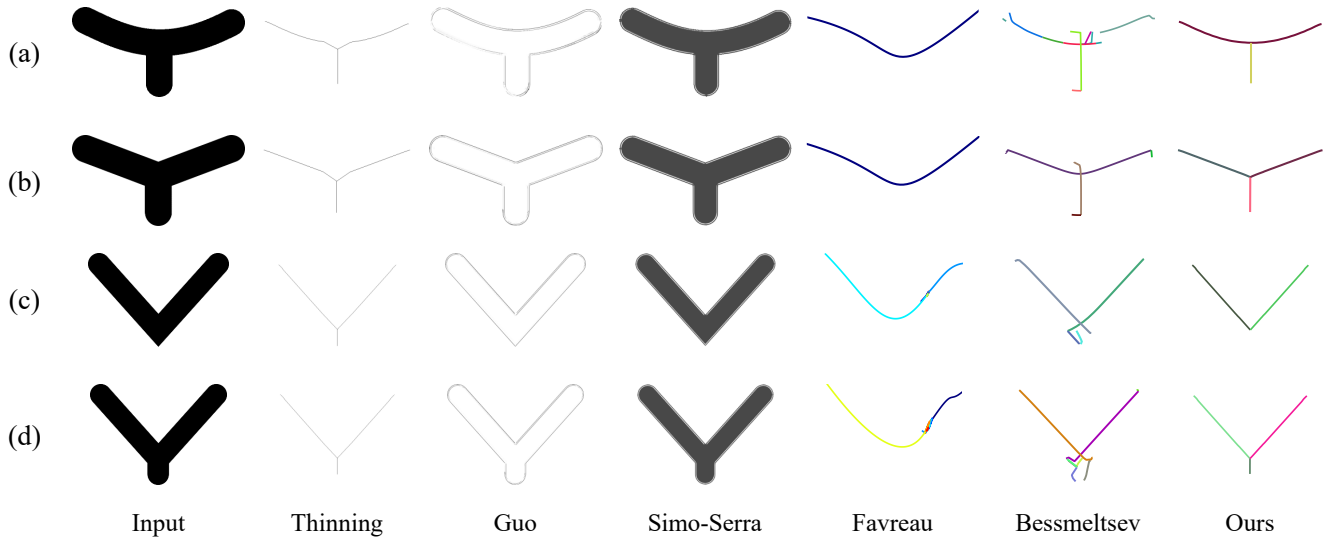


Figure 2: *Hints suggested by boundaries.*

regression (with MSE loss) of the centerline will be burdened by the failure of smooth mapping between the image space and the Cartesian space. Another problem lies in the failure of deep neural networks to both understand global trajectories from higher-level semantics (larger receptive fields) and extract precise centerline locations from lower-level semantics (smaller receptive fields). In contrast, in this paper, we propose a method that can reconstruct the topology of line drawings of arbitrary thickness.

3. Overview

The overview of our method is illustrated in Fig. 3. Given a raster input line drawing (Fig. 3(a)), we first binarize it into a black-and-white image to map each pixel into either a line pixel or a background pixel. Then we extract the skeleton of the line pixels via a classic thinning algorithm [ZS84], as shown in Fig. 3(b). Though the extracted skeleton pixels may not be consistent with the human-perceived centerline and topology, they can already provide good approximations to the ground-truth. Despite the imprecision of the extracted skeleton, we find that the boundaries of the input line drawings can be precisely extracted and provide significantly valuable hints to obtain the precise skeleton location and topology. To extract the boundaries, we capture the pixels with at least one 8-connected neighbor that is a background pixel, as shown in Fig. 3(c).

With the obtained skeleton pixels and boundary pixels, we can refine the skeleton based on the boundaries. We first construct segments from pixels for both the skeleton (Fig. 3(d)) and the boundary (Fig. 3(e)). These constructed segments and their mutual connectivity actually show an initial guess of the topology of the line drawing. However, this initial topology is usually incorrect at spikes, close lines, junctions, corners, sharp turns, and so on. So, we further refine the skeleton and reconstruct the topology based on the boundary. We first filter out all unreliable skeleton segments, such as falsely predicted spikes and connectors (Fig. 3(f)), based on our proposed skeleton-boundary tangent consistency constraint and boundary-guided skeleton completeness constraint to refine the topology. Then we reconstruct the skeleton segments at junction locations, corners, and sharp turns based on the boundaries again to obtain precise skeleton segments (Fig. 3(g)). The reconstructed line topology can well conform to the human's perception. Our skeleton extraction and topology reconstruction process are detailed in the following section.

4. Method

Our method contains four main steps: 1) extracting skeleton pixels and boundary pixels (introduced in Section 3); 2) constructing skeleton topology and boundary topology by constructing skeleton segments and boundary segments from pixels; 3) refining skeleton topology by removing unreliable skeleton segments; 4) reconstructing skeleton segments to obtain precise skeleton locations. We will introduce the details of the last three steps in this section.

4.1. Skeleton and Boundary Topology Construction

With the extracted skeleton and boundary pixels as stated in the previous section, we group the pixels into segments and construct segment-based topology diagrams for both the skeleton and the boundary. The constructed topology will be used for further refining the skeleton topology and reconstructing the skeleton segments.

4.1.1. Skeleton Topology Construction

To construct a segment-based topology of the skeleton, we first detect all endpoints (valence-1) and junction pixels (valence$\geq$3) from the skeleton pixels. We then identify the skeleton segments as a connected list of pixels if they are: (a) between two junction pixels; (b) between two endpoints; or (c) between an endpoint pixel

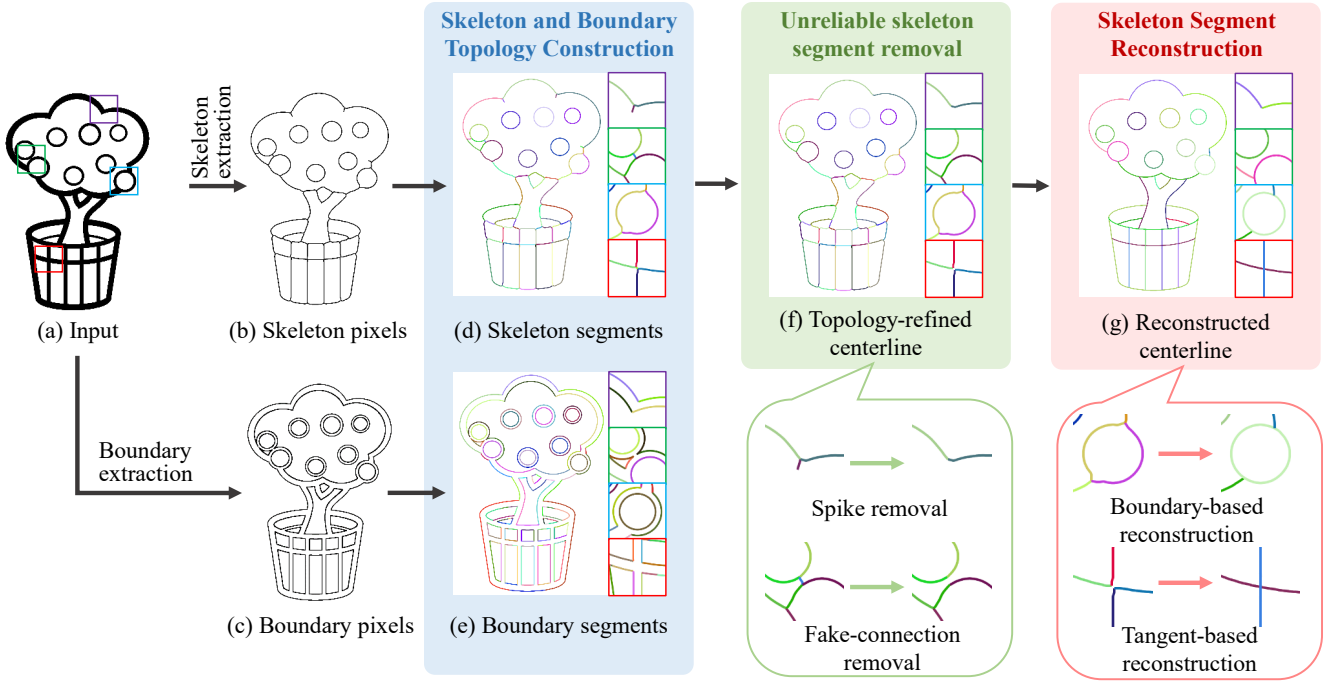


Figure 3: *An overview of our methods.*

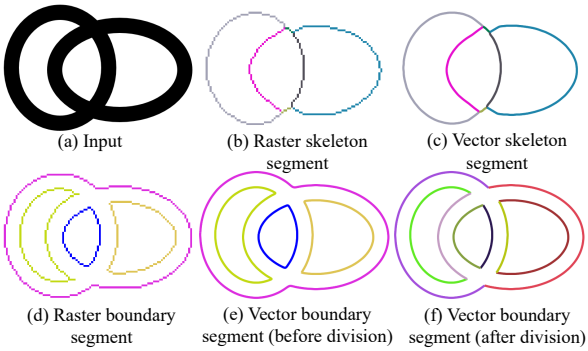


Figure 4: *Line topology construction for skeleton and boundary.*

and a junction pixel as the skeleton segments. Fig. 4(b) visualizes the identified skeleton segments where each line segment is color-coded with a different color. Due to the discrete nature of raster pixels, the identified skeleton segments are not smooth. They may provide incorrect information of local properties, such as tangent and curvature. Therefore, we fit each raster skeleton segment with an optimization-based smooth spline [DB78], where the smoothing parameter is set to 0.05. The obtained vector skeleton segments are visualized in Fig. 4(c).

4.1.2. Boundary Topology Construction

To construct the segment-based topology of the boundary, we first adopt a similar approach as identifying raster skeleton segments to identify the raster boundary segments, as shown in Fig. 4(d). However, the extracted boundary line segments may be composed of two different semantic boundary lines. For example, in Fig. 4(e), the magenta boundary segment is composed of two semantic boundary segments which are the boundaries of different circles. Therefore, we further subdivide each boundary segment based on the forward directions of the boundary pixels.

The straightforward idea is to calculate the turning angle for all boundary pixels and subdivide the boundary segments at pixels with a large turning angle. We first vectorize the raster lines to obtain precise tangents at all points. Directly adopting the gradient in the raster image may not give precise tangents at pixels near corners, junctions, or crowded lines. To vectorize the raster lines, we apply curve fitting to fit curves for the raster lines. During curve fitting, we subdivide the raster lines at sharp corners to avoid smoothing out the corners. To detect the corners, we identify the point on the fitted curve that deviates most from the raster line with more than 1-pixel offset as the pivot. Then we subdivide the raster line at the pivot and fit curves for the subdivided lines respectively. This process is repeated until no pivot is identified. The final vector boundary segments are visualized in Fig. 4(f).

4.2. Unreliable Skeleton Segment Removal

With the constructed skeleton topology (Fig. 5(b)), we can observe that some skeleton segments are not expressing the actual geometry of the raster line inputs. These unreliable line segments can be categorized as spurious spikes and spurious connectors, as shown in Fig. 5(c). The root cause of the problem is due to the design

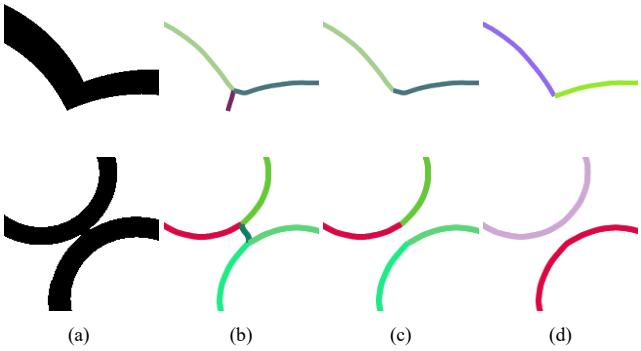


Figure 5: *Spurious spike removal (top row) and spurious connector removal (bottom row). (a) Input. (b) Skeleton topology. (c) Refined skeleton topology. (d) Output.*

flaws in the original thinning algorithm [ZS84]. The algorithm may squeeze the skeleton estimation at the corners or the locations with varying thicknesses. Consequently, the estimated skeleton may join earlier than the actual closing point of the raster lines and lead to spurious spikes. On the other hand, the algorithm will create spurious connectors as part of the skeleton if two thick lines are getting too close. We propose to eliminate these unreliable line segments based on two geometry constraint assumptions and elaborate the following procedures.

4.2.1. Spurious Spike Removal

To detect and remove spurious spikes, we first search through all skeleton segments to filter out ones with a valence-1 endpoint and a valence$\geq$3 endpoint as the spurious spike candidates. To classify the spurious spikes, we formulate two criteria: skeleton-initiated tangent consistency and boundary-guided skeleton completeness. If the line segment matches any either criterion, it will be eliminated from the skeleton topology.

4.2.1.1. Skeleton-initiated Tangent Consistency One critical observation at the spurious spike locations is that the tangent direction between the boundaries and the skeleton is usually inconsistent, as illustrated in the second and fourth columns of Fig. 6(c), where we can see the tangent directions of spikes are entirely inconsistent. On the contrary, we also illustrate consistent tangent directions between the boundaries and skeleton in the first and third columns of Fig. 6(c).

Based on this observation, we propose the skeleton-initiated tangent consistency at all spike candidate locations as a measurement to classify spurious spikes. To do so, we first extract a dense point list $\{c_1, c_2, \cdots\}$ for each candidate skeleton segment c by sequential sampling in the vector domain with a step size of 0.5 pixels, i,e., the distance between two consecutive points is always less than or equal to 0.5 pixels. Compared to the original raster representation of the line segment pixels, this dense point list is of float value coordinates, providing higher resolution for better spike estimations. Similarly, we can also extract a dense point list for each boundary line segment as $\{b_1, b_2, \cdots\}$. With the point lists extracted, for each

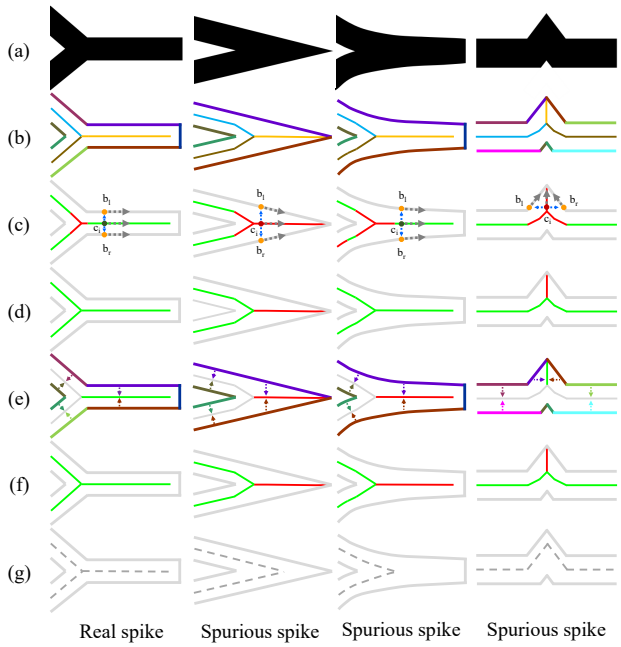


Figure 6: *The procedure of spike removal. (a) Input. (b) Line segmentation results. (c) Per-point tangent consistency estimation. (d) Per-line-segment tangent consistency estimation. Line segments with all points labelled as tangent inconsistent will be removed (shown in red color). (e) Boundary-guided skeleton completeness. The spike candidate in red color is to be removed, as it corresponds to the same boundary with its neighbors. (f) The final decision of line segment removal by summing up (d) and (e). The red color indicates removal. (g) The final output of our whole procedure, serving as a reference of the ground truth.*

skeleton point c_i in the list, we can find the corresponding boundary points by casting a line to the normal direction at the location of c_i. We label the left and right intersections to the boundaries as $b_l(c_i)$ and $b_r(c_i)$ according to the normal direction of c_i, as shown in Fig. 6(c). Based on this skeleton-boundary correspondence, we calculate the tangent consistency between c_i and its corresponding boundary points as

$$D_{skeleton}(c_i) = \max\left(\frac{t(c_i)\cdot t(b_l(c_i))}{|t(c_i)|\cdot|t(b_l(c_i))|}, \frac{t(c_i)\cdot t(b_r(c_i))}{|t(c_i)|\cdot|t(b_r(c_i))|}\right) \quad (1)$$

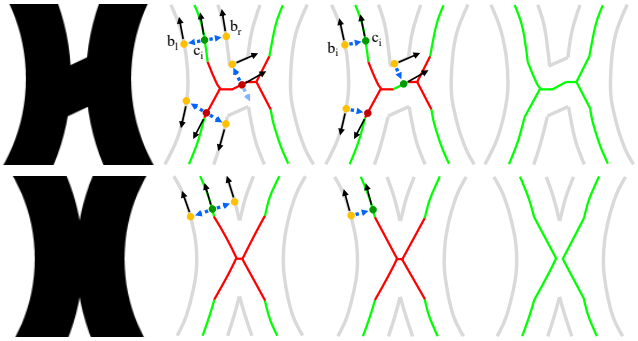
where t is the tangent operator computed as $(p_{i+1,j+1} - p_{i,j})$, and $p_{i,j}$ is the coordinate of point p.

If the tangent consistency of all points on a spike candidate c are smaller than a fixed threshold, i.e.

$$\forall c_i \in c,\ D_{skeleton}c_i < t_c, \quad (2)$$

this spike candidate will be classified as a spurious spike and deleted from the topology. t_c is set to 0.99 in all our experiments.

We illustrate the tangent consistency based classification of spurious spikes in Fig. 6(d), where the red color in (f) indicates the line segment to be removed.



(a) Input (b) Skelton-initiated consistency (c) Boundary-initiated consistency (d) Connector removal

Figure 7: *Spurious connector removal. (a) Input. (b) Skeleton-initiated tangent consistency. (c) Boundary-initiated tangent consistency. Green and red colors represent per-point consistency. We remove the line segment whose points are all tangent-inconsistent to form the result (d). The top row is a real connector case that contains tangent-consistent points at the connector. The bottom row is a spurious connector case where all points are tangent-inconsistent. Note that the result will be further refined in the next reconstruction part.*

4.2.1.2. Boundary-guided Skeleton Completeness The boundary information is also helpful to distinguish spurious spikes. The traditional thinning algorithm tends to squeeze the extension of skeletons at corner locations. Under such a circumstance, the two skeletons near the corner will be combined unexpectedly, causing the creation of a spurious spike. We illustrate this issue in the last three cases in Fig. 6(e). Tangent consistency is not enough for removing such spurious spikes as the spikes may be long and contain tangent-consistent skeleton pixels, as shown in the third column of Fig. 6(e). To remove this spurious spike, we further examine the continuity of the skeleton segment to its corresponding boundaries. If a single boundary segment near the corner is correlated to multiple skeleton segments, these skeleton segments are likely to be related to a spurious spike. Specifically, we define a skeleton segment c and a boundary segment b are correlated, if only for any point c_i in c we can find its counterpart point b_i on b on the normal direction of c_i. We then check all boundary segments in the topology to find if more than one skeleton segment is correlated to it, i.e.

$$|unique(Ind(c(b_r(c_i))))| > 1 \wedge |unique(Ind(c(b_l(c_i))))| > 1, \quad (3)$$

where $c(b_i)$ indicates the skeleton point which is correlated to b_i, $b_l(c_i)$ and $b_r(c_i)$ are the left and right intersections to the boundaries according to the normal direction of c_i as defined in Section 4.2.1.1, Ind(x) indicates the segment index of x, unique$(\cdot)$ is the duplicate removal operator, and $|\cdot|$ is the cardinality operator. If so, we label all these skeleton segments as failing the boundary-guided completeness criteria.

If the skeleton segment fails the skeleton-initiated tangent-consistency criteria or the boundary-guided completeness criteria ($\forall c_i \in c,\ D_{skeleton}c_i > t_c \vee |unique(Ind(c(b_r(c_i))))| > 1 \wedge |unique(Ind(c(b_l(c_i))))| > 1$), it will be classified as a spurious spike and removed.

4.2.2. Spurious Connector Removal

Besides the spurious spikes, we also handle the mistakenly estimated spurious connectors during thinning-based skeleton extraction when two thick lines are too close to each other. To identify the spurious connectors, we first find all the skeleton segments with two valence$>$1 endpoints as candidates. For each candidate, we distinguish a spurious connector by two kinds of tangent consistency estimations: one is the previously mentioned skeleton-initiated tangent consistency, and the other is a new boundary-initiated tangent consistency, as shown in Fig. 7(b)&(c) respectively. The boundary-initiated tangent consistency is computed similarly to the skeleton-initiated one, but the computation originated from the boundary line segment points. For each skeleton point c_i, we search through all boundary point b_i where the normal direction of b_i will cast to c_i and compute the boundary-initiated tangent consistency score as

$$D_{boundary}(c_i) = max(\frac{t(c_i) \cdot t(b_i)}{|t(c_i))| \cdot |t(b_i)|}), \forall b_i \text{ casting to } c_i \quad (4)$$

where t is the tangent operator. The reason for adopting a new boundary-initiated tangent consistency is because connectors are usually very short and very likely to deviate from the actual location during the thinning. With skeleton-initiated tangent consistency only, one may not be able to find the correct boundary points correlated to this skeleton point and therefore is unable to determine whether it is a spurious connector. With the additional boundary-initiated tangent consistency, this short-segment problem can be well resolved.

In this way, we can compute the two tangent consistency scores for each skeleton point c_i. If both tangent consistency cannot be satisfied, the skeleton segment will be classified as a spurious connector. In our experiment, if $D_{boundary}(c_i)$ is lower than 0.95 and $D_{centered}(c_i)$ is lower than 0.99, we shall regard this point as tangent-inconsistent. If all points of the skeleton segment are tangent-inconsistent, we will label this skeleton segment as a spurious connector and remove it from the topology, as shown in Fig. 7(d).

4.3. Skeleton Segment Reconstruction

After removing the spurious spikes and spurious connectors, we expect the topology of the lines to be correct. However, we still observe distorted curvature and imprecise skeleton points, especially at corners, junctions, and sharp turns. For example, the remaining green skeleton in Fig. 6(f) after the spurious spike removal is still not reflecting the ground-truth. The issue is more severe in Fig. 7(d) after the spurious connector removal. Moreover, these artifacts are more random and cannot be determined on a segment basis. As a result, we propose to classify the skeleton pixels that may be deviated from the ground-truth. Then we replace these unreliable pixels with more accurate ones derived either from the boundary or the adjacent reliable skeleton pixels.

To be specific, skeleton pixels that are skeleton-initiated tangent-inconsistent are classified as the unreliable pixel. Moreover, for a

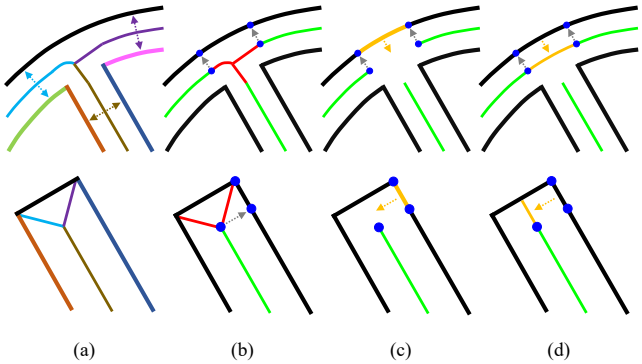


Figure 8: *Boundary-aided reconstruction. We first find out the reference boundary line segment in (a) and remove the unreliable points by checking the tangent consistency in (b). We then copy a segment of the reference in parallel to the removed skeletons in (c), translate and paste it back to complete the skeleton in (d).*

skeleton pixel with no corresponding boundary point, we also label it as unreliable. In this manner, we remove all unreliable skeleton pixels. So, most of the mistakenly estimated skeletons will be removed, and most of the junctions will be disconnected. Finally, we propose a boundary-aided reconstruction to repair the skeletons with corresponding solid boundaries and a topology-aided reconstruction to repair the remaining skeletons, whose reconstruction cannot be derived from the boundaries.

4.3.1. Boundary-aided Reconstruction

As shown in the top row of Fig. 8(a), the thinning algorithm may create fluctuated curves near the junction by its nature, which is a false estimation. A good skeleton extraction should eliminate the fluctuation and correctly connect the two line segments with smoothly changing tangents near the junction. However, it is challenging to make a reasonable assumption of the junction location and the tangent value at each skeleton point by only observing the thinned skeleton segments. Fortunately, we can still recover almost accurate junction locations and tangent values from the boundaries. For example, in the top row of Fig. 8(b), we can recover the accurate skeleton from the boundary segments corresponding to the removed unreliable pixels.

Concretely speaking, for each junction, we find all skeleton segments connected to this junction. Then for each of these skeleton segments, we find the closest reliable point to the junction as the cut-off point. If any two cut-off points have the same corresponding boundary segment, i.e., the two skeleton segments are continuous in boundary, we connect these two skeleton segments at the cut-off points by copying the segment from the boundary. As illustrated in the top row of Fig. 8(c), we intercept part of the boundary segment corresponding to the cut-off point, translate and rescale it as a new segment and paste it back to the missing locations to finish the patching. The two skeleton segments will be merged into one skeleton segment in the topology after the connection. Similar operations are performed at the unreliable part near the line endpoints, as shown in the bottom row of Fig. 8(c).

4.3.2. Topology-aided Reconstruction

Even though we can recover most of the accurate skeleton with the boundary-aided reconstruction, there are still some skeleton estimation mistakes that cannot be recovered due to the absence of boundary correspondence. For example, in the top row of Fig. 8(d)), the straight line segment cannot be connected to the reconstructed skeleton curve, as there is no boundary line to guide the reconstruction of the missing skeleton segment. This problem is also obvious at the X-junction locations (Fig. 9(a)), where the initially estimated skeleton is twisted and should be reconstructed. However, there are barely any boundary cues that can be used for this X-junction skeleton reconstruction. We propose to apply a blind skeleton reconstruction from the broken skeleton itself. Firstly, we eliminate all skeleton segment points around junctions if: a) it is not skeleton-initiated tangent consistent to its corresponding boundary; b) it is not even corresponding to a boundary line segment, as shown in Fig. 9(a), where red color indicate a point removal.

To recover the topology from the remaining broken lines, we first detect the tangent direction of the cut-off points. If the tangent directions of two cut-off points are co-linear (within a 0.01 cosine distance between two tangents, as with Eq. 4), we instantly connect the cut-off points with a straight line to recover the topology and merge the corresponding skeleton segments into one segment, as shown in the first row of Fig. 9(b).

For those non-matched skeleton segments near the junction, as shown in Fig. 10(a)&(b), we propose to enforce the topology reconstruction with a skeleton extrapolation strategy. To do so, we use the same vector-based skeleton sampling approaches to construct a point list of the currently broken skeleton segment. After that, we apply polynomial curve fitting to the point list to approximate the analytical representation of the skeleton segment. In our experiment, we set the degree to 2, which is enough for our objectives. After the line fitting, we extend all broken skeletons from the cut-off point until they reach the boundaries, as shown in Fig. 10(c). During the extension, the broken topology is automatically restored, with the skeleton extension reconstructing the junctions. Moreover, the analytical curve fitting also ensures smooth varying tangents at all skeleton locations. Finally, we clean up all over-extended skeletons from the farthest extended junction point to the boundary to conclude the topology reconstruction, as shown in Fig. 10(d). At this stage, all skeletons are extended and cut off based on curvature and intersection only. Cut-off points that are very close will be merged into one junction point for beautification purposes at a later stage.

A complete centerline extraction process is shown in Fig. 11.

5. Results and Discussions

5.1. Dataset Preparation

Since there is no publicly available dataset with line drawings of different thickness and the corresponding ground-truth skeletons, we hereby prepare such a dataset to evaluate our skeleton extraction performance. To do so, we first rasterize arbitrarily constructed Bézier curves into 1-pixel-wide strokes in a canvas as ground-truth

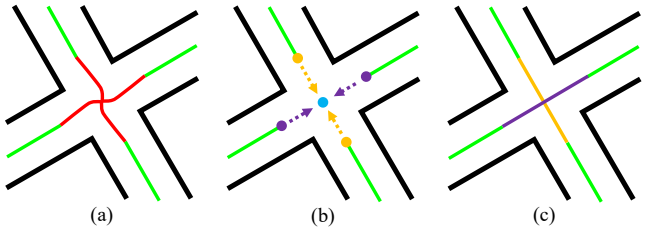


Figure 9: *Topology-aided skeleton reconstruction with matched tangents at cut-off points. (a) Per-point skeleton-initiated tangent consistency, with inconsistent points shown in red. (b) Tangent direction matching at cut-off locations. (c) Result.*

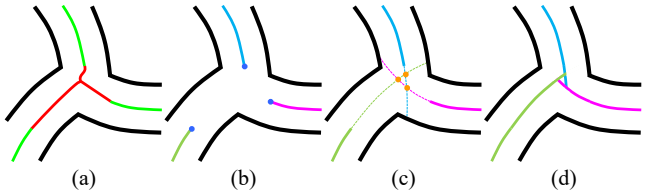


Figure 10: *Blind topology reconstruction with unmatched tangents at cut-off points. (a) Per-point skeleton-initiated tangent consistency, with inconsistent points shown in red. (b) The remaining skeleton segments and the cut-off points after inconsistent point removal. (c) Skeleton extension based on curve fitting. The newly formed junctions are also illustrated. (d) Skeleton reconstruction by extending the skeleton to the farthest junction.*

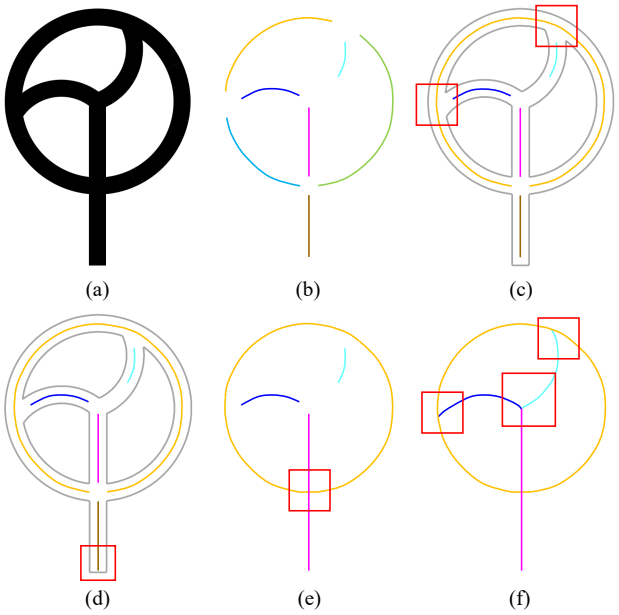


Figure 11: *A complete flow of our method. (a) Input. (b) Skeleton segments after removing unreliable skeleton pixels. (c) Boundary-aided reconstruction of unreliable skeleton pixels at junctions. (d) Boundary-aided reconstruction of unreliable skeleton pixels at endpoints. (e) Topology-aided reconstruction of unreliable pixels via the tangent-based connection. (f) Topology-aided reconstruction of unreliable pixels via curve extension.*

skeletons. Then, we synthesize line drawings from the ground-truth skeletons with different thickness based on three rules: a) Some lines are synthesized with fixed thickness; b) Some lines are synthesized as constantly width-growing or width-reducing; c) The rest lines are synthesized with smoothly changing thickness controlled by a polynomial function. For all these strategies, we use a parameter w to control the average thickness for synthesis and we expect the average thickness for all lines in the image will distribute uniformly centered at w. In this way, we have more intuitive and targeted test data and ground-truth. All test images are synthesized on a canvas size of 512 by 512. The average count of synthesized curves for each image is 2.1. To test the robustness of centerline extraction under the condition of different line widths, we separately generate three subsets, with w set to 10, 20, and 30, respectively. For each subset, we synthesize 150 images.

Moreover, to observe the centerline extraction quality in real-life cases, we collect around 100 internet line drawings with varying stroke widths. Based on the hybrid dataset of both synthetic and real line drawings, we compare our method to a list of competitors, including three traditional methods [FLB16, BS19, ZS84] and one learning-based method [GZH*19], qualitatively and quantitatively.

5.2. Qualitative Evaluations

We conduct visual comparisons in skeleton extraction on high-quality internet images, including fixed-thickness images (e.g., Fig. 12) and varying-thickness images (e.g., Fig. 13). The fixed-thickness images are obtained from rasterized vector graphics and the varying-thickness images are obtained from hand-made drawings.

Favreau's method [FLB16] tends to output over-segmented skeleton segments. Moreover, it cannot handle complex junctions and non-closed strokes without user interaction, as highlighted in the red boxes of Fig. 12(b). Bessmeltsev's method [BS19] cannot handle thick line strokes properly, causing double estimation of skeletons, as shown in the first row of Fig. 12(c). Moreover, this method cannot handle closed strokes and tends to output over-segmented results with spurious spikes.

Guo's method [GZH*19] is learning-based. Hence, the overall skeleton extraction performance is highly dependent to the model architecture and the training set. We find that their model architecture is unable to handle very thick strokes for two main reasons. The first reason is that their skeleton extraction model has limited receptive field and cannot handle thick strokes well. The second reason is that the junction of thick strokes will confuse their line ordering model, so the topology reconstruction will also fail. The same problem occurs with Simo-Serra's method [SII18], which cannot output lines with a width of 1 pixel when the input is thick lines, as shown in Fig. 12(e). We attempted to re-train their model with our dataset. However, we found that the overall quality is severely reduced regardless of the average width of strokes. Thus we keep

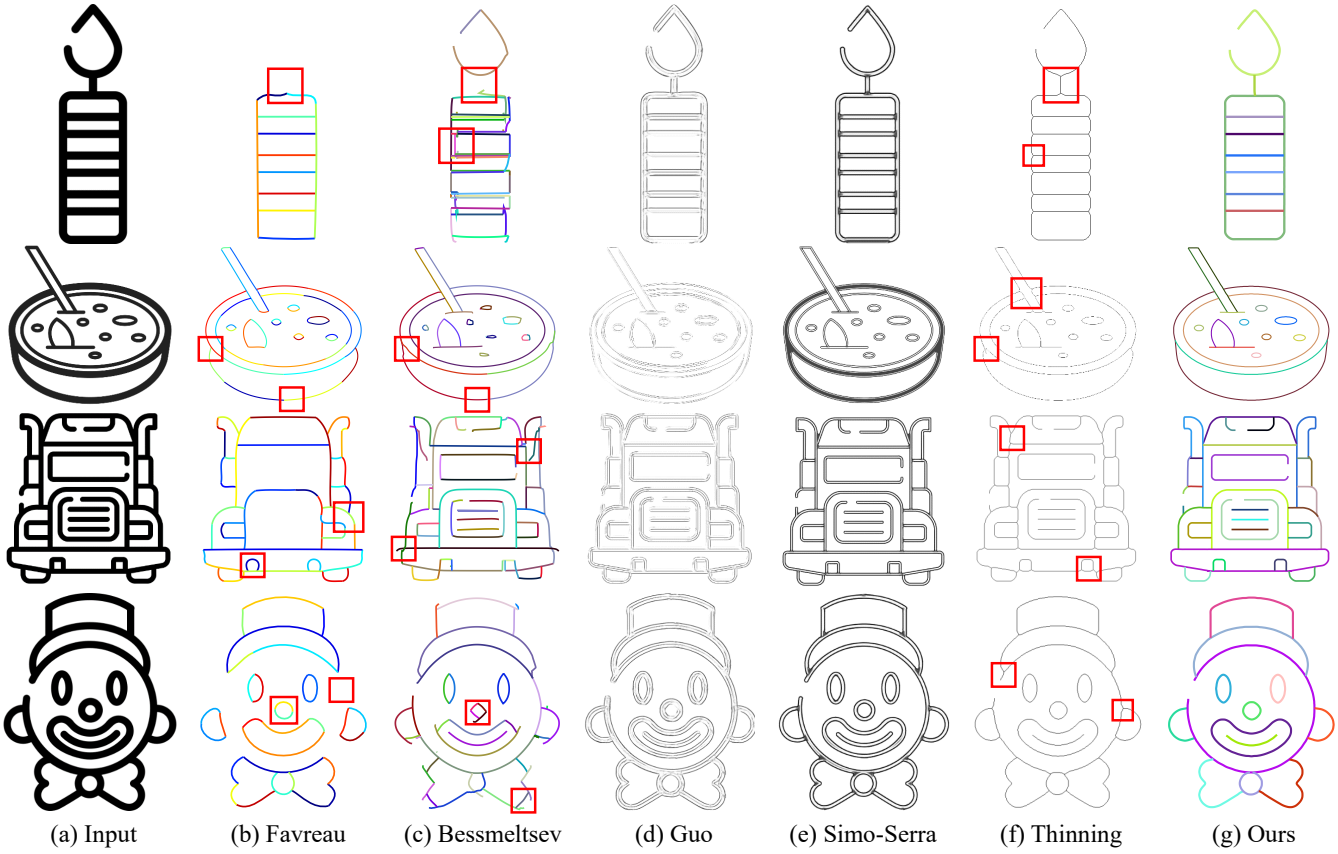


Figure 12: *Comparison to existing method on the fixed-width line drawing. The red boxes show part of the inaccuracies.*

using the original model weight shipped in [GZH*19, SII18] and illustrate their results in the comparison.

We also provide the results of the classical thinning algorithm. We would like to highlight their major flaws, including the early centerline combination, spurious spikes and distorted centerline orientation near junctions. Finally we show our results in Fig. 12(f) and Fig. 13(f). Our method successfully extracts clean and accurate skeletons and is free from common structural estimation mistakes such as spurious spikes, spurious connectors, and double-edge extraction. Moreover, our method is considerably more precise at junctions, compared to all other competitors. We also show more complex examples in Fig. 14.

5.3. Quantitative Evaluations

We also perform quantitative evaluations with our synthetic dataset, as the real-life images are usually not shipped with the ground-truth skeleton. To evaluate the extraction quality, we first compute the Chamfer distance between the extracted centerline and the ground truth. Chamfer distance is intuitively conforming to human perception and is widely used as the distance metric in shape matching. Compared to pixel-wise metrics such as PSNR and MAE, Chamfer distance is much more robust to subtle spatial drifts (referred to as "nudges"). That is, due to the grid nature of rasterization (i.e., aliasing), the 1-pixel-wide estimated centerline pixels might not be fully overlapping the ground truths but with a slight drift. In this case of nearly precise estimation, the pixel-wise PSNR metric will drastically increase, while Chamfer distance will maintain the numerical stability. The Chamfer distance is defined as:

$$C(A,B) = \frac{1}{2|A|}\sum_{i,j\in A} DT_B[i,j] + \frac{1}{2|B|}\sum_{i,j\in B} DT_A[i,j] \tag{5}$$

where A and B are two binary images representing the prediction and the ground truth. DT_A is the normalized distance transformation of A. The perfect matching distance of the Chamfer distance is 0, and two maximally dissimilar images have a distance of $\sqrt{2}$.

While the Chamfer distance is an effective perspective to estimate the centerline extraction quality, it may sometimes give inaccurate distance estimation if some components of B do not exist in A and vice versa. Due to the sensitivity of shape completeness in Chamfer distance, we compute an occupancy metric as the other perspective of the skeleton extraction precision. The occupancy counts the relaxed precision score of skeleton prediction with the ground-truth skeleton dilated to 3-pixel-wide. Specifically, we define the occupancy as

$$Occupancy(A,B) = \frac{A \cap d_3(B)}{d_3(B)} \tag{6}$$

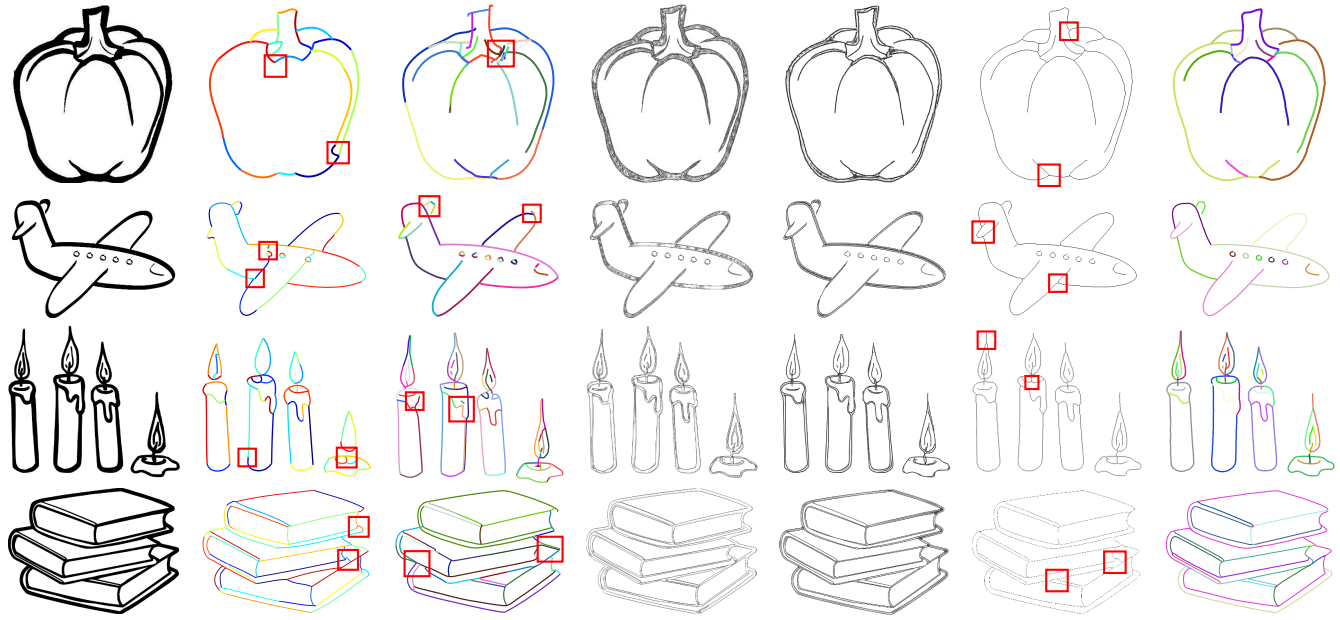


Figure 13: *Comparison to existing method on the variable-width line drawing. The red boxes show part of the inaccuracies.*

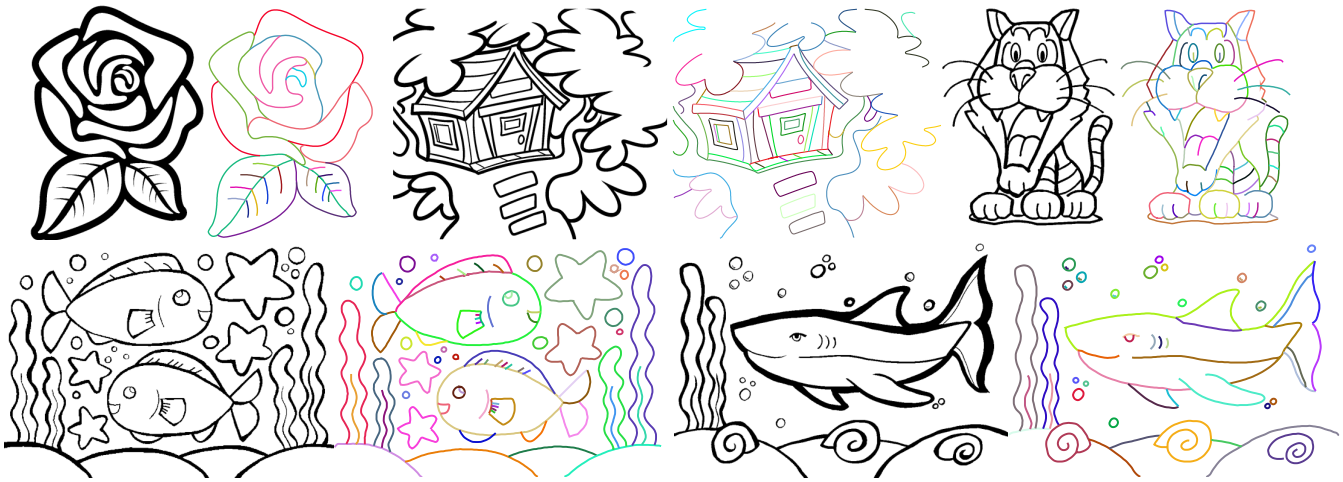


Figure 14: *More Results.*

where A is the precision, B is the ground truth, $d_k(I)$ is the dilation operator to image I with a kernel size of k.

We show the quantitative comparison results in Table 1. Even though the pixel-wise metrics may not faithfully reflect the precision of centerline extraction, we keep it here as a supplement. We can see our method outperforms the competitors under almost all conditions, except for slightly worse pixel-wise metrics on thin lines. We believe the aliasing problem causes this. Also, our method manages to maintain a comparable high precision on thick lines, while the competitors are imprecise or even refuse to work.

Besides an averaged quantitative study of extraction precision, we also perform a targeted study to investigate the skeleton extraction quality at junction locations. Most of the unreliable skeleton estimations happen on the junction location because the junctions are of complex topology, which increases the estimation ambiguity. Additionally, there are fewer boundary cues near the junction locations, which further increases the estimation ambiguity. Thus, we present an evaluation on the skeleton extraction performance in regard to the distance of the prediction skeleton pixel to its nearest junction. As shown in Fig. 15, our method achieves mostly the best

Table 1: *Quantitative analysis of different methods with datasets of different average line widths. For PSNR, SSIM and occupancy scores, higher is better. For MAE and Chamfer distance, lower is better. The numbers in brackets stand for the standard deviation.*

Method	Average line width = 10					Average line width = 20					Average line width = 30					All line widths				
	PSNR	SSIM	MAE	Occupancy	Chamfer Distance	PSNR	SSIM	MAE	Occupancy	Chamfer Distance	PSNR	SSIM	MAE	Occupancy	Chamfer Distance	PSNR	SSIM	MAE	Occupancy	Chamfer Distance
Favreau	22.59 (1.08)	0.933 (0.019)	1.45 (0.36)	42.78 (0.41)	0.06609 (0.03661)	22.77 (0.97)	0.935 (0.017)	1.38 (0.27)	46.95 (0.41)	0.06319 (0.02917)	22.40 (1.03)	0.929 (0.023)	1.51 (0.38)	40.20 (0.37)	0.06619 (0.03893)	22.59 (1.03)	0.932 (0.020)	1.45 (0.34)	43.31 (0.40)	0.06516 (0.03490)
Bessmeltsev	23.00 (0.83)	0.954 (0.008)	1.30 (0.25)	0.77.32 (0.08)	0.00229 (0.00081)	21.82 (0.75)	0.935 (0.012)	1.70 (0.29)	48.90 (0.15)	0.00752 (0.00521)	22.04 (0.59)	0.936 (0.006)	1.61 (0.22)	31.18 (0.09)	0.00835 (0.00386)	22.29 (0.72)	0.942 (0.010)	1.54 (0.023)	52.47 (0.08)	0.00605 (0.00329)
Guo	**26.99** (2.32)	**0.976** (0.011)	**0.58** (0.28)	90.54 (0.11)	0.00314 (0.00353)	21.10 (0.91)	0.911 (0.018)	2.02 (0.42)	0.01 (0.00)	0.01922 (0.01124)	21.10 (0.92)	0.885 (0.023)	2.03 (0.42)	0.00 (0.00)	0.02737 (0.00780)	23.06 (1.38)	0.924 (0.017)	1.54 (0.37)	30.18 (0.04)	0.01658 (0.02257)
Thinning	26.12 (1.30)	0.975 (0.007)	0.65 (0.20)	93.83 (0.06)	0.00111 (0.00102)	25.13 (1.01)	0.969 (0.007)	0.80 (0.19)	87.40 (0.08)	0.00242 (0.00183)	24.17 (1.17)	0.961 (0.012)	1.02 (0.31)	79.92 (0.12)	0.00371 (0.00289)	25.23 (1.16)	0.968 (0.009)	0.82 (0.23)	87.05 (0.08)	0.00241 (0.00191)
Ours	26.26 (1.38)	0.975 (0.007)	0.64 (0.21)	**95.45** (0.06)	**0.00105** (0.00104)	**25.40** (1.17)	**0.971** (0.007)	**0.76** (0.21)	**91.09** (0.08)	**0.00223** (0.00200)	**24.46** (1.21)	**0.963** (0.011)	**0.95** (0.29)	**85.45** (0.11)	**0.00335** (0.00308)	**25.28** (1.25)	**0.970** (0.008)	**0.78** (0.24)	**90.66** (0.08)	**0.00221** (0.00204)

accuracy on all occasions. Moreover, our method is the most robust among all methods.

We evaluate the robustness of different image sizes and different tangent-consistency thresholds. We select the data with size 512×512 and line width of 20 pixels and resize them to size 256×256 and 1024×1024 to evaluate the performance of our method on images of different resolutions. The evaluation results are shown in Table 2. We can see that our method performs stably on the input with different resolutions. To evaluate the influence of the tangent-consistency threshold on the results, we performed ablation experiments with three different tangent-consistency thresholds of 0.95, 0.99, 0.9999 at the size of 512×512 and the line width of 20 pixels for the experiment, as shown in Table 3. When the threshold is too small, the reliability standard is very lax, and many pixels that are not accurate will be included in the reliability segment and retained. When the threshold is too large, the reliability standard is so strict that most and even all pixels will be classified as unreliable and the basis for reconstruction cannot be found. From the perspective of occupancy and Chamfer distance, a too-large threshold may have a larger effect on our method than a too-small threshold. Still, we can see from the statistics that our method is generally stable under different thresholds.

We also perform a time analysis with a testbed with an Intel i7-9700K @ 3.0GHz CPU and 16GB RAM. Note that our method does not require GPUs. For the learning-based method [GZH*19], we perform model inference with a single NVIDIA 2080 Ti GPU. We compute the average time on the combined dataset of various average line widths. Moreover, we rescale our dataset to different resolutions to study the computational complexity. We illustrate the study results in Table 4. Our method is fairly efficient with the second fastest overall computational time on CPU, and is of linear complexity to the side of the image. While [FLB16, BS19] takes a significantly longer time to complete the vectorization.

5.4. Limitations

While the boundaries of the line drawings are usually smooth and give clear hints on the topology of the lines, it happens that boundaries of the lines are not smooth because of artistic styles or low-quality images. Since our method is highly dependant on boundaries to obtain the correct topology with precise skeleton locations, it will fail when the boundaries are not smooth and unable to give correct hints, as shown in the first row of Fig. 16(a) and (b). Moreover, some very short and tangent-inconsistent skeleton segments might be mistakenly classified as spurious spikes and removed, as shown in the nose of the bear in Fig. 16 (c) and (d). Our method is not designed for sketchy images. Sketches require semantically simplified lines, and our method preserves each curve faithfully, as shown in Fig. 16 (e) and (f), an input from Yan's benchmark of sketch cleanup [YVG20] and its result.

6. Conclusions

We propose a vectorization method of line drawings via boundary-based topology reconstruction. The key of our method is to construct and refine the topology based on boundary. In particular, we propose to first obtain the initial topology by extracting the skeleton segments. Then we remove the unreliable skeleton segments based on the guidance of the boundaries. Finally, both the skeleton segments and topology are refined based on the boundary information again. Compared to the existing methods, our method performs better both visually and quantitatively, especially in line drawings composed on thick lines. Our method is more efficient than the other traditional methods and can be easily run in almost any experimental environment. We will further explore to apply our method on more stylized line drawings in future research.

Acknowledgments

This work was supported partly by National Natural Science Foundation of China (No. 62002232 and No. 61973221), Natural Science Foundation of Guangdong Province, China (No. 2019A1515011165), the COVID-19 Prevention Project of Guangdong Province, China (No. 2020KZDZX1174), the Major Project of the New Generation of Artificial Intelligence (No. 2018AAA0102900), the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. UGC/FDS11/E01/21), and the Institutional Development Grant of Caritas Institute of Higher Education, Hong Kong Special Administrative Region, China (Project No. IDG200107).

References

[BS19] BESSMELTSEV M., SOLOMON J.: Vectorization of line drawings via polyvector fields. *ACM Trans. Graph. 38*, 1 (2019), 9:1–9:12. 1, 2, 8, 11

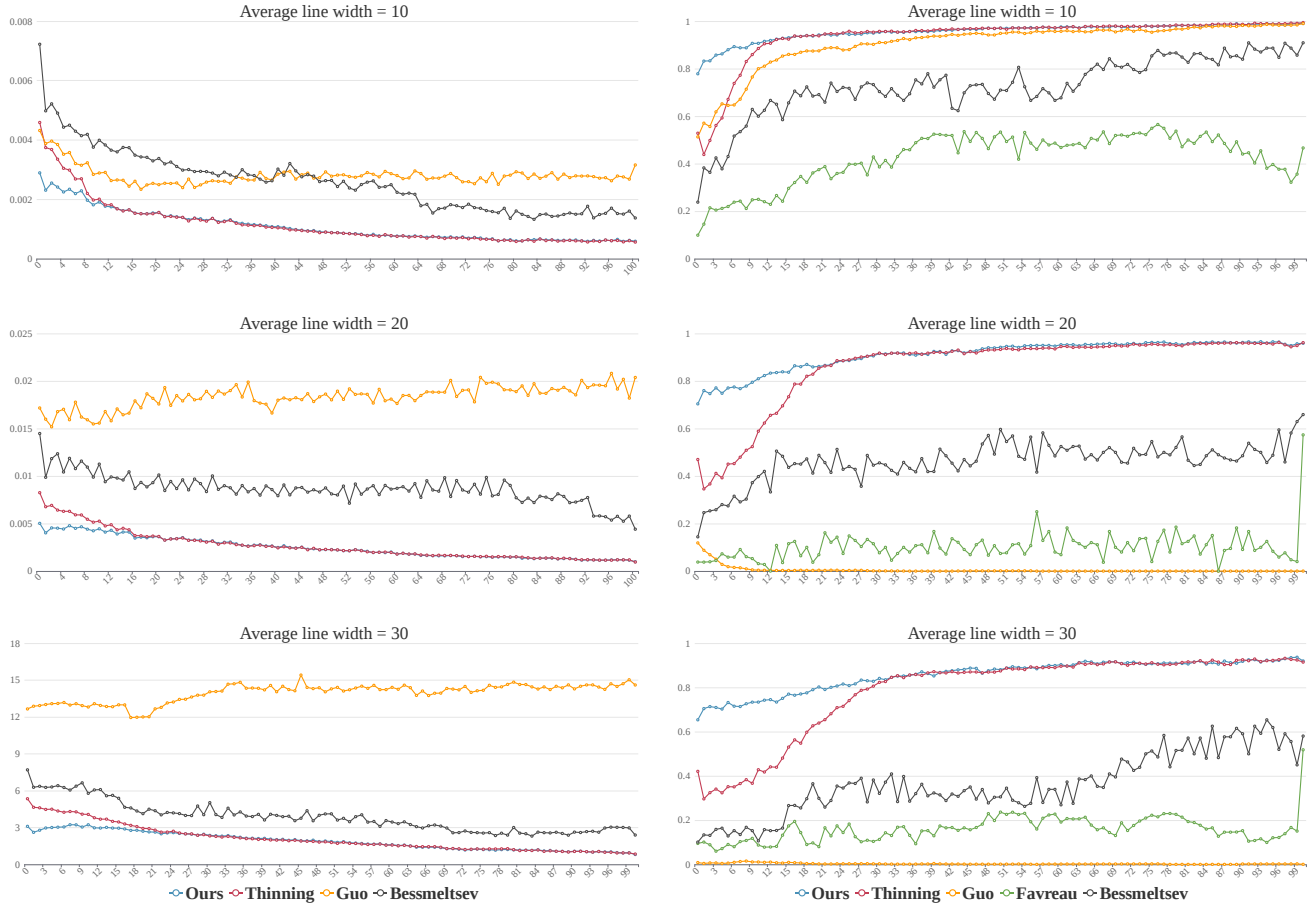


Figure 15: *Chamfer distance (left column) and occupancy (right column) are in regard to the distance from junctions. Ours stays the most robust among all method. The x-axes stand for the distance from any point to its nearest junction and the y-axes stand for the occupancy and Chamfer distance respectively.*

Table 2: *Robustness evaluation of our method with datasets of different resolutions. For PSNR, SSIM and occupancy scores, higher is better. For MAE and Chamfer distance, lower is better. The numbers in brackets stand for the standard deviation.*

256 × 256					512 × 512					1024 × 1024				
PSNR	SSIM	MAE	Occupancy	Chamfer Distance	PSNR	SSIM	MAE	Occupancy	Chamfer Distance	PSNR	SSIM	MAE	Occupancy	Chamfer Distance
23.55 (1.62)	0.955 (0.015)	1.21 (0.48)	90.85 (0.09)	0.00114 (0.00101)	25.40 (1.17)	0.971 (0.007)	0.76 (0.21)	91.09 (0.08)	0.00223 (0.00200)	27.03 (0.92)	0.980 (0.004)	0.52 (0.11)	91.10 (0.08)	0.00370 (0.00374)

[CLMP15] CHEN J., LEI Q., MIAO Y., PENG Q.: Vectorization of line drawing image based on junction analysis. *Sci. China Inf. Sci. 58*, 7 (2015), 1–14. 2

[CY98] CHANG H., YAN H.: Vectorization of hand-drawn image using piecewise cubic bézier curves fitting. *Pattern Recognit. 31*, 11 (1998), 1747–1755. 1, 2

[DB78] DE BOOR C.: *A practical guide to splines*, vol. 27. springer-verlag New York, 1978. 4

[DCP17] DONATI L., CESANO S., PRATI A.: An accurate system for fashion hand-drawn sketches vectorization. In *2017 IEEE International Conference on Computer Vision Workshops, ICCV Workshops 2017, Venice, Italy, October 22-29, 2017* (2017), IEEE Computer Society, pp. 2280–2286. 1, 2

[DCP19] DONATI L., CESANO S., PRATI A.: A complete hand-drawn sketch vectorization framework. *Multimedia Tools and Applications 78*, 14 (2019), 19083–19113. 2

[dGCAD11] DE GOES F., COHEN-STEINER D., ALLIEZ P., DESBRUN M.: An optimal transport approach to robust reconstruction and simplification of 2d shapes. *Comput. Graph. Forum 30*, 5 (2011), 1593–1602. 2

[FLB16] FAVREAU J., LAFARGE F., BOUSSEAU A.: Fidelity vs. simplicity: a global approach to line drawing vectorization. *ACM Trans. Graph. 35*, 4 (2016), 120:1–120:10. 1, 2, 8, 11

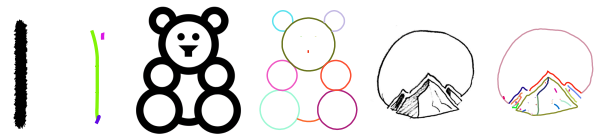
[GV16] GRAMBLICKA M., VASKY J.: Comparison of thinning algo-

Table 3: *Ablation experiment of our method with different tangent-consistency threshold. For PSNR, SSIM and occupancy scores, higher is better. For MAE and Chamfer distance, lower is better. The numbers in brackets stand for the standard deviation.*

tangent-consistency threshold = 0.95					tangent-consistency threshold = 0.99					tangent-consistency threshold = 0.9999				
PSNR	SSIM	MAE	Occupancy	Chamfer Distance	PSNR	SSIM	MAE	Occupancy	Chamfer Distance	PSNR	SSIM	MAE	Occupancy	Chamfer Distance
25.56 (1.20)	0.973 (0.008)	0.74 (0.21)	90.06 (0.08)	0.00205 (0.00184)	25.40 (1.17)	0.971 (0.007)	0.76 (0.21)	91.09 (0.08)	0.00223 (0.00200)	25.41 (1.33)	0.970 (0.009)	0.77 (0.25)	87.92 (0.11)	0.00251 (0.00240)

Table 4: *Timing statistics on differnt image resolutions.*

	256*256	512*512	1024*1024
Favreau	88s	209s	351s
Bessmeltsev	257s	>2000s	-
Guo(GPU)	0.19s	0.24s	0.4s
Guo(CPU)	0.6s	2.3s	7.6s
Simo-Serra(CPU)	0.08s	0.7s	1.6s
Simo-Serra(GPU)	0.06s	0.3s	0.5s
Thinning	0.1s	0.3s	0.6s
Ours	0.9s	1.7s	2.9s



(a) (b) (c) (d) (e) (f)

Figure 16: *Limitations. (a)&(b) Our method fails when the boundaries are not smooth and unable to give correct hints. (c)&(d) Our method occasionally mis-classify very short and tangent-inconsistent skeleton segments as spurious spikes. (e)&(f) Our method is not designed for sketch cleaning, so it cannot extract centerlines that need to be semantically simplified.*

rithms for vectorization of engineering drawings. *Journal of Theoretical and Applied Information Technology 94*, 2 (2016), 265. 2

[GZH*19] GUO Y., ZHANG Z., HAN C., HU W., LI C., WONG T.: Deep line drawing vectorization via line subdivision and topology reconstruction. *Comput. Graph. Forum 38*, 7 (2019), 81–90. 1, 2, 8, 9, 11

[HT06] HILAIRE X., TOMBRE K.: Robust and accurate vectorization of line drawings. *IEEE Trans. Pattern Anal. Mach. Intell. 28*, 6 (2006), 890–904. 1, 2

[JV97] JANSSEN R. D. T., VOSSEPOEL A. M.: Adaptive vectorization of line drawing images. *Comput. Vis. Image Underst. 65*, 1 (1997), 38–56. 1, 2

[KWÖG18] KIM B., WANG O., ÖZTIRELI A. C., GROSS M. H.: Semantic segmentation for line drawing vectorization using neural networks. *Comput. Graph. Forum 37*, 2 (2018), 329–338. 1, 2

[LL06] LECOT G., LÉVY B.: Ardeco: Automatic region detection and conversion. In *Proceedings of the Eurographics Symposium on Rendering Techniques, Nicosia, Cyprus, 2006* (2006), pp. 349–360. 2

[LLS92] LAM L., LEE S., SUEN C. Y.: Thinning methodologies - A comprehensive survey. *IEEE Trans. Pattern Anal. Mach. Intell. 14*, 9 (1992), 869–885. 1, 2

[MSSG*21] MO H., SIMO-SERRA E., GAO C., ZOU C., WANG R.: General virtual sketching framework for vector line art. *ACM Transactions on Graphics (TOG) 40*, 4 (2021), 1–14. 2

[NHS*13] NORIS G., HORNUNG A., SUMNER R. W., SIMMONS M., GROSS M. H.: Topology-driven vectorization of clean line drawings. *ACM Trans. Graph. 32*, 1 (2013), 4:1–4:11. 1, 2

[NS19] NAJGEBAUER P., SCHERER R.: Inertia-based fast vectorization of line drawings. *Comput. Graph. Forum 38*, 7 (2019), 203–213. 1, 2

[OBW*08] ORZAN A., BOUSSEAU A., WINNEMÖLLER H., BARLA P., THOLLOT J., SALESIN D.: Diffusion curves: a vector representation for smooth-shaded images. *ACM Trans. Graph. 27*, 3 (2008), 92. 2

[SBBB20] STANKO T., BESSMELTSEV M., BOMMES D., BOUSSEAU A.: Integer-grid sketch simplification and vectorization. In *Computer graphics forum* (2020), vol. 39, Wiley Online Library, pp. 149–161. 2

[SII18] SIMO-SERRA E., IIZUKA S., ISHIKAWA H.: Real-time data-driven interactive rough sketch inking. *ACM Trans. Graph. 37*, 4 (2018), 98:1–98:14. 1, 2, 8, 9

[YVG20] YAN C., VANDERHAEGHE D., GINGOLD Y.: A benchmark for rough sketch cleanup. *ACM Transactions on Graphics (TOG) 39*, 6 (2020), 1–14. 11

[ZS84] ZHANG T. Y., SUEN C. Y.: A fast parallel algorithm for thinning digital patterns. *Commun. ACM 27*, 3 (1984), 236–239. 1, 2, 3, 5, 8