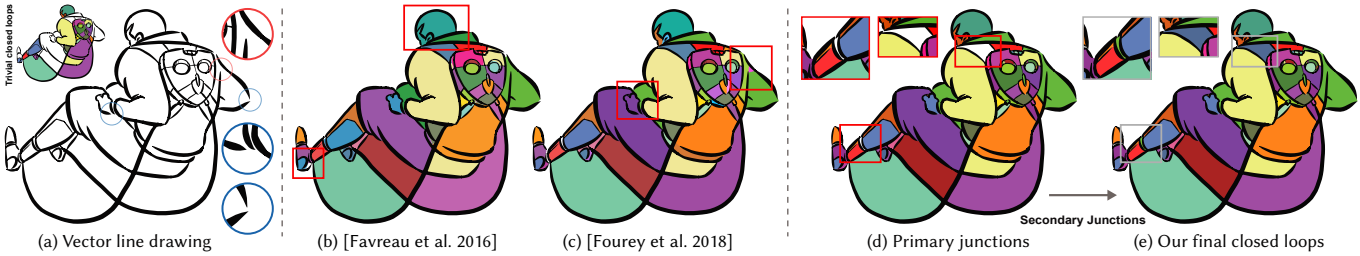
Detecting Viewer-Perceived Intended Vector Sketch Connectivity

JERRY YIN* and CHENXI LIU*, University of British Columbia, Canada
REBECCA LIN, University of British Columbia, Canada
NICHOLAS VINING, University of British Columbia, Canada and NVIDIA, Canada
HELGE RHODIN, University of British Columbia, Canada
ALLA SHEFFER, University of British Columbia, Canada



(a) Vector line drawing (b) [Favreau et al. 2016] (c) [Fourey et al. 2018] (d) Primary junctions (e) Our final closed loops

Fig. 1. (a) Free-hand vector line drawings are often imprecise with strokes intended to intersect stopping short of doing so; loops formed by raw strokes visualized on top left, each closed loop interior colorized with a different color, with the background left white. We successfully extract viewer perceived intended stroke connectivity distinguishing between intended junctions (a, e.g. circled in blue) and intended gaps (a, e.g. circled in red) (e) outperforming prior art (b, c). We arrive at this solution by combining local feature based predictions of the likelihood of pairs of strokes to form intended junctions (d) with global perceptual cues (e). Please zoom in to see image details throughout the paper. Input image ©The "Hero" artist Team under CC BY 4.0.

Many sketch processing applications target precise vector drawings with accurately specified stroke intersections, yet free-form artist drawn sketches are typically inexact: strokes that are *intended* to intersect often stop short of doing so. While human observers easily perceive the artist intended stroke connectivity, manually, or even semi-manually, correcting drawings to generate correctly connected outputs is tedious and highly time consuming. We propose a novel, robust algorithm that extracts viewer-perceived stroke connectivity from inexact free-form vector drawings by leveraging observations about local and global factors that impact human perception of inter-stroke connectivity. We employ the identified local cues to train classifiers that assess the likelihood that pairs of strokes are perceived as forming end-to-end or T- junctions based on local context. We then use these classifiers within an incremental framework that combines classifier provided likelihoods with a more global, contextual and closure-based, analysis. We demonstrate our method on over 95 diversely sourced inputs, and validate it via a series of perceptual studies; participants prefer our outputs over the closest alternative by a factor of 9 to 1.

CCS Concepts: • **Computing methodologies** → **Image manipulation**.

*Both authors contributed equally to this research.

Authors' addresses: Jerry Yin, jerryyin@student.ubc.ca; Chenxi Liu, chenxil@cs.ubc.ca, University of British Columbia, Vancouver, Canada; Rebecca Lin, ryelin@student.ubc.ca, University of British Columbia, Vancouver, Canada; Nicholas Vining, nvining@cs.ubc.ca, University of British Columbia, Vancouver, Canada, NVIDIA, Toronto, Canada; Helge Rhodin, rhodin@cs.ubc.ca, University of British Columbia, Vancouver, Canada; Alla Sheffer, sheffa@cs.ubc.ca, University of British Columbia, Vancouver, Canada.

0730-0301/2022/7-ART1 $15.00
https://doi.org/10.1145/3528223.3530097

ACM Reference Format:
Jerry Yin, Chenxi Liu, Rebecca Lin, Nicholas Vining, Helge Rhodin, and Alla Sheffer. 2022. Detecting Viewer-Perceived Intended Vector Sketch Connectivity. *ACM Trans. Graph.* 41, 4, Article 1 (July 2022), 10 pages. https://doi.org/10.1145/3528223.3530097

1 INTRODUCTION

Free-form line drawings are ubiquitous, both as an art form and as inputs to different computer applications. Thanks to the broad availability of touch pen and stylus devices, such drawings, especially those that artists intend to process later with different algorithmic tools, are increasingly recorded and stored in vector format [Blender 2022; Eitz et al. 2012; Gryaditskaya et al. 2020; Van Mossel et al. 2021]. Drawing processing applications typically require inputs with precisely identified stroke intersections (Fig. 1e). Unfortunately, artist drawings are inherently imprecise [Johnson et al. 2009], and routinely contain *unfinished* strokes that artists intend to intersect other strokes, but that stop short of doing so (Fig. 1a) [Parakkat et al. 2021; Sýkora et al. 2009; Yan et al. 2020; Yang et al. 2018]. While human observers easily mentally complete unfinished strokes [Kanizsa 1979; Sýkora et al. 2009] and identify the intersections they are *intended* to form, extracting this intended connectivity algorithmically remains an open challenge (Sec. 2). We propose a new perception-driven algorithm for identifying intended intersections that produces outputs well aligned with human perception and significantly outperforms the state of the art.

Locating intended junctions algorithmically is highly challenging, as little research exists on the cues that humans employ when mentally separating intended intersections from intended *gaps*. Observations of manual annotations of intended junctions and intentionally dangling stroke *endpoints* (Fig. 2), suggest that viewers leverage both local and contextual cues when making such decisions (e.g.

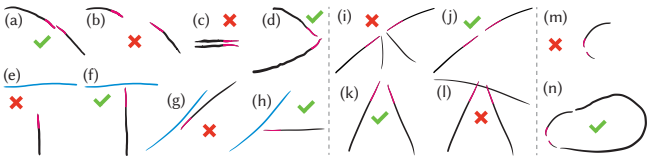
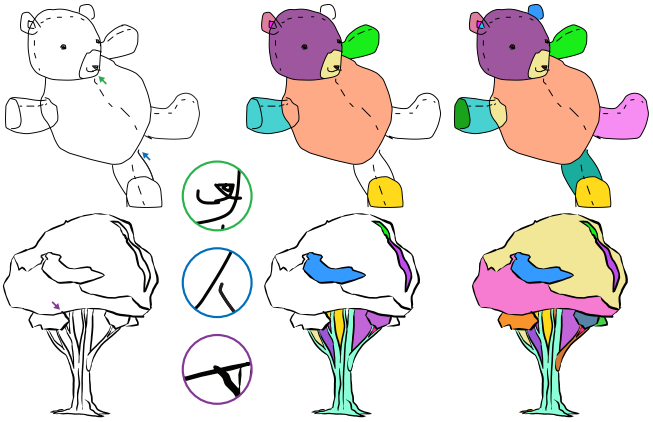


Fig. 2. Human observers employ local and global cues to determine if a dangling endpoint (red) is intentional, or is intended to be part of a junction. As highlighed in (ab) and (ef), distance is a major factor in distinguishing between intended junctions (af) and intended gaps (be). Different tangent directions can impact the perception of junction intent for endpoint (cd) or endpoint and stroke pairs (gh) at the same distance from one another. (i-n) The presence of other strokes can change the perception of whether strokes do or do not form junctions.

the relation between the strokes in Fig. 2m is not evident, but these same strokes are seen as clearly forming an intended end-to-end junction given the extra stroke in Fig. 2n). We use a combination of perception literature review, observation of artist drawings, and manually annotated junctions to identify the factors that determine whether humans perceive dangling stroke endpoints as parts of intended junctions or not (Sec. 3). We hypothesize that these decisions are impacted by both local and global cues, and thus depend both on the geometry of the strokes in question (Fig. 2a-h) , their immediate surroundings (Fig. 2i-l) and on the more global drawing context (Fig. 2mn).

A possible approach for addressing perception motivated tasks is to learn the desired outcomes from human annotated data. Applying this approach in our context raises two conflicting challenges. First, the global nature of the human decisions noted above means that a purely learning-based method would require a very large body of fully annotated sketches to adequately perform the task at hand. At the same time, typical drawings contain many dozens of strokes, and manually separating all intended junctions from intended gaps in a single drawing takes 20 minutes or more on even moderately complex sketches using interfaces optimized for this task (Sec 6); manually annotating large collections of drawings is thus impractical. We overcome these difficulties by developing a hybrid approach which combines data-driven and perception-driven components, and allows us to robustly compute outputs well aligned with human perception from just 31 drawings with partial manual annotations.

We leverage the collected annotations to design two local classifiers. Our first classifier uses the local geometry and context of a pair of dangling endpoints to predict how likely they are to form an end-to-end junction. Our second classifier uses similarly local information to predict the likelihood that a dangling endpoint and a stroke form a T-junction. Both classifiers utilize geometric features we expect to strongly correlate with human perception of junctions, and are trained on our collected manual annotations. Our classifiers achieve an accuracy of 99% in leave-one-drawing-out cross-validation. We embed these classifiers in an incremental decision making process that combines purely local considerations with global properties. It first performs basic pairwise classification across all pairs of valence two end-end and simple T-junction candidates, identifying *primary* junctions (Fig. 1d); it then leverages these decisions to form high probability, more complex, *secondary* junctions, and finally uses all previously made decisions to analyze global context and identify junction candidates whose likelihood of forming junctions is strongly boosted by global perceptual cues (Fig. 1e).



(a) Raw vector line drawing (b) Trivial closed loops (c) Our result

Fig. 3. Typical free-hand drawings (a) contain jaggy, fragmented, and overdrawn strokes (pointed and circled in green), unintentionally dangling endpoints (pointed and circled in blue) and strokes that extend past their intended end-junctions (pointed and circled in purple). Directly extracting closed stroke loops from such drawings (b) produces heavily undersegmented outputs. By identifying unintentionally dangling endpoints and forming intended junctions we form loops consistent with viewer expectations. Top input image ©Mathias Eitz, James Hays and Marc Alexa under CC BY 4.0. Bottom input image ©The "Hero" artist Team under CC BY 4.0.

We design and test our method to operate on free-hand artist drawings which exhibit a range of inaccuracies and drawing artifacts (Fig. 3). We evaluate our method on 95 diversely sourced inputs, and validate it via comparisons to manual annotations and a perceptual study comparing our outputs against prior art. In our annotation comparison our method achieves comparable accuracy to human annotators (92% versus 94%). Comparative study participants preferred our results by a factor of 9 to 1 over the best performing competitor (Sec. 7). These advancements are made possible by leveraging novel insights about human perception of intended junctions and converting those into an actionable robust algorithm for distinguishing intended intersections form intended gaps. Our code and data are at https://www.cs.ubc.ca/labs/imager/tr/2022/SketchConnectivity/.

2 RELATED WORK

Artistic Practices. Artist drawings are inherently imprecise [Johnson et al. 2009] and contain a range of inaccuracies. In particular, artists routinely leave *unintended gaps* between strokes they intend to intersect [Parakkat et al. 2021; Sýkora et al. 2009; Yan et al. 2020; Yang et al. 2018] and continue to do so even when explicitly asked to draw as precisely as possible [Yan et al. 2020]. A potential explanation for this behavior is that artists simply do not notice the errors they introduce, since the human visual system naturally connects weak edges [Kanizsa 1979].

Sketch Processing. A diverse range of tools assist users in processing hand drawn sketches [Adobe Inc. 2021; Blender 2022], facilitating tasks such as colorization, editing, or sketch-based modeling.

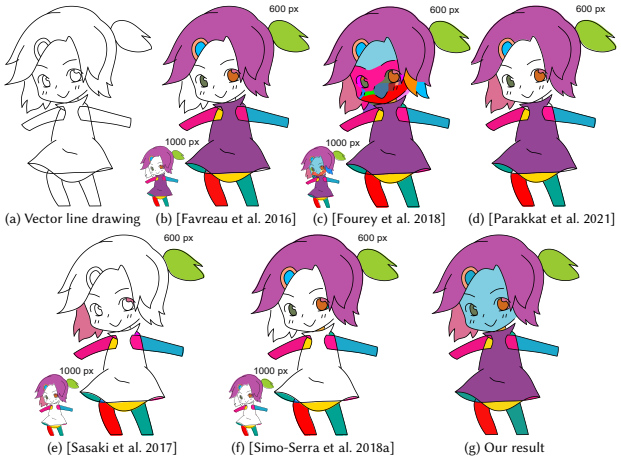


Fig. 4. Rasterizing vector sketches and then applying the methods of [Favreau et al. 2016] (b), [Fourey et al. 2018] (c), [Parakkat et al. 2021] (d), [Sasaki et al. 2017] (e), and [Simo-Serra et al. 2018a] (f) to compute closed stroke loops produces sub-par outputs with both unintended junctions (e.g. Fourey et al. [2018] over-segments character's face) and unresolved dangling endpoints (e.g. none of [Favreau et al. 2016; Parakkat et al. 2021; Sasaki et al. 2017; Simo-Serra et al. 2018a] separates character's face from the background). Our outputs (g) correctly identify both intended junctions and intended dangling endpoints. We show both high and low resolutions (600px and 1000 px) for (b, c, e, f); and the authors' automatically selected resolution (600px) for (d). Light gray spots in the output of [Parakkat et al. 2021] correspond to pixels unassigned by their method. Input image ©Jiang et al. [2020].

The majority of these tools are designed to operate on clean vector format inputs, and assume that the input stroke connectivity reflects artist intent [Shao et al. 2012; Xu et al. 2014; Yan et al. 2020]. Our work facilitates conversion of raw artist sketches into clean drawings ready to use by standard tools (Fig. 3).

Vector Sketch Consolidation and Beautification. Vector sketch consolidation methods [Barla et al. 2005; Liu et al. 2018, 2015; Orbay and Kara 2011] take oversketched vector line drawings as input and fit single curves to groups of strokes that viewers perceive as jointly depicting such individual curves. Beautification methods assist artists in creating more clean and aesthetic drawings [Baran et al. 2010; Cheema et al. 2012; Fišer et al. 2016; Igarashi et al. 1997; Murugappan et al. 2009; Pavlidis and Van Wyk 1985].

Most such methods either do not seek to recover the artist intended stroke connectivity, or use simple distance or trapped-ball radius-based thresholding to connect nearby strokes [Liu et al. 2018, 2015]. As pointed out by Company et al. [2019] and demonstrated in Fig. 2 these criteria are not sufficient to robustly predict intended connectivity. Some beautification methods [Cheema et al. 2012; Fišer et al. 2016; Igarashi et al. 1997; Murugappan et al. 2009] support an interactive gap closure process, and have the user decide which small gaps are intended and which are not. Our work targets the setup where artists first create their drawings using their software of choice and then seek to clean them after the fact. It complements the methods above in its focus on robustly detecting intended connectivity in consolidated or overdrawing-free inputs.

Sketch Vectorization. Vectorization methods convert raster sketches into vector form and typically do not attempt to recover intended stroke connectivity [Bessmeltsev and Solomon 2019; Chen et al. 2018, 2015; Donati et al. 2019; Guo et al. 2019; Kim et al. 2018; Mo et al. 2021; Noris et al. 2013; Puhachov et al. 2021; Stanko et al. 2020]. Several methods aim to simultaneously vectorize and consolidate the inputs, and to identify and close unintended gaps along artist intended stroke loops [Favreau et al. 2016; Parakkat et al. 2021; Zhang et al. 2009]. They detect such gaps using a mixture of methods described next.

Closing Unintended Gaps. Existing methods for gap closure are designed to be used in a semi-automatic setup and can be roughly grouped into two categories: methods that first collect user input, then use this input to apply the core algorithm; and methods that first perform automatic gap closure and then have the user improve this initial result interactively.

Sketching systems such as [Adobe Inc. 2021; Asente et al. 2007; Gangnet et al. 1994] belong to the first category, and support closing gaps between groups of dangling endpoints or dangling endpoints and their closest strokes that are smaller than a user specified threshold. Other methods in this category focus on identifying stroke loops that segment raster drawings into artist-intended regions [Noris et al. 2012; Qu et al. 2006; Sýkora et al. 2009]; these methods are designed to be used in a fully interactive setting where each step of the method is guided by human input. As discussed by [Parakkat et al. 2020] such methods tend to be very sensitive to properties of the initial inputs, requiring trial-and-error to produce a desired outcome. Our method detects unintended gaps fully automatically and does not require user input as a starting point.

The second category of methods attempts to first detect unintended gaps automatically and resort to user input only when necessary. Gryaditskaya et al. [2020] close gaps in design drawings using a distance threshold defined as a function of stroke width. While we use distance normalized by stroke width as one of our classifier features, our experiments and prior research (e.g. [Company et al. 2019; Parakkat et al. 2021]) suggest that proximity based features alone are far from sufficient to distinguish between intended and unintended gaps (Fig. 2).

Sasaki et al. [2017] close unintended gaps in raster drawings using learned in-painting. Simo-Serra et al. [2018a,b, 2016] implicitly close unintended gaps in raster sketches by learning the relationships between raw and clean drawings. As Fig. 4 shows, both types of methods often fail to distinguish between intended and unintended gaps.

Methods for detection of intended end-to-end junctions in wireframe drawings of polyhedra [Company et al. 2019; Wang and Yu 2009; Wang et al. 2020] leverage the fact that by construction these drawings contain only straight lines and do not contain intentionally dangling endpoints. We address a much more general setup where drawings frequently contain intentionally dangling endpoints and where strokes are often not straight and can form both end-to-end and T-junctions. Our more general method can be applied as-is to drawings of polyhedra (see Fig. 1 in our supplemental document).

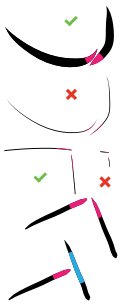
Several methods [Favreau et al. 2016; Zhang et al. 2009] use a combination of trapped-ball initialization and subsequent diffusion to

locate closed regions in raster line drawings. While fully automatic, the authors acknowledge that when using their default parameters the method may often fail to produce results aligned with user expectations (Figs. 1, 4). Many methods for detecting such closed regions operate in a semi-interactive mode [Fourey et al. 2018; Parakkat et al. 2021, 2020; Zhang et al. 2009] but support a fully automatic workflow as well. In all cases, the authors acknowledge that these methods tend to be sensitive to properties of the initial inputs and using them without manual intervention is likely to lead to notable artifacts (Fig. 4). We leverage information provided by vector data to address a more general problem of locating and closing unintended gaps, including both gaps along region boundaries and between strokes internal to such regions (in the inset, the blue strokes delineate an intended region, while the black strokes are internal to the region). Comparing results automatically generated by these methods on rasterized vector data and our results on the original vector input (Figs. 1, 4 and Sec. 7) demonstrates that our outputs are significantly better aligned with human expectations.

Jiang et al. [2021] close gaps in clean vector drawings traced using a custom interface. They operate under the assumption that the vast majority of dangling endpoints are unintended, leading their method to often close gaps that viewers perceive as intentional (Fig. 13 in their paper). Thus, like prior works, they propose a semi-manual interface that enables users to correct such undesirable connections. We aim to process raw artist drawings present in the wild and seek a much higher degree of automation than all prior approaches. Similar to methods in the latter category, our method can be integrated into a workflow where on challenging inputs users first perform automatic gap closure and then interactively correct any remaining issues (Sec. 7).

3 PERCEPTION OF INTENDED SKETCH CONNECTIVITY

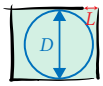
The goal of our algorithm is to identify intended junctions and gaps as perceived by human observers. Like prior work [Shao et al. 2012; Xu et al. 2014] we operate under the assumption that artists aim for their drawings to be well understood, and that viewer understanding of the drawings is in general consistent with artist intent. As in many similar problem settings, one of the big challenges in detecting the intended connectivity is that the exact mechanism observers use to mentally perform this task is unknown. At the same time, our review of related literature points to a number of cues observers are likely to use when identifying intended junctions (Fig. 2).



Stroke-Pair Properties. Prior research [Company et al. 2019; Fourey et al. 2018; Parakkat et al. 2021, 2020] convincingly demonstrates that *distances* between potentially connected strokes play a large role in the perception of intended junctions. Notably, the perception of inter-stroke distance is relative rather than absolute: wider strokes are perceived as closer to one another than thin strokes located at the same distance, and same distance longer strokes are seen as closer than shorter ones. Prior research [Company et al. 2019] further suggests that stroke *directions* at the evaluated potential junction locations serve as a similarly strong cue (e.g. the distance between the pairs of strokes in in Fig. 2cd is identical but due to different endpoint directions we view the strokes in Fig. 2d as intended to connect, and the ones in Fig. 2c as not). Lastly we note that strokes are perceived as more likely to form an end to end junction if their nearest points are at or near their endpoints and a T-junction if the endpoint of one is close to the *middle* of the other; notably this distinction suggests that the type of intended junction formed by the strokes depends on the *relative location of the projection* of the endpoints of one stroke onto the other.

Local Context. We note that perception of intended junctions is impacted not just by the geometry of the participating strokes but by both local and global context. Specifically the presence of other strokes in the immediate vicinity of the assessed pair can impact the perception of junctions (Fig. 2i-l), providing *local context*. In particular the presence of *nearby intersections* between the assessed and other strokes impacts the perception of whether an endpoint is dangling or not, and thus the expectation of its stroke being part of any additional junctions.

Closure. Lastly, and critically, we note that the *closure* principle of Gestalt psychology [Koffka 1955; Wagemans et al. 2012] is highly relevant when analyzing perceived sketch connectivity, as it suggest that viewers are highly likely to mentally close gaps between strokes if doing so results in formation of closed loops, or regions. Liu et al. [2015] utilize this principle for sketch cleanup, merging together overdrawn strokes perceived to bound the same region. They suggest that to evaluate whether a sequence of strokes (or stroke segments) is forming a perceived loop one can consider the ratio between the diameter D of the biggest circle inscribed inside the loop and the length of the largest gap L between consecutive strokes in the sequence,



$$R_C = \frac{D}{L}. \tag{1}$$

The larger this *gap ratio* is the more likely viewers are to perceive the strokes as forming a loop. We observe that conversely, when the ratio is sufficiently small, viewers are unlikely to perceive strokes as forming a loop even if other cues suggest otherwise. In particular our analysis of manually annotated sketches (Sec. 7) suggests that viewers do not mentally close gaps between strokes when doing so results in loops with the ratio R_C being below 1. We refer to this property as *minimal cycle ratio*. This property is connected to topological persistence in the curve and surface reconstruction literature [Dey 2006; Sadri and Singh 2014].

4 ALGORITHM

Our method takes as input raw, free-hand sketches collected in the wild (Sec. 7). We process drawings with both constant and variable width strokes. We pre-process these drawings removing hooks, merging overdrawn strokes, and detecting all *trivial* stroke intersections (locations where pairs of strokes overlap) as discussed in Sup. Sec. 2.2. The output of our method is a set of intended junctions between pairs of locations on these strokes, where these strokes do not overlap a priori in the drawings. We specifically focus on junctions formed by connecting stroke endpoints with either

Fig. 5. Method Overview. Given a vector line drawing (a), we first detect trivial stroke-wise intersections forming closed stroke loops (b, right). We then identify likely end-to-end (red) and T- (blue) junctions (b, left zoom-ins). With these pairs and their predictions, we constructs primary junctions, supporting arbitrary valence (c, see left zoom-ins for examples). We proceed to identify *secondary* T-junctions formed by the remaining dangling endpoints and composite strokes (d, see left zoom-ins for example connections). In the final closure integrated step, we close remaining undesirable gaps by jointly evaluating classifier predictions and gap ratios along the boundaries of potential cycles (e, see the zoom-in for a connection classified as marginally negative in our primary step and accepted in this step). Input image ©The "Hero" artist Team under CC BY 4.0.

other endpoints (end-to-end junctions) or with locations along other strokes (T-junctions). Our last closure-aware step (Sec. 6) considers both junctions involving endpoints and those involving pairs of nearest mid-stroke locations on adjacent strokes.

We identify these intended junctions by leveraging a combination of the perceptual cues listed above and ground-truth data. While access to ground truth data is highly beneficial for producing results consistent with human perception, a core challenge we face is data scarcity: as noted in Sec. 1 annotating intended junctions is a time consuming and mentally non-trivial task. Typical artist drawings (e.g. Fig. 1) have over a hundred strokes ; and take 20 minutes or more to annotate. Thus we require an approach capable of correctly detecting intended intersections using limited data.

We achieve these goals using a method that relies on a combination of four key elements.

Pairwise Junction Classifiers. We utilize the collected annotations to train two classifiers, one for predicting how likely a pair of stroke endpoints is to form an end-to-end junction and one for predicting how likely an endpoint and a stroke are to form a T-junction (Sec. 5). Our classifiers utilize a compact set of features encoding the properties of the evaluated pairs and their local context.

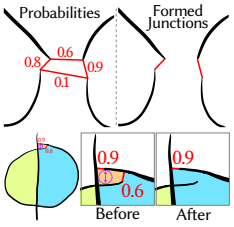
Filters. We utilize the perceptual cues above to narrow down the set of endpoint and endpoint-stroke pairs that the classifiers are applied to by automatically discarding *impossible pairs*, i.e. ones that humans are virtually guaranteed to *not* see as forming intended junctions. This filtering allows us to dramatically reduce the number of negative (perceived as not forming a junction) training examples we need to collect (naive classifier would otherwise need to evaluate all pairs of strokes and endpoints in a drawing against one another). It also allows us to reduce the number of features that the classifiers operate on and thus further reduce the amount of negative and positive training data they require to achieve robust performance. Our pre-filtering considers both the properties and the local context of the classified pairs.

Intended High-Valence Junctions. Freehand sketches can contain junctions with arbitrary high valence; including both complex end-to-end junctions and ones connecting multiple endpoints to a shared T-junction point (see inset). Notably, while many drawings contain such junctions, they each contain only a handful of them. This sparsity means that while in theory one could train separate classifiers for different high-valence junction topologies, collecting enough data to train such classifiers would require annotating a very large number of drawings. We overcome this challenge by developing a junction processing workflow where we successfully utilize the pairwise junction classifiers to predict the likelihood of higher valence junctions (Sec. 4.0.1, 4.0.2).

Closure-Aware Classification. While the perception of closure plays an important role in human perception of stroke connectivity, closure evaluation is inherently global in that it involves considering multiple strokes and multiple potential intended junctions at once. As such it is hard to account for using only pairwise or other purely local classifiers; learning closure based choices directly would therefore require a dramatic increase in the amount of training data. We address closure in a data efficient manner by using a delayed decision process where in the first rounds of our computation, closure, and specifically, minimal cycle ratio, is only used as a hard negative constraint preventing us from forming junctions that would create loops violating this constraint. Once all local decisions are made we efficiently compute potential loops induced by these decisions and revisit all prior negative classification decisions, incorporating closure as a positive cue (Sec. 4.0.3).

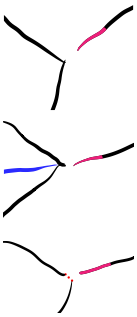
Armed with these tools we formulate our intended intersection detection as the following gradual decision process (Fig. 5).

4.0.1 Primary Junctions Classification. Our primary junctions classification step first identifies pairs of endpoints or endpoints and strokes that have the potential to form intended junctions and then uses our two classifiers to compute the probability of each pair forming such a junction (Sec. 5). It uses this information to form likely binary and high-valence junctions when doing so does not violate our minimal cycle ratio constraint. Specifically, when the classifier deems two or more pairs containing one or more same endpoints as each being likely to form an intended junction we form the subset of these interconnected junctions that maximizes the joint probability across them (see inset, top). After making these decisions, we evaluate the gap ratio (Eq. 1) for each newly formed cycle; if a cycle violates the minimal cycle ratio constraint ($R_C < 1$), we break it by removing the lowest probability junction along its perimeter (see inset, bottom).



4.0.2 Secondary Junctions Classification. Our *secondary* classification step addresses the formation of intended high-valence junctions. Specifically, it analyzes remaining dangling endpoints and

nearby previously identified junctions and evaluated whether the dangling strokes should be extended to connect to these junctions. The previous junctions considered include both trivial junctions (inset, two top rows) and previously detected intended junctions (inset, bottom). Our evaluation leverages two observations: we first note that two strokes whose endpoints are part of a common junction can be conceptually seen as a single *composite* stroke (pairs of black strokes on the left of each inset), thus for our purposes we can evaluate if a dangling endpoint should be connected to such a junction utilizing our T-junction classifier and applying it to the dangling endpoints and such *composite* strokes. While at a high-valence junctions one can potentially composite multiple strokes (inset, middle), given a dangling endpoint, humans are likely to only consider the composite stroke (black in this inset) formed by the two strokes that are immediately next to the assessed endpoint (with respect to a circular ordering around the junction location) and ignore those occluded by these two (blue in the inset). This observation again suggests that we can evaluate if a dangling endpoint should be connected to such a high-valence junction utilizing our T-junction classifier by applying it to the dangling endpoint and temporary *composite* strokes formed by the strokes (or portions of strokes) that are part of the junction and are immediately next to the endpoint. We process all classifier decisions utilizing such composite strokes using a process identical to the one in Sec. 4.0.1.

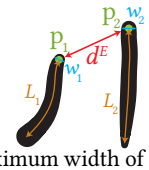


4.0.3 Global Closure-Aware Classification. Our final, *closure integrated step* forms potential stroke cycles containing remaining dangling endpoints or pairs of nearest points on adjacent strokes and uses a combination of previously computed classifier probabilities at these endpoints and the gap ratios along these cycles to determine whether the remaining gaps along these cycles should be closed or remain open (Sec. 6).

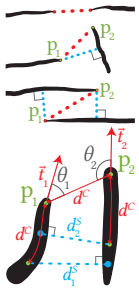
5 JUNCTION CLASSIFIER

End-end classifier. Given endpoints p_1 and p_2 on strokes S_1 and S_2, we construct four sets of features, motivated by the perceptual cues in Sec. 3. In the description below, each asymmetric feature, reported for p_1, is computed for both endpoints; the minimum and maximum over both values are used to make the classifier commutative by design.

Distance. We use three features to encode the distance between the stroke endpoints. We measure the distance between the stroke envelopes $d^E = \|p_1 - p_2\| - \frac{1}{2}(w_1 + w_2)$ (see inset). We convert this distance into three viewer-perceived scale-invariant features by normalizing it by (1) the mean of the maximum width of each stroke (W_1, W_2), and by the (2) min and (3) max of the stroke lengths (L_1, L_2).



Directions. We use four types of features to encode the interaction between the directions of the two strokes at the endpoints of interest overcoming drawing inaccuracies. We codify the *type* of the junction, characterizing it as belonging to one of the three categories in the inset, by counting whether two, one, or zero of the endpoints project onto the opposing stroke endpoints. We include the angles θ_1, θ_2 between the stroke tangents and the line connecting the two endpoints (see inset); we compute the tangent $\vec{t}_1$ by stepping back from the endpoint along the stroke by the distance $d^C = \|p_1 - p_2\|$. We also include two ratios that are even less susceptible to noise than the tangents: the *step-away ratio*, measured as the distance d_2^S from the step-away point to S_2 divided by d^C, and the *projection ratio*, the distance from p_1 to S_2 divided by d^C.



Relative Location. We encode the distance between the projection of p_1 on S_2 and its closest endpoint along S_2, normalized by L_2.

Local Context. We encode local context as the distance from p_1 to the closest stroke in the drawing other than S_1 and S_2, normalized by gap size d^C.

T-junction classifier. We use similar features for the T-junction classifier, modifying them to account for its asymmetric nature. To this end, we compute distance measures between endpoint p_1 on S_1 and the closest point p_2 on stroke S_2, the directional features are computed only from the endpoint to the other stroke, and we skip the projection ratio since the projection and p_2 are the same for T-junctions. When computing local context, points that are occluded by S_2 are excluded. We incorporate larger context for T-junction decisions than for end-end ones by using an additional *endpoint density* feature, defined as a function of the distances from this endpoint to all endpoints,

$$b = \sum_{p_e \in \{\text{endpoints}\}} e^{-\frac{1}{2}\left(\frac{1}{\sigma}\frac{\|p_1 - p_e\|}{w_e}\right)^2}.$$

The contribution of each endpoint is a Gaussian of the distance to it normalized by w_e, the average of the widths along its stroke. We use $\sigma = 1$, which ensures that endpoints fall to a negligible contribution when they are 3 stroke widths away.

Training. We implement both the endpoint-endpoint and T-junction classifiers as random forests that are trained on either endpoint or endpoint and stroke pairs that are labelled as intended or unintended junctions.

6 ALGORITHM DETAILS

Filtering Junction Candidates. The number of possible junctions grows quadratically with the number of endpoints, and the vast majority of end-end or end-stroke pairs are not intended to connect. Our filtering avoids a massive imbalance between positive and negative examples during training, reduces the runtime cost of our method, and drastically reduces the number of negative annotations we need to collect for training. Following the observation about the impact of local context, for each endpoint, we only consider connections with the closest three endpoints and strokes, respectively. We further filter those that are too far, lack a line of sight, or have diverging directions (inset). We similarly prevent end-to-end junctions between parallel

strokes. The closest-three filter reduces the number of pairs from quadratic to linear, and the later filters further reduce the number of pairs passed to the classifier by a factor of 2 or more on a typical input. See our supplementary materials for additional details.

Global Closure-Aware Classification. At this point of the process, we expect the vast majority of intended intersections to be appropriately classified, enabling us to compute meaningful stroke cycles containing only a handful of unintended gaps.

We first locate all pairs of closest points on immediately adjacent strokes; for each pair we compute the cycles formed by connecting the pair; we mark the pair as forming an intended junction if the distance between its points is smaller than the length of the largest previously closed gap along these cycles and $R_C > 20$ for both cycles.

Our closure-aware step then leverages the probabilities computed by the classifier and combines those with the evaluation of the closure ratio R_C (Eq. 1) to identify pairs of strokes that are likely to form junctions once closure is accounted for. Our analysis of training data suggests that the closure ratio can be viewed as a boost signal, increasing the likelihood of a gap being intended by approximately a linear factor. In other words, given a ground truth probability P of the end-points of a gap forming a junction, the gap ratio boosts this probability to $P' = P + C(R_C - 1)$. In practice, the probabilities provided by our classifier are approximate. We thus use a conservative step-function approximation of the formula above with $C = 0.025$. Using a step size of 0.05 and starting at $P = 0.45$ for each gap with classifier probability of P and above we close the gap if $P' > 0.5$.

Specifically, at each step value of P we order all junction candidate pairs with probability P or more; for each pair we compute the cycles formed by connecting the pair. We mark the junction as intended if $P' > 0.5$ for both cycles. We repeat this process considering two candidate pairs at once. We have not encountered cases for which evaluating three or more pairs was necessary.

Random Forest. Our random forest classifiers have 100 trees each. We limit the maximum depth to 10 for the endpoint-endpoint classifier and 12 for the T-junction classifier. We use the scikit-learn library [2011] for all training.

Training Set. We trained our method using 31 sparsely annotated drawings. In assembling this set we aimed for a diverse set of sources spanning different styles, content and levels of expertise. Our training set consists of 13 drawings from the Blender Art Gallery [2021], 5 from *Quick, Draw!* [2018], 2 design sketches from OpenSketch [2019] and 11 original drawings. All drawings are provided in the supplementary material. In total, we have 290 positive endpoint-endpoint examples, 1778 negative endpoint-endpoint examples, 460 positive T-junction examples, and 2817 negative T-junction examples. The annotatins were created using an in-house interface that incrementally colorizes regions based on user annotations.

Closing gaps. We visualize closed gaps by using shortest straight lines connecting participating endpoints and strokes. To close gaps in a geometrically-pleasing manner, one can use the method of [Jiang et al. 2021] or modify a curve-fitting method (e.g. [Van Mossel et al. 2021]) to enforce junctions.

7 RESULTS AND VALIDATION

We tested our method on 95 previously unseen inputs from a diverse set of sources spanning different styles, content and levels of expertise. To this end we include 30 professional drawings of characters and organic shapes created using the Blender Grease Pencil Tool and provided in the Blender Art Gallery [2021]; rough amateur sketches, including five each from [Ha and Eck 2018], [Eitz et al. 2012] and Ge et al. [2020], and 10 each from [Sangkloy et al. 2016] and [Qi et al. 2021]. We also included 11 drawings of polyhedra from [Company et al. 2019], and one input from Jiang et al. [2021]. In addition to these raw inputs, we applied our methods to pre-consolidated sketches: 8 from StrokeStrip [2021] and 3 from OpenSketch (using the ground truth consolidations for the former, and the StrokeAggregator [2018] consolidations for the latter), as well as algorithmic vectorizations of 7 raster drawings from Parakkat et al. [2021, 2020]. Representative examples are shown in the paper; the rest are included in the supplementary.

We validate the key aspects of our method in a number of ways: we evaluate our classifiers using leave-one-drawing out cross-validation, evaluate our methods final classification decisions by comparing them against manual annotation, and compare our method to algorithmic alternatives via a comparative user study.

Classifier Cross-Validation. We evaluate our classifiers using a round-robin cross-validation process where we leave one drawing out, train on the remaining drawings, and then test on the ground truth labels in the left-out drawing. Under cross-validation, we achieve an accuracy of 99%, a precision of 97%, and a recall of 96%. As expected, visual analysis of the few failure cases points to global cues discussed above as the main reason for failure.

Perceptual Validation. While assessing artist intent requires direct access to the artist, sketch processing literature [Gryaditskaya et al. 2020; Shao et al. 2012; Xu et al. 2014] strongly suggests that artist intent is well correlated with viewer perception. We thus focus our evaluations on compariosn against viewer expectations. In addition to the ground truth labels used for training the classifiers, we collected manual annotations of 91 potential end-to-end and T-junctions across 10 drawings from the test set, with each potential junction annotated by 8 non-expert study participants. Across all junctions, participants agreed with the majority response 94% of the time, and were evenly split on 1 junction. The *final* classification decisions made by our algorithm agree with the majority response 92% of the time, nearly identical to the human agreement level—the most we can expect from an algorithm. This agreement number suggests that for a typical drawing with approximately 100 dangling tips after running our method users are unlikely to require more than 2-3 corrections to obtain an output consistent with their expectations.

Comparisons Against Prior Art. We compare our method against prior interactive and automatic methods. When comparing against the former, we seek to assess the time it takes a user to generate a desired drawing connectivity using ours versus alternative approached. We focus this comparison on the LazyBrush [Sýkora et al. 2009] method, as it has been implemented in a commercial software package [Krita 2021]. On a representative input (Fig. 7a) it took an

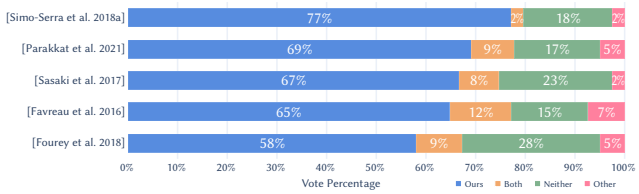


Fig. 6. Study summary: participants preferred our method over all alternatives by a factor of 9 to 1 or more.

(a) Vector line drawing (b) User input for [Sýkora et al. 2009] (c) Our result and corrections (d) Final colorization

Fig. 7. Comparison against interactive region detection. Given an input (a), the interactive LazyBrush tool [Sýkora et al. 2009] required 31 minutes (70 strokes, one erased) (b); starting from our automatically computed output (c) users required 2 minutes (7 corrections) to obtain the same final output (d). Input image ©The "Hero" artist Team under CC BY 4.0.

artist 31 minutes to achieve the desired output (Fig. 7c). To achieve this result, they used 70 scribble of different widths (including one erased in the process), Fig. 7b. Starting from our automatically generated output (Fig. 7c) the user required 2 minutes to generate the same output, using 7 corrections.

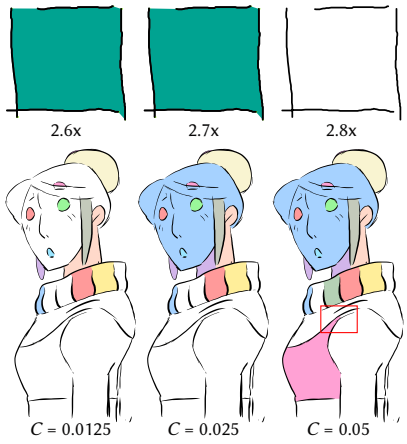


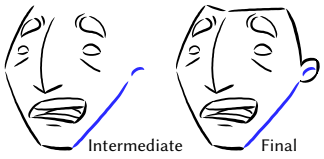
Fig. 8. Impact of increasing the top left gap size (top) and the closure factor C (bottom) during our final, global closure-aware classification step. Top input image ©Company et al. [2019]. Bottom input image ©The "Hero" artist Team under CC BY 4.0.

We also compare our method to five state of the art automatic gap closure methods, whose code we were able to access [Favreau et al. 2016; Fourey et al. 2018; Parakkat et al. 2021; Sasaki et al. 2017; Simo-Serra et al. 2018a]. As discussed in Sec. 2 these methods detect closed cycles in raster data. To compare against these methods we rasterize our inputs and run them on the raster data, We use the output colorizations of [Favreau et al. 2016; Fourey et al. 2018; Parakkat et al. 2021]. We colorize the outputs of [Sasaki et al. 2017; Simo-Serra et al. 2018a] using flood-fill. We render the vector strokes on top of both colorizations and use a consistent raster resolution for all drawings (600px for shorter image side); as shown in the supplementary material, this resolution produces the best result on average across these methods.

Fig. 4 and Fig. 9 compare our results against those generated by these methods. A complete set of comparisons is provided in the supplementary material. To compare the perceptual accuracy of our method against these prior approaches we conducted a comparative study (Sup. Sec. 3). Participants were shown an input line drawing, and colorizations of this drawing obtained using our method and an alternative method. They were asked to "envision which strokes in [the input] drawings are intended by the artist to form closed loops," to "Identify the differences between the two [shown] colorings (ignore small color bleedings)," and then answer "Which of the images on the bottom. (B) or (C), better corresponds to the partition you envisioned?" Overall we had collected answers to 135 comparative questions, 6 answers per question. The study findings are summarized in Fig. 6. Participant debriefings suggest that viewers looked for both under- and over-segmentations when evaluating alternative colorizations. When both colorizations were imperfect, they preferred the colorization with fewer or less visually disruptive errors. In a comparison with the best alternative [Favreau et al. 2016] our method was preferred 65% of the time, and judged equally good 12% of the time; the method of Favreau et al. [2016] was preferred just 8% of the time, and neither result was judged as corresponding to the participant envisioned one 15% of the time. The demonstrated preferences were highly statistically significant ($p < 0.001$ for all methods). These numbers convincingly demonstrate our significant improvement over the state of the art, in the context of detecting intended junctions in vector drawings.

Closure Ablation. We conducted a geometry variation and a parameter variation ablation experiment on the closure cue. Fig. 8 (top) demonstrates how the closing of gaps is robust to the gap size. For this specific input, the closure step continues to connect the gap until the distance becomes 2.8 times larger than the original. Fig. 8 (bottom) shows the impact of changing the value of the closure factor C during our final, global closure-aware classification step (Sec. 6). If C is too low, the gap ratio R_C does not have sufficient influence in this stage, and major regions such as the face are not captured. If C is too high, however, the global step may yield undesirable false positives, such as the front of the tunic in the example figure. This both validates the importance of incorporating the gap ratio during this final stage, and our choice of closure factor. An interesting area for future work could be to learn an adaptive closure factor based on local or global drawing properties.

Incremental Processing. As an alternative to our approach, we also experimented with an incremental workflow based on drawing order. Unfortunately, when presented with an incomplete drawing (inset images ©The "Hero" artist Team under CC BY 4.0) both our method and human observers perceive some intended gaps as unintended; automatically closing such intended gaps mid-drawing is highly disruptive to the artist. Processing complete drawings provides our method with more complete decision-making context, leading to outputs better reflecting artist intent.



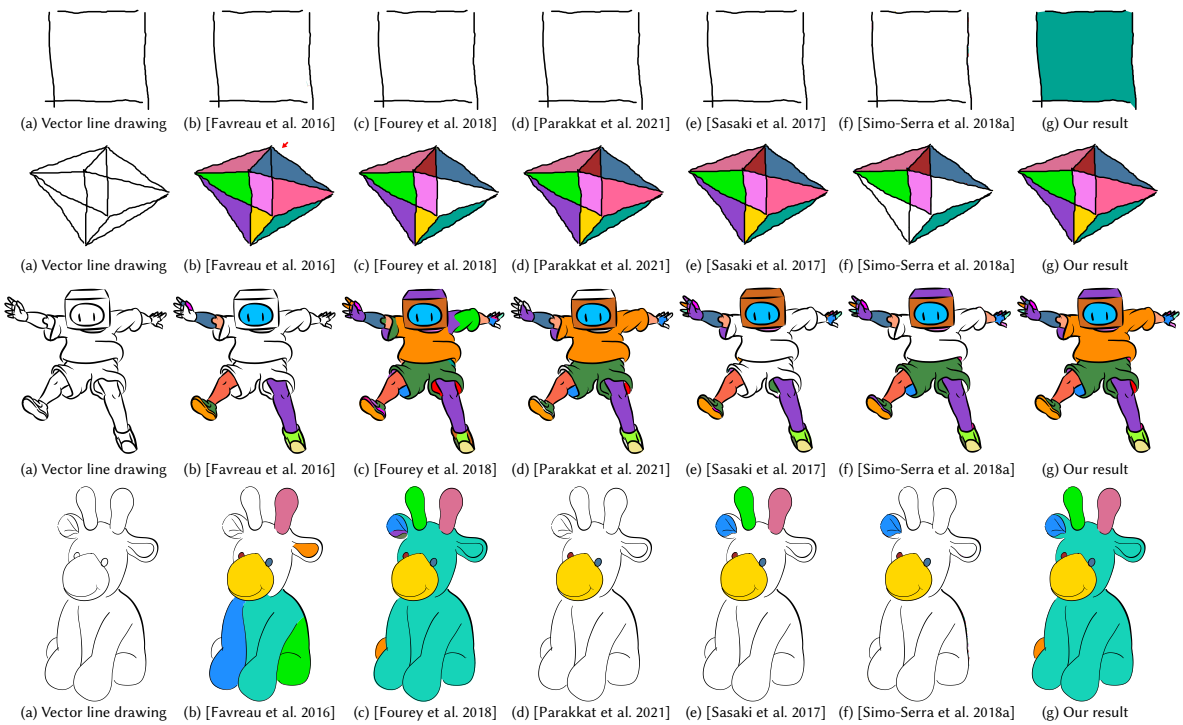


Fig. 9. Additional results and comparisons. Input images from top to bottom ©Company et al. [2019]; ©Lien-ze Tsao under CC BY 4.0; ©Enrique Rosales.

Junction Statistics. Across the 95 inputs in our test set, our method forms 1584 junctions in total: 638 (40.3%) junctions are binary end-end junctions, 855 (54.0%) junctions are binary T-junctions, and only 91 (5.7%) junctions are high-valence junctions. We note that the sparsity of high-valence junctions (fewer than 1 per input on average) validates our design choice to use the pairwise junction classifiers to predict the likelihood of higher valence junctions.

Junction Formation Counts per Step. Across the 95 inputs in our test set, 1452 (91.7%) junctions are formed during our primary step, 58 (3.7%) are formed during our secondary step, and 74 (4.7%) are formed during our closure-aware step. This confirms that our primary step forms most of the junctions, whereas the secondary and closure-aware steps form fewer, yet visually critical, junctions (as demonstrated in Fig. 1 and Fig. 5.)

8 CONCLUSION

We presented a new method for detecting intended junctions in raw free-hand vector drawings and demonstrated it to significantly outperform the state of the art. At the core of our method are binary classifiers that reliably predict the likelihood of pairs of endpoints or endpoints and strokes forming intended junctions. They are trained on features motivated by observations about human perception of such junctions. We address high-valence junctions by applying the T-junction classifier to composite strokes assembled based on observations of human perception and improve classification accuracy at test time by accounting for global closure cues. Our method successfully processes drawings in the wild and has been validated across inputs with different accuracy levels (e.g. Fig. 1 vs Fig. 3).

Limitations. While as demonstrated above our method significantly outperforms all state-of-the-art alternatives, a non-negligible number of comparative study participants selected the "neither" option when faced with our and alternative inputs. This suggests that while we significantly advance the state of the art, additional effort is necessary to detect intended junctions in free-hand drawings fully automatically. Furthermore, while our pre-processing is capable of detecting and consolidating sketches which contain some amount of overdrawing, our core method is designed to operate on inputs with no or minimal overdrawing. Using our method on sketchy inputs with large amount of overdrawing or hatching requires a more robust consolidation pre-process; while the methods reviewed in Sec. 2 for this task can often be used for such pre-processing, robust consolidation remains an open research problem.

ACKNOWLEDGMENTS

We thank the "Hero" artist team and Lien-ze Tsao for sharing their artworks (shown with modifications in this work), the study participants, Pedro Company, Jiang Jie and other previous work authors for sharing their data, and the reviewers for their helpful suggestions. This work was supported by NSERC.

REFERENCES

Adobe Inc. 2021. *Adobe Illustrator.* https://adobe.com/products/illustrator
Paul Asente, Mike Schuster, and Teri Pettit. 2007. Dynamic Planar Map Illustration. *ACM Trans. Graph.* 26, 3 (2007), 10 pages.
Ilya Baran, Jaakko Lehtinen, and Jovan Popović. 2010. Sketching Clothoid Splines Using Shortest Paths. *Comput. Graph. Forum* 29, 2 (2010), 655–664.
Pascal Barla, Joëlle Thollot, and François X. Sillion. 2005. Geometric Clustering for Line Drawing Simplification. In *ACM SIGGRAPH 2005 Sketches (SIGGRAPH '05).* Association for Computing Machinery, 96–es.
Mikhail Bessmeltsev and Justin Solomon. 2019. Vectorization of Line Drawings via Polyvector Fields. *ACM Trans. Graph.* 38, 1 (Jan. 2019), 9:1–9:12.
Blender. 2021. Blender Cloud. https://cloud.blender.org/p/gallery/5b642e25bf419c1042056fc6
Blender. 2022. Grease Pencil. https://www.blender.org/features/grease-pencil/

Salman Cheema, Sumit Gulwani, and Joseph LaViola. 2012. QuickDraw: Improving Drawing Experience for Geometric Diagrams. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. Association for Computing Machinery, 1037–1064.

Jiazhou Chen, Mengqi Du, Xujia Qin, and Yongwei Miao. 2018. An Improved Topology Extraction Approach for Vectorization of Sketchy Line Drawings. *Vis Comput* 34, 12 (Dec. 2018), 1633–1644.

JiaZhou Chen, Qi Lei, YongWei Miao, and QunSheng Peng. 2015. Vectorization of Line Drawing Image Based on Junction Analysis. *Sci. China Inf. Sci.* 58, 7 (July 2015), 1–14.

Pedro Company, Raquel Plumed, Peter A. C. Varley, and Jorge D. Camba. 2019. Algorithmic Perception of Vertices in Sketched Drawings of Polyhedral Shapes. *ACM Trans. Appl. Percept.* 16, 3 (Aug. 2019), 18:1–18:19.

Tamal K Dey. 2006. *Curve and surface reconstruction: algorithms with mathematical analysis*. Vol. 23. Cambridge University Press.

Luca Donati, Simone Cesano, and Andrea Prati. 2019. A Complete Hand-Drawn Sketch Vectorization Framework. *Multimed Tools Appl* 78, 14 (July 2019), 19083–19113.

Mathias Eitz, James Hays, and Marc Alexa. 2012. How Do Humans Sketch Objects? *ACM Trans. Graph.* 31, 4 (July 2012), 44:1–44:10.

Jean-Dominique Favreau, Florent Lafarge, and Adrien Bousseau. 2016. Fidelity vs. Simplicity: A Global Approach to Line Drawing Vectorization. *ACM Trans. Graph.* 35, 4 (July 2016), 120:1–120:10.

Jakub Fišer, Paul Asente, Stephen Schiller, and Daniel Sýkora. 2016. Advanced Drawing Beautification with ShipShape. *Computers & Graphics* 56 (May 2016), 46–58.

Sébastien Fourey, David Tschumperlé, and David Revoy. 2018. *A Fast and Efficient Semi-Guided Algorithm for Flat Coloring Line-Arts*. The Eurographics Association.

Michel Gangnet, Jean-Manuel Thong, and Jean-Daniel Fekete. 1994. Automatic Gap Closing for Freehand Drawing. In *ACM SIGGRAPH 94 Technical Sketch*.

Songwei Ge, Vedanuj Goswami, Larry Zitnick, and Devi Parikh. 2020. Creative Sketch Generation. In *International Conference on Learning Representations*.

Yulia Gryaditskaya, Felix Hähnlein, Chenxi Liu, Alla Sheffer, and Adrien Bousseau. 2020. Lifting Freehand Concept Sketches into 3D. *ACM Trans. Graph.* 39, 6 (Nov. 2020), 167:1–167:16.

Yulia Gryaditskaya, Mark Sypesteyn, Jan Willem Hoftijzer, Sylvia Pont, Frédo Durand, and Adrien Bousseau. 2019. OpenSketch: A Richly-Annotated Dataset of Product Design Sketches. *ACM Trans. Graph.* 38, 6 (Nov. 2019), 232:1–232:16.

Yi Guo, Zhuming Zhang, Chu Han, Wenbo Hu, Chengze Li, and Tien-Tsin Wong. 2019. Deep Line Drawing Vectorization via Line Subdivision and Topology Reconstruction. *Computer Graphics Forum* 38, 7 (Oct. 2019), 81–90.

David Ha and Douglas Eck. 2018. A Neural Representation of Sketch Drawings. In *International Conference on Learning Representations*. https://openreview.net/forum?id=Hy6GHpkCW

Takeo Igarashi, Satoshi Matsuoka, Sachiko Kawachiya, and Hidehiko Tanaka. 1997. Interactive Beautification: A Technique for Rapid Geometric Design. In *Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology (UIST '97)*. Association for Computing Machinery, 105–114.

Jie Jiang, Hock Soon Seah, and Hong Ze Liew. 2021. Handling Gaps for Vector Graphics Coloring. *Vis Comput* 37, 9 (Sept. 2021), 2473–2484.

Jie Jiang, Hock Soon Seah, Hong Ze Liew, and Quan Chen. 2020. Challenges in Designing and Implementing a Vector-Based 2D Animation System. In *The Digital Gaming Handbook*. CRC Press.

Gabe Johnson, Mark D. Gross, Jason Hong, and Ellen Yi-Luen Do. 2009. Computational Support for Sketching in Design: A Review. *Found. Trends Hum.-Comput. Interact.* 2, 1 (2009), 1–93.

Gaetano Kanizsa. 1979. *Organization in Vision: Essays on Gestalt Perception*. Praeger.

Byungsoo Kim, Oliver Wang, A. Cengiz Öztireli, and Markus Gross. 2018. Semantic Segmentation for Line Drawing Vectorization Using Neural Networks. *Comput. Graph. Forum* 37, 2 (2018), 329–338.

K. Koffka. 1955. *Principles of Gestalt Psychology*. Routledge & K. Paul.

Krita. 2021. Krita. https://krita.org/

Chenxi Liu, Enrique Rosales, and Alla Sheffer. 2018. StrokeAggregator: Consolidating Raw Sketches into Artist-Intended Curve Drawings. *ACM Trans. Graph.* 37, 4 (July 2018), 97:1–97:15.

Xueting Liu, Tien-Tsin Wong, and Pheng-Ann Heng. 2015. Closure-Aware Sketch Simplification. *ACM Trans. Graph.* 34, 6 (Oct. 2015), 168:1–168:10.

Haoran Mo, Edgar Simo-Serra, Chengying Gao, Changqing Zou, and Ruomei Wang. 2021. General Virtual Sketching Framework for Vector Line Art. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH 2021)* 40, 4 (2021), 51:1–51:14.

S. Murugappan, S. Sellamani, and K. Ramani. 2009. Towards Beautification of Freehand Sketches Using Suggestions. In *Proceedings of the 6th Eurographics Symposium on Sketch-Based Interfaces and Modeling (SBIM '09)*. Association for Computing Machinery, 69–76.

Gioacchino Noris, Alexander Hornung, Robert W. Sumner, Maryann Simmons, and Markus Gross. 2013. Topology-Driven Vectorization of Clean Line Drawings. *ACM Trans. Graph.* 32, 1 (Feb. 2013), 4:1–4:11.

G. Noris, D. Sýkora, A. Shamir, S. Coros, B. Whited, M. Simmons, A. Hornung, M. Gross, and R. Sumner. 2012. Smart Scribbles for Sketch Segmentation. *Comput. Graph. Forum* 31, 8 (Dec. 2012), 2516–2527.

G. Orbay and L. B. Kara. 2011. Beautification of Design Sketches Using Trainable Stroke Clustering and Curve Fitting. *IEEE Trans. Vis. Comput. Graph.* 17, 5 (May 2011), 694–708.

Amal Dev Parakkat, Marie-Paule Cani, and Karan Singh. 2021. Color by Numbers: Interactive Structuring and Vectorization of Sketch Imagery. In *CHI '21: Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*.

Amal Dev Parakkat, Prudhviraj Madipally, Hari Hara Gowtham, and Marie-Paule Cani. 2020. *Interactive Flat Coloring of Minimalist Neat Sketches*. The Eurographics Association.

Theo Pavlidis and Christopher J. Van Wyk. 1985. An Automatic Beautifier for Drawings and Illustrations. *SIGGRAPH Comput. Graph.* 19, 3 (July 1985), 225–234.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-Learn: Machine Learning in Python. *J. Mach. Learn. Res.* 12 (2011), 2825–2830.

Ivan Puhachov, William Neveu, Edward Chien, and Mikhail Bessmeltsev. 2021. Keypoint-Driven Line Drawing Vectorization via PolyVector Flow. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)* 40, 6 (Dec. 2021).

Anran Qi, Yulia Gryaditskaya, Jifei Song, Yongxin Yang, Yonggang Qi, Timothy M. Hospedales, Tao Xiang, and Yi-Zhe Song. 2021. Toward Fine-Grained Sketch-Based 3D Shape Retrieval. *IEEE Trans. Image Process.* 30 (2021), 8595–8606.

Yingge Qu, Tien-Tsin Wong, and Pheng-Ann Heng. 2006. Manga Colorization. *ACM Trans. Graph.* 25, 3 (July 2006), 1214–1220.

Bardia Sadri and Karan Singh. 2014. Flow-Complex-Based Shape Reconstruction from 3D Curves. *ACM Trans. Graph.* 33, 2, Article 20 (apr 2014), 15 pages.

Patsorn Sangkloy, Nathan Burnell, Cusuh Ham, and James Hays. 2016. The Sketchy Database: Learning to Retrieve Badly Drawn Bunnies. *ACM Trans. Graph.* 35, 4 (July 2016), 119:1–119:12.

Kazuma Sasaki, Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. 2017. Joint Gap Detection and Inpainting of Line Drawings. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 5768–5776.

Cloud Shao, Adrien Bousseau, Alla Sheffer, and Karan Singh. 2012. CrossShade: shading concept sketches using cross-section curves. *ACM Trans. Graph.* 31, 4 (2012), 45:1–45:11.

Edgar Simo-Serra, Satoshi Iizuka, and Hiroshi Ishikawa. 2018a. Mastering Sketching: Adversarial Augmentation for Structured Prediction. *ACM Trans. Graph.* 37, 1 (Jan. 2018), 11:1–11:13.

Edgar Simo-Serra, Satoshi Iizuka, and Hiroshi Ishikawa. 2018b. Real-Time Data-Driven Interactive Rough Sketch Inking. *ACM Trans. Graph.* 37, 4 (July 2018), 98:1–98:14. https://doi.org/10.1145/3197517.3201370

Edgar Simo-Serra, Satoshi Iizuka, Kazuma Sasaki, and Hiroshi Ishikawa. 2016. Learning to Simplify: Fully Convolutional Networks for Rough Sketch Cleanup. *ACM Trans. Graph.* 35, 4 (July 2016), 121:1–121:11.

Tibor Stanko, Mikhail Bessmeltsev, David Bommes, and Adrien Bousseau. 2020. Integer-Grid Sketch Simplification and Vectorization. *Computer Graphics Forum (Proc. SGP)* 39, 5 (7 2020).

Daniel Sýkora, John Dingliana, and Steven Collins. 2009. LazyBrush: Flexible Painting Tool for Hand-Drawn Cartoons. *Comput. Graph. Forum* 28, 2 (2009), 599–608.

Dave Pagurek Van Mossel, Chenxi Liu, Nicholas Vining, Mikhail Bessmeltsev, and Alla Sheffer. 2021. StrokeStrip: Joint Parameterization and Fitting of Stroke Clusters. *ACM Trans. Graph.* 40, 4 (July 2021), 50:1–50:18.

J. Wagemans, J. H. Elder, M. Kubovy, S. E. Palmer, M. A. Peterson, M. Singh, and R von der Heydt. 2012. A Century of Gestalt Psychology in Visual Perception I. Perceptual Grouping and Figure-Ground Organization. *Psychological Bulletin* 138, 6 (2012), 1172–1217.

Shuxia Wang and Sui-huai Yu. 2009. Endpoint fusing of freehand 3D object sketch with Hidden-part-draw. *2009 IEEE 10th International Conference on Computer-Aided Industrial Design & Conceptual Design* (2009), 586–590.

Shuxia Wang, Qian Zhang, Shouxia Wang, Xiaoke Jing, and Mantun Gao. 2020. Endpoint Fusing Method of Online Freehand-Sketched Polyhedrons. *Vis Comput* 36, 2 (Feb. 2020), 291–303.

Baoxuan Xu, William Chang, Alla Sheffer, Adrien Bousseau, James McCrae, and Karan Singh. 2014. True2Form: 3D Curve Networks from 2D Sketches via Selective Regularization. *Transactions on Graphics (Proc. SIGGRAPH 2014)* 33, 4 (2014).

Chuan Yan, David Vanderhaeghe, and Yotam Gingold. 2020. A Benchmark for Rough Sketch Cleanup. *ACM Trans. Graph.* 39, 6 (Nov. 2020).

Wenwu Yang, Hock-Soon Seah, Quan Chen, Hong-Ze Liew, and Daniel Sýkora. 2018. FTP-SC: Fuzzy Topology Preserving Stroke Correspondence. *Comput. Graph. Forum* 37, 8 (2018), 125–135.

Song-Hai Zhang, Tao Chen, Yi-Fei Zhang, Shi-Min Hu, and Ralph R. Martin. 2009. Vectorizing Cartoon Animations. *IEEE Trans. Vis. Comput. Graph.* 15, 4 (July 2009), 618–629.