Automatic structuring of organic shapes from a single drawing

Even Entem, Amal Dev Parakkat, Loic Barthe, Ramanathan Muthuganapathy, Marie-Paule Cani

Automatic structuring of organic shapes from a single drawing

Even Entem[a], Amal Dev Parakkat[b,d], Loïc Barthe[c], Ramanathan Muthuganapathy[b], Marie-Paule Cani[d]

[a] *IRIT, Université de Toulouse, LJK, Université de Grenoble-Alpes, CNRS and Inria*
[b] *AGCL, Indian Institute of Technology Madras*
[c] *IRIT, Université de Toulouse, CNRS*
[d] *LIX, Ecole Polytechnique, CNRS*

1. Introduction

Contour drawings are commonly used for shape depiction. They are both easy to create and easy to interpret for a human and thus it makes them a convenient and expressive solution for visual communication. They are found in children's books, advertisements, technical books, and more. In contrast, these drawings are difficult for a computer to interpret. They usually depict silhouette curves and internal contours as well as expressive strokes, and they may represent fully visible, self-occluding, or locally hidden regions.

Many contour drawings are directly authored in vector graphics applications or are easily converted to a compatible representation using vectorization tools. Automatically decomposing them into distinct and simple structural parts, layered in depth (as in Figure 1 (b)), permits users to edit and manipulate the drawings intuitively by rescaling, moving, rotating, copying, and pasting parts without the need for intricate manual modifications and corrections which would otherwise be required for such operations.

In this paper, we present an automated geometric method for extracting apparent structure and depth layers from clean contour line-drawings. We assume that the input drawing is intended to represent an organic shape, i.e., any free-form 3D solid with smooth connections between its 3D structural parts. The extracted depth-ordered structure is similar to the collection of blobs that artists sometimes use to temporarily define the construction lines and volumes of the shape they want to depict (see results of a web image search with the terms "tutorial drawing construction animals").We record additional information, namely, where these volumes blend together and where contours should be erased. This information can then be used for both current and nearby views editing to achieve new poses such as in Figure 1 (c). Although view-dependency may prevent the structure from being complete relative to the actual structure of the 3D depicted shape, we claim that it is still a useful reference for editing the current drawing to depict nearby postures or viewpoints.

The input drawing may be composed of silhouette curves as well as different categories of internal and external curves. They include internal open contours connected to silhouette curves, e.g., the contours of the feather groups in Figure 1 (a). Regions in the drawing that are demarcated by silhouette curves may also include a number of internal regions depicting parts, possibly lying on top of one another, such as the eye of the swan in Figure 1. Highly ambiguous curves, such as disconnected internal curves and connected external open curves, are considered in our work as decorative curves. We also detect and discard internal elements that fail to define their own silhouettes (see Section 3.2).

Our three contributions towards solving structuring and layering problems for drawings are as follows:

- We describe a simple and efficient method for the aesthetic closing of part contours. This method provides a consistent solution when disconnected endpoints are not explicitly defined in the input drawing (Section 4).
- We introduce the *radial variation metric* (RVM), a novel part-aware metric for complex 2D drawings, inspired by the *volumetric shape image* used for shape segmentation of 3D models [1]. Its variation along the medial axis of parts in a drawing enables the identification of salient connections between parts (Section 5).
- We describe a recursive algorithm enabling the successive identification of parts in a complex sketch and their assignment to depth-layers (Section 6). The key insight lies in processing the possible junction zones between the identified parts in a specific order based on the types of contours involved. This enables us to handle cases of multiple connected internal contours forming a tree-like structure, as can be observed for the swan wing in Figure 1.

The structure and layering information we obtain can be used to represent and edit the input sketch in current or nearby views. We also demonstrate the automatic conversion of the sketch into a Vector Graphics Complex (VGC), a structure that eases the computational and manual editing of vector drawings, and that readily allows for manipulation, editing, and animation (Section 7).

In Section 8, we discusses two of the possible applications of our system, namely the creation of cardboard articulated puppets and of 3D models from a sketch.

2. Related work

Recovering the structural parts of 3D objects in 2D vector contour drawings is a long standing and complex problem. This is due to the lack of information required for the unambiguous and automated shape understanding of most drawings. For instance, the understanding of the main features in a drawing is often based on contextualized interpretations.

Given that a drawing is only composed of lines, several approaches have been proposed to identify which visual mechanisms are used to help interpret the drawn lines in terms of self-consistent shapes and contours [2].

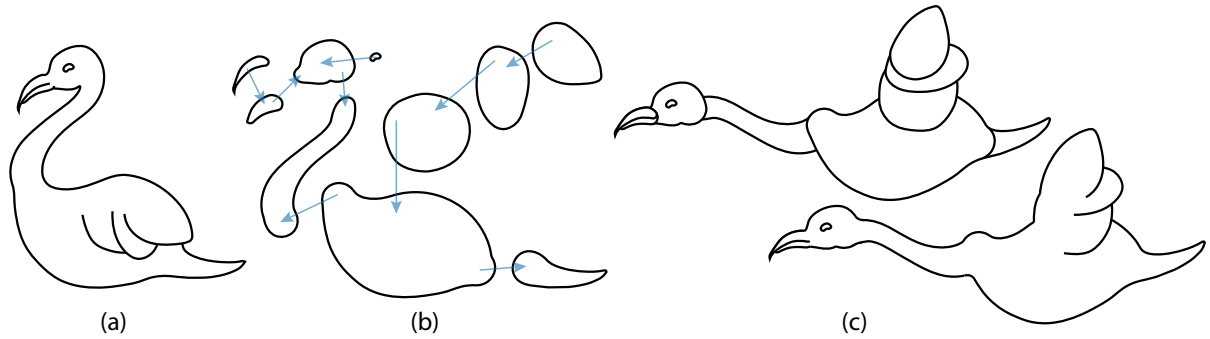

Fig. 1. An illustration of the steps of our automatic structuring and layering system. (a) The input drawing. (b) Structure analysis and part-completion estimate the constituent parts and their layering. The arrows represent the partial depth ordering ("over" relation). (c) Manipulation of the depth ordering as desired (top), followed by the union of these elements composed with the original internal contours to produce a new drawing (bottom).

Several methods address specific aspects of this complex process and techniques have been developed to evaluate them on well-defined data sets. Alternatively, more practical approaches aim at providing actionable interpretation methods by leveraging knowledge about particular object classes and relying on Gestalt principles.

One such Gestalt principle, the principle of closure – how our visual system tends to perceive the missing parts of curves or contours – has been used by algorithms to complete hidden and subjective contours [3, 4, 5]. Many solutions rely on plausible, visually appealing curves such as Minimum Energy Curves (MEC) [6] that maximize the curve smoothness, and Minimum Variation Curves (MVC) [7] that generate fair solutions. Our technique uses a variation of the latter with the aim of efficiently generating aesthetic curves.

Our approach is also inspired by the fact that the human visual system tends to segment a complex shape into simpler parts. Shape segmentation problems have been tackled for a very long time. Pioneering work relied on the branches of the skeletal representation of 2D shapes for inferring segmentation [8]. Observing that the quality of the correspondence between branches and depicted shape parts drops as the complexity of the object rises, subsequent work rather made use of maxima of negative curvature on the contour to identify part boundaries and focused on trying to disambiguate pairing between such boundary elements [9, 10]. Shape segmentation then became a classical problem for both 2D and 3D shapes. The surveys by Yang et al. [11] and Shamir [12] present a detailed study of these methods Fully accepted general solutions do not yet exist, and segmentation remains an area of active research even for the 2D case [13, 14, 15]. In parallel, interactive sketch segmentation and/or completion methods based on human input were proposed to fill in drawings [16], to select layers for shape manipulation [17], to segment sketchy drawings [18] or to simplify them [19, 20].

In this work, we build on, or draw inspiration from, a number of the processing steps introduced for 2D [21] and 3D [1] shape segmentation. However, our goal is to automatically segment complex sketches with not only contours but with internal silhouettes as well. Indeed, internal silhouettes are great perceptual conveyors of shape, as shown by their extensive use for expressive depictions of 3D models and high-reliefs [22, 23, 24]. Therefore, taking them into account is essential for being able to segment a larger variety of drawings.

This new goal brings us to prior work in the area of sketch-based modeling, where a number of methods relied on the analysis of complex sketches to infer a 3D shape or a 2.5D high relief from a single sketch, e.g., [25, 26, 27, 28]. Most methods in the area built on *a priori* knowledge (or contextual information) in order to resolve ambiguities - such as requiring exact symmetric shapes [29], being restricted to garments [30] or to side views of animals [31] or requesting 3D information such as a 3D skeleton from the user [32]. Others methods obtained great results by relying on interactive user annotations for helping to infer high-reliefs from photos or drawings [33, 34, 35, 28].

Closer to our goal, interpreting drawings of general organic shapes (ie. smooth 3D solids) depicting all visible silhouettes including cusps was tackled by Karpenko's pioneering work [25] in the context of 3D modeling from a sketch. To this end, drawn lines were represented as networks of *oriented* curves, enabling the authors to identify holes and to propose a reconstruction method for hidden silhouettes indicated by T-junctions. While we build on this work, we chose not to ask the user for the orientation of contours, thus interpreting a two-circle drawing of a torus as two superposed spheres. We focus instead on extending the handling of internal curves beyond cusps, enabling us to handle more complex suggestive contours as well as extra decorative elements.

In summary, we provide the first fully automatic method able to extract structural parts from cartoon drawings of organic shapes. We provide in Section 7 a detailed comparison between our results and those of previous work, by re-using a number of their examples.

Finally, we note that our method outputs a layered structure of parts locally ordered in depth, so as to ease the subsequent editing of the drawing. Different structures and representations have been developed to handle partial depth orderings [36, 37, 38]. In this work we output results in the VGC format [38], and we choose to define a self-consistent global depth ordering of the shape parts. The extracted structure is view-dependent (there is no structure inferred for parts that are completely occluded, for instance), and the rule that treats in-

ternal silhouettes as distinct cases may lead to occasional surprises. For example, the pupil of the cat's eye in Figure 2 is in fact a hole inside the eye, and thus its correct manipulation would require an intersection operator with the eye region to not protrude out of it.

3. Overview

Our method decomposes an input drawing into a set of 'structural parts', layered in depth, and computes the location of hinges enabling the pose and the animation of the drawn model. The decomposition is based on connected internal silhouettes and inner closed contours in the drawings, which provide explicit indicators of part layering; the resulting parts are intended to be meaningful for artists as shape-defining-volume silhouettes, and can be used for editing and posing in nearby views.

This section introduces the terminology we use throughout the paper, defines the assumptions we make on the input drawing, and presents the main features of our algorithm.

3.1. Terminology and assumptions

The input of our method is a vector line-drawing $\mathcal{D}$ defined in the (x, y) plane as a set C of parametric curves that may only intersect at their endpoints. The drawing may be either directly created in this form or obtained from a rasterized drawing using a vectorization algorithm [39] and then cutting curves at all intersections. Parametric curves are also uniformely sampled as polylines for some of the further processing. In the following, we therefore refer to points along the curves in C as *samples*.

As in Smoothsketch [25], our algorithm is designed to handle contour-drawings of smooth, closed shapes, which we refer to as *organic shapes*. However, to be able to handle a larger category of drawings, we also allow them to include specific categories of decorative curves such as those often used in cartoon drawing.

Therefore, our algorithm includes a mechanism for the automatic detection of contour curves within C. Since we are not asking for any additional information (such as contour orientations) from the user, we assume that the depicted shapes have no surface-to-surface contact, have genus 0, and are not self-overlapping (e.g., no animal's tail passing under and behind the body and forming a new background region).

We use the following terminology throughout the paper (see Figure 2):

Contour: A contour is a curve in C that corresponds to a silhouette of the depicted 3D shape, i.e. to points where the normal to the shape is orthogonal to the viewpoint. The contour graph is the planar graph structure formed by the contours and their intersections.

Region: A region $\mathcal{R}_i$ is a 2D connected component of $\mathcal{D}$ delimited by a counterclockwise face cycle in the contour graph.

Part: Structural parts (or parts) are the 2D counterparts of the structural elements of the 3D shape represented by the drawing, as depicted in Figure 1(b). Our goal is to extract them.

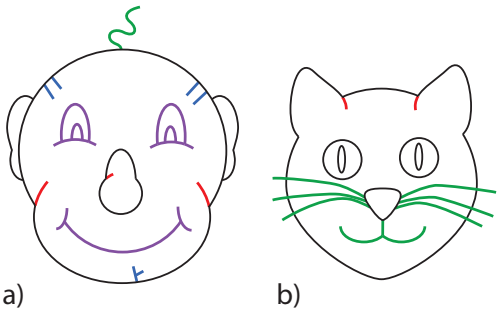

Fig. 2. Stroke classification. Red curves are suggestive contours; green curves are hair; blue and purple curves are part of non-hair decorative elements. The remaining contours, in black, are part of the external silhouettes of shape parts.

Suggestive contour: A suggestive contour is an internal curve within a region $\mathcal{R}_i$ of the drawing, connected with tangent continuity to its external contour (and thus forming a T-junction). Such curves are used in drawings to partially depict the visible contour of a structural part.

Note that this definition is slightly different from the one found in the literature, since our term is in between the concepts of suggestive contours [22] and of cusps [25], to better match what we observed in typical line drawings.

Decorative elements: These include all curves in $\mathcal{D}$ that are not visible contours of the depicted shape, but can instead represent ornamental details, 1D elements such as hair, or strokes used to represent bas-relief carvings. In our case, a curve is considered as a decorative curve if it falls in any of the following categories:

- Inner isolated subgraphs that do not contain external contours when processed as independent input drawings (in purple in Figure 2). Although they might contain or actually be inner contours, such subgraphs are highly ambiguous. We leave interpreting and processing them for future work.
- Inner trees of curves that are connected to the external contour of a region, but without tangent continuity (blue curves in Figure 2)
- Trees of curves located outside of the region they are connected to (green curves in Figure 2, which we call *hair*).

In our method, the decorative curves are identified and ignored in further contour processing, but are kept in the description of the corresponding to-be-segmented part.

3.2. Processing pipeline

Let $\mathcal{G}$ be the contour graph, the planar half-edge graph defined by the curves C that constitute the input drawing $\mathcal{D}$. As in standard planar graph processing, each half-edge corresponds to a given orientation of a curve. If half-edges are part of a closed contour, they are considered to lie respectively in the interior and in the exterior of the corresponding face cycle of the

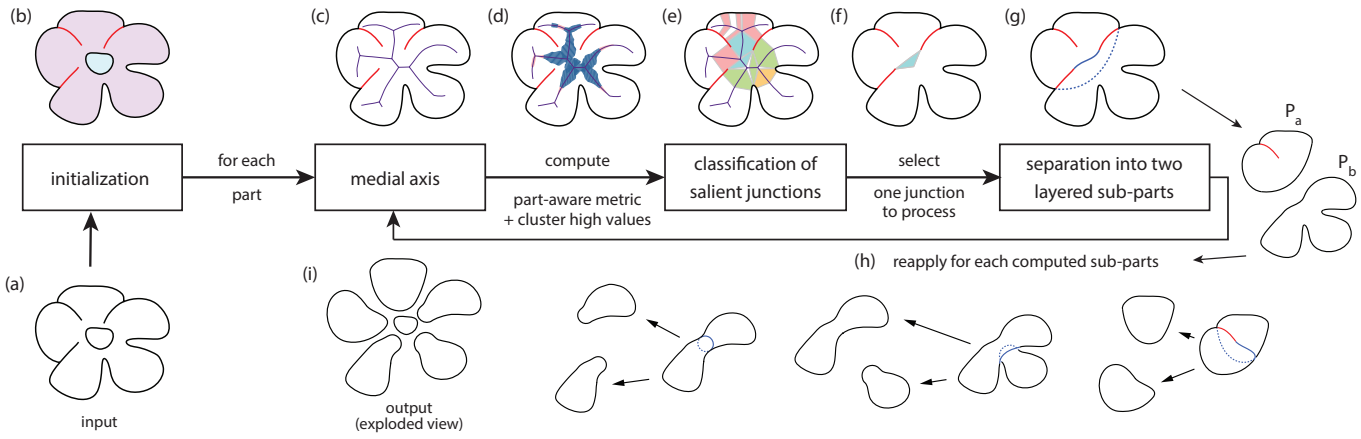

Fig. 3. Processing pipeline: The input is (a) and the output is (h) with partial depth ordering (here depicted in exploded view). Note that our algorithm successfully decomposes the petal even in the absence of a suggestive contour (see the top left petal in (h)). This holds because our decomposition relies on the measurement of salient transition between two different regions and not on the presence of suggestive contours.

graph. In the remainder of this section we use both "edge" and "curve" to denote the edges of $\mathcal{G}$, depending on context.

The goal of our method is to process $\mathcal{G}$ in order to extract and progressively refine the set $\mathcal{P}$ of 2D structural parts of $\mathcal{D}$, as well as the associated partial depth ordering $PO_{\mathcal{P}}$ expressing their relative depths. More precisely, $\mathcal{P}$ is defined as: $\mathcal{P} = \{P_i = (e_i, \mathcal{S}_i, \mathcal{O}_i)\}$, where e_i is the external silhouette contour (as a list of edges) of P_i and where $\mathcal{S}_i$ (respectively $\mathcal{O}_i$) is the subgraph of $\mathcal{G}$ corresponding to the suggestive contours (respectively, the decorative elements) located within P_i, or attached to it.

Our processing pipeline, detailed below, is depicted in Figure 3. Starting with an initial set of silhouette-complete parts extracted at the initialization stage, our algorithm recursively decomposes each part into simpler parts and updates $PO_{\mathcal{P}}$ accordingly. This is done until none of the parts in $\mathcal{P}$ has any suggestive contour left (for all i, $\mathcal{S}_i$ is empty).

Initialization:

We aim at initializing $\mathcal{P}$ to a first set of structural parts, such as the heart of the flower versus the part with all its petals in Figure 3 or the head versus the ears and the nose of the head in Figure 2 (a), and to set $PO_{\mathcal{P}}$ to the corresponding partial depth ordering. This involves completing the contours of partially hidden parts, such as the ears. Although this decomposition and figure completion problem was already solved in the past [5], this was for drawings only depicting silhouette outlines of occluded surfaces. To handle more complex cases of cartoon drawings with decorative lines such as the whiskers in Figure 2 (b), we extend the original method into a four-stage process:

First, if $\mathcal{G}$ contains any vertex v of valence 4 or more (such as vertices where the whiskers of the cat cross the head's contour in Figure 2 (b)), we convert $\mathcal{G}$ into a non-planar graph by dissociating the curves at v, enabling us to further process the whiskers as decorative elements attached to the nose. This is done based on the Gestalt rule of continuity, as follows: For each vertex v of valence 4 or more with adjacent edges in $\mathcal{G}$ that correspond to pairs of tangent-continuous curves, we join each pair of curves at v into a single curve. This disconnects the pairs of curves from each other, enabling them to be attached to different structural parts. A constraint is also set to prevent more than one pair of merged curves from being interpreted as a contour curve in further processing, since this would violate our hypotheses of organic shape depiction (e.g., two overlapping circles do not correspond to any valid contour of organic shape, and are thus interpreted as a single 2D region crossed by a closed decorative curve). After this stage, any connected component of $\mathcal{G}$ that still contains a vertex of valence larger than 3 is considered in its whole as a decorative element, since a set of curves that connect without any tangent continuity cannot include silhouettes of smooth organic shapes.

Then, we process each remaining connected component CC_j of $\mathcal{G}$ to separate contour curves from curves corresponding to the associated suggestive silhouettes or decorative elements.

Since the input drawing is supposed to contain no self-overlapping part, this can be done by simply moving every edge whose half-edges both belong to the same face cycle (ie. lie in the same 2D region) from CC_j to another subgraph $\mathcal{A}_j$, which gathers candidate edges for either $\mathcal{S}_i$ or $\mathcal{O}_i$. Note that this operation may split CC_j into smaller connected components, since the edges moved to $\mathcal{A}_j$ may include bridges between different subgraphs. In this case, CC_j and the corresponding $\mathcal{A}_j$ are split into smaller subgraphs. We decide to which sub-connected component the bridge sub-graph should be associated to by looking at its tangent continuity with the neighboring curves: We insert the bridge into the $\mathcal{A}_k$ set associated with the subgraph CC_k of CC_j to which it has tangent continuity at one of its endpoints (e.g., a suggestive curve partially hidden by the contour of an inner part). If there is tangent continuity at both ends, the decision is taken at random. If there is none, the bridge is considered as a decoration and is attached to the subgraph corresponding to the contour of the region where it lies.

At this stage, each CC_j should only contain contour edges (for instance, the drawing in Figure 2 (a) is split into two connected components, namely (1) the nose and (2) the head plus ears, where only black curves remain). This enables us to use the existing algorithm in [5] to split then into structural parts, by using T-junctions to identify partially hidden parts (such as

the ears) and smoothly complete their contours. Failure cases of this algorithm and other limitations will be discussed in Section 7.

In contrast with previous work, we use our own efficient solution, presented in Section 4, to compute closure curves. All resulting parts $\mathcal{P}_i$ are stored in $\mathcal{P}$ together with their contour edges e_i. The corresponding partial depth ordering information is added to $PO_{\mathcal{P}}$. We also add temporary depth relations with the remaining connected components, depending on the number and order of intersections with each other component to reach the background of the drawing with a line (parts with the same number of intersections being considered to have the same depth level).

In a final stage, the set $\mathcal{S}_i$ of suggestive contours of each part $\mathcal{P}_i$ is extracted from the set $\mathcal{A}_j$ associated to its former connected component CC_j, by selecting curves with smooth T-junctions with contours e_i and that lie within $\mathcal{P}_i$. The other curves in $\mathcal{A}_j$ connected to e_i are classified as decorative elements and added to O_i. Lastly, every curve in other $\mathcal{A}_k$ sets that were initially crossing any of the current contour curves are stored in the set O_k of the part in which it is located since they no longer can be contours.

The set of parts $\mathcal{P}$ is now ready for further decomposition.

Recursive part decomposition:

The core of the algorithm is a procedure that decomposes $\mathcal{P}$ into parts that are themselves decomposed recursively if possible.

This enables us to process complex suggestive contours indicating embedded parts, such as the wing of the swan in Figure 1: The full wing is extracted first and is then recursively split into partially overlapping parts. The recursive loop proceeds as follows:

For each part P in $\mathcal{P}$, we identify the salient potential junction region zones between parts, and iterate from best-to-worst until a valid pair (P_a, P_b) of parts is identified. Missing contours are then inferred for P_a and P_b (see Figure 3 (g)) and the depth ordering relation between them is added to $\mathcal{PO}$. P_a and P_b are added to the list of parts $\mathcal{P}$, enabling us to recursively apply this process until no further decomposition is possible (Figure 3 (h)).

This recursive decomposition method raises three challenges, leading to our three key technical contributions:

(1) The description of a robust method for completing the contours of the extracted parts in a perceptually valid way.

(2) The design of an effective metric for identifying the regions of possible junction between the parts of a 2D shape. In contrast with previous work, the 2D shapes we process may include suggestive contours, which are not constrained to come in even numbers, or to be short cusps.

(3) The definition of an order in which to process the identified alternative solutions for segmentation into parts.

This order is important for extracting consistent parts as it enables us to reuse the same algorithm in a recursive fashion.

Our aesthetic and efficient contour completion is presented in Section 4 and we discuss our new metric for identifying junctions in Section 5. Section 6 details our recursive structuring algorithm. It makes use of our new metric and completion method, with an emphasis on the priority order we set for processing possible junctions.

4. Aesthetic and efficient contour completion

In this section, we present the completion method we use for closing contours of both partially occluded structural parts (initialization step) and of parts extracted during recursive part decomposition. Although not proved to be the best possible perceptual completion method, our solution is simple, efficient, and produces adequate results in practice.

4.1. Scale-Invariant MVC

We first note that perceptually pleasing contour completion is a different problem than completing illusory contours [5]. We seek to find aesthetic curves that are appropriate for the editing and subsequent animation of the drawing, during which hidden or omitted contour segments may become visible. In the literature, both curves minimizing the total curvature (i.e. "smooth" curves, MEC's) and curves minimizing the total variation of curvature (i.e., "fair" curves, MVC's) have been proposed as possible solutions to the problem. We choose to use MVC's because they tend to form more circular arcs; these are particularly well suited to organic shapes.

Let A and B be the two endpoints of the open contour to be connected, along with corresponding unit tangent directions T_A and T_B as shown in Figure 4 (a). Our goal is to generate a perceptually plausible curve between A and B that best matches an inferred silhouette for the resulting part.

We define the curve connecting these two input points as follows. Let B_{AB} be a Bézier cubic curve connecting A and B, defined by the four control points (A, P_1, P_2, B), and whose tangents are aligned along the unit vectors T_A and T_B, i.e., $P_1 = A + c_1 T_A$, and $P_2 = B + c_2 T_B$. We optimize the free parameters c_1 and c_2 in order to minimize a "fairness energy" i.e. a variation of the SIMVC energy originally introduced by Moreton [7]. Let $C(s)$ be a parametric 2D curve with curvature $\kappa(s)$. Then the SIMVC energy [7] is defined as:

$$E_{\text{SIMVC}-Moreton} = \left(\int \mathrm{d}s\right)^3 \int \left(\frac{\mathrm{d}\kappa(s)}{\mathrm{d}s}\right)^2 \mathrm{d}s \tag{1}$$

where $\left(\int \mathrm{d}s\right)^3$ is the product of a regularization factor $\left(\left(\int \mathrm{d}s\right)/\|B-A\|\right)^3$ and a scale-invariance factor $\|B-A\|^3$.

This regularization factor relies on the cube of the scale-relative arc length of the curve. We increase the regularization factor's exponent from 3 to 5 in order to reward slightly shorter curves. This avoids cases where closure curves would slightly jut out from the desired boundaries. We thus propose to minimize the following energy:

$$E_{\text{SIMVC}} = \frac{\left(\int \mathrm{d}s\right)^5}{\|B-A\|^2} \int \left(\frac{\mathrm{d}\kappa(s)}{\mathrm{d}s}\right)^2 \mathrm{d}s \tag{2}$$

As in Equation 1 the first term makes the curves scale-invariant. The use of Bézier curves guarantees that the curve lies in the convex hull of its control points and is therefore well suited to interactive sampling and intersection queries. The choice to optimize its parameters with the SIMVC functional produces a scale-invariant result.

In practice, we use the standard Gauss-Kronrod quadrature [40] to numerically integrate the different integral terms. Powell's method [41] is used for minimizing E_{SIMVC}.

4.2. *Efficient implementation*

Given that the completion curves we compute are invariant to scaling, translation and rotation, there remain only two configuration parameters, namely θ and φ. They are the oriented angles formed by the two tangents with respect to a line between the two points to be connected, as shown in Figure 4 (a).

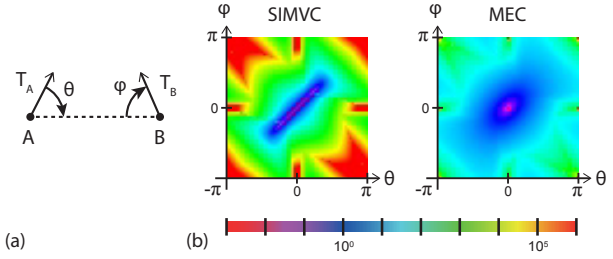

Fig. 4. To optimize the computation of our SIMVC curves, we precompute a table of the energies and associated parameters that the curve can take as a function of the two angles θ and φ defined in (a). We illustrate the sampled function SIMVC energy values and compare them with the MEC energy that minimized total curvature.

The SIMVC energy of our curve defined as a function of these two parameters is continuous and smooth in the sub-space of non self-intersecting curves, as shown in Figure 4 (b). This enables us to precompute a table of the different SIMVC Bézier cubic curves, with their SIMVC energy and parameters c_1 and c_2, as a function of θ and φ. Parameters can then be interpolated, with either bilinear or bicubic interpolation, to provide both an approximated curve and good initial parameters for the final curve computation, leading to an important speedup of the gradient descent with two variables.

In practice, we use Bézier curves for easily detecting invalid contours that occurs when the closures for partially occluded or partially occluding parts protrude outside the union of the related parts. To avoid unintended intersections we rotate tangents, at the points to be connected, inwards and by a small angle, keeping the misalignment of tangents almost imperceptible. During our experiments, we chose a rotation of 2 degrees.

5. Extracting Regions of Possible Junction within a Part

While previous work already addressed the segmentation of 2D shapes into perceptually salient parts, these methods do not tackle the segmentation of shapes carrying extra structural information in the form of suggestive contours. This is the problem we are addressing here.

5.1. *Regions of possible junction*

In the remainder of this paper, we define a **region of possible junction** as a region where the drawing of a part exhibits a perceptual change, enabling it to be divided into two parts in a perceptually consistent way.

We define **junction boundaries** as being the portions of the part contours that delimit such a junction. These portions can either be a point (e.g., the open end of a suggestive contour) or a contour segment, as will be the case when the exact point where the segmentation should occur is unclear.

Among the common approaches to 2D shape segmentation that we review in Section 2, computing junction boundaries as the segments of the contour of maximal negative curvature cannot capture approximate regions such as those indicated by the orange and blue junction boundary in Figure 5 (a), since the blue one is not a curvature maximum and thus would not be detected. Similarly, segmentation methods that directly make use of the branches of a skeletal representation such as the Medial-Axis Transform to identify parts, also fail in a number of cases, as shown in Figure 5 (b,c). Fortunately, such correspondence between a Medial-Axis branch and a structural part remains generally true for foreground structural parts delimited by suggestive contours, such as the middle part in Figure 5 (b). The solution we describe below therefore builds on skeletal representations, while being based on a new metric.

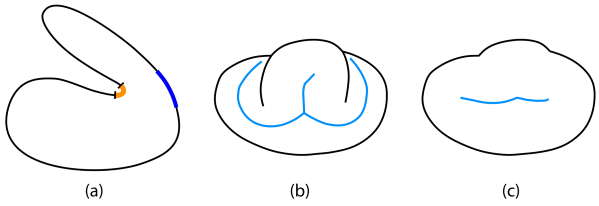

Fig. 5. (a) A maximal negative curvature (in orange) suggests the presence of a junction boundary on the contour but its counter-part boundary (in blue) does not exhibit any maximum, which avoids the detection of a boundary. (b) S-skeleton of a part (in blue), i.e. Medial-Axis considering internal silhouettes: the lower structural part is not captured, while the middle part at the top is. (c) Processed Medial-Axis of the external contour of (b), where no branch is associated with the top structural part.

5.2. *Radial Variation Metric*

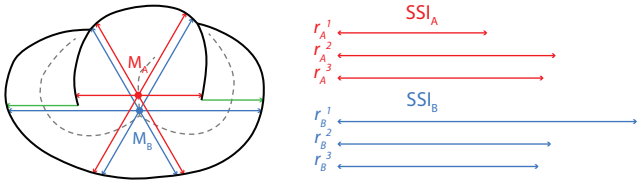

Fig. 6. Example of SSI for points M_A (resp. M_B) constructed by measuring the local reaches r_A^k (resp. r_B^k) in the kth direction of a uniformly sampled set of m directions (here $m = 3$). Some discontinuities of reach are present and shown here as green arrows.

Our new parts-aware metric is inspired by a similar metric [1] developed for 3D shape segmentation. In particular, it makes use of a *Surfacic Shape Image*, a modified 2D version of the

Volumetric Shape Image (VSI) introduced in [1]. However, we tailor this more specifically to our problem, as detailed below.

We define the *Surfacic Shape Image* (SSI) as a signature of silhouette visibility from a given point of view inside a part of the input drawing (see Figure 6). A signature SSI_i is defined as a set of m distances l_i^k between closest points $p_{0,i}^k$ and $p_{1,i}^k$ to the point i in the k^{th} direction of a pre-defined uniformly sampled set of m directions. From 60 to 100 directions per set are used for the different examples in this paper. Both external contours and internal contours are considered at this step, while decorative curves are discarded.

Similar to the 3D case, the local change of SSI (differential of SSI) between two neighboring points can be used to detect a region of possible junction between two structural shape parts. We propose to compute this differential between points A and B as follows:

$$\Delta(\mathrm{SSI})_{A,B} = \frac{1}{\sum_k w_{k,A,B}} \sum_{k=1}^{m} w_{k,A,B} \frac{\left|l_A^k - l_B^k\right|}{\|B - A\|} \tag{3}$$

where

$$l_i^k = \left\| p_{1,i}^k - p_{0,i}^k \right\| \tag{4}$$

Discontinuities near open ends of suggestive contours as depicted in Figure 6 generate outliers. To tackle this problem, we fit a Gaussian to the distribution of such values (as in [1]) using the weights w_k defined as follows:

$$w_{k,A,B} = \begin{cases} e^{-(d_{k,A,B}-u)^2/(2\sigma^2)}, & \text{if } d_{k,A,B} < u + 2\sigma \\ 0, & \text{if } d_{k,A,B} >= u + 2\sigma \end{cases} \tag{5}$$

where $d_{k,A,B} = \left|l_A^k - l_B^k\right| / \|B - A\|$, u is the mean of the $d_{k,A,B}$'s, and σ the standard deviation.

Note that in contrast to [1], $\left|l_A^k - l_B^k\right|$ is not squared in Equation (3), in order to make the equation more linear. In addition, we regularize this term by dividing it by the distance ($\|B - A\|$) between neighboring points of view. This allows for similar results whatever the sampling resolution at which this measure is used. Thus Δ(SSI) tends to be scale-invariant for high resolutions when there is no discontinuity of visibility.

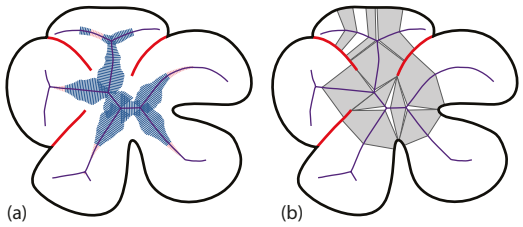

Fig. 7. (a) Computed d_{SSI} between vertices of each edge of the S-skeleton of a part, and represented for each skeleton edge as the size of an orthogonal segment passing through the center of this edge (values are squared for visibility). (b) regions of possible junction (in grey).

Let us now define a part-aware metric built on the SSI. Note that we cannot reuse the method introduced by Liu et al. [1], where the distance used to segment a 3D mesh was defined as the integral of the VSI distance along a geodesic path between vertices. Indeed, in our case, there is no 3D surface to support and define the actual shortest path between two facing vertices on opposite sides of a shape. Therefore, our approach identifies salient parts by defining a 2D Part-Aware metric *along the curves of a specific skeleton*, called the **salient skeleton (S-skeleton)**, as described below.

We initialize the S-skeleton of a shape part as the medial axis of the region bounded by the external contour and the suggestive contours of this part, as shown in Figure 5 (b). Decorative curves are discarded. As usual, it is defined as the locus of the disks of a Medial-Axis Transform (MAT) which are the maximal disks that do not intersect the set of contours (see Figure 3). For the S-skeleton to only reflect the main shape features, we proceed in a fashion similar to the Scale-Axis Transform [21]. The goal is to locally remove small disks by considering those that can be covered by others in a version of the MAT with larger radii. The final result can be realized by computing the MAT of the grown shape defined by the union of scaled up disks from the initial MAT, and then scaling down the radii. However for reasons of computational efficiency and simplicity, we choose to iteratively remove disks from branch extremities in the grown MAT that are covered by others and then scale down the radii of the remaining disks (in practise, we use a scaling factor of 1.3). This avoids several computations leading to similar results. Additionally, the associated contour points of these removed disks are given to the nearest remaining neighboring disk in order to keep a mapping between contours and the S-skeleton.

Given that the drawing represents the silhouettes of a volumetric, organic shape, the S-skeleton is a good candidate for extracting structural information about salient parts. We should however take care of correctly assigning curves in the drawing to the different branches of the S-skeleton. To achieve this, we represent the curves in the drawing, as well as the S-skeleton itself, using half-edges. This enables us to assign the duplicate vertices on two sides of a suggestive contour to different branches of the S-skeleton (see Figure 7).

Our new 2D part-aware metric called **radial variation metric** (d_{SSI}) is defined over the S-skeleton as the integral of the SSI differential (Equation (3)) along the shortest path joining two vertices of the S-skeleton. Let M_A and M_B be two vertices of the S-skeleton and E be the set of edges of the shortest path between them. d_{SSI} is then defined as:

$$d_{\mathrm{SSI}}(M_A, M_B) = \sum_{e \in E} \Delta(\mathrm{SSI})_e \tag{6}$$

The last point is the sampling rate used along the S-skeleton. We note that the contribution of the salient discontinuity Δ(SSI) in region of possible junctions is related to the number of rays included in the parallax angle of a discontinuity location seen from neighbor points. We tried two alternative sampling methods to correctly measure this contribution along the S-skeleton. The first uses a uniform spacing, relative to the drawing size (in practice 0.5% of the drawing's height) while the second uses a dynamic spacing relative to local Medial-Axis disk radii (in practice $0.02 * local\_radius$). While the first method better matches perceptual principles, the second one improves the processing of small details. The results presented in this paper have been generated using the first method for better readability

of figures. We emphasize that even though the measure extends to the continuous case where there is no discontinuity, the resolution of the SSI sample points should not be higher than the one of the input polylines so as not to reflect the lack of curvature of the polylines (otherwise a wave like pattern would emerge).

5.3. Region of possible junction detection

With the S-skeleton S being computed from a medial axis transform, each edge $e \in S$ is associated with two facing portions of the contours, i.e. the segments or point of the contours that could be generated by drawing discs from this specific edge of the skeleton. This enables us to use the S-skeleton to define both regions of possible junction (the regions we are looking for) and the junction boundaries that delimit them on the contours.

We initialize regions of possible junction as the 2D regions corresponding to segments of the S-skeleton with d_{SSI} values over a threshold k (see Figure 7 (b)). Thanks to the scale-independent nature of the metric, a single threshold value k is used regardless of the scale of the input drawing (we use $k = 0.45$ for all our results). These segments are stored using lists of edges of the S-skeleton. Since sharp extremities of structural parts may correspond to large-but-irrelevant d_{SSI} values, we remove them in a second pass: Starting from S-skeleton extremities, we iteratively remove edges while their d_{SSI} values decrease. Increases in d_{SSI} values due to noise can lead to unwanted decomposition of pointy ends, but are mostly avoided by smoothing the d_{SSI} values along the S-skeleton first. Due to the nature of the SSI, sampling points near the middle of a radially symmetric transition will yield low d_{SSI} values compared to other transitions as illustrated in Figure 8. However, we note that the negative curvature cues on both sides are implicitly given by the diamond shape formed between the junction zones (Figure 8 (a)). We merge these regions of possible junction if the distances d_{C0} and d_{C1} between the junctions zones along contours are both inferior to the distance d_M along the S-skeleton, and inferior to the half of the average radius of the Medial-Axis disks of the junctions zones.

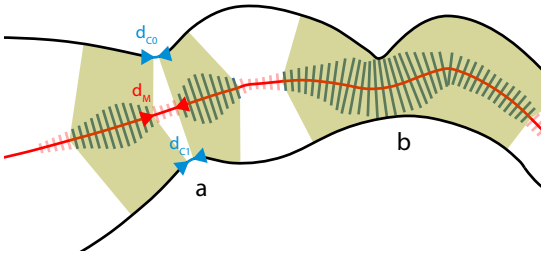

Fig. 8. (a) A region of possible junction misidentified as two junctions due to radial symmetricity of the shape at the center of the transition (in near radial directions to the Medial-Axis). Both are merged before further processing. (b) A more common case of region of possible junction with no merging necessary.

6. Recursive part decomposition

Given the general methods that we have described for closing contours and for extracting regions of possible junction within a structural part, we now detail how a given part is decomposed, i.e., how its regions of possible junction are prioritized, and how the corresponding parts are extracted and completed.

6.1. Prioritizing regions of possible junction

Decomposing a part not only requires extracting possible junction regions between parts, but also assigning them a prioritized order. We achieve this via a classification of regions of possible junction, depending on the type of contour segments that contributed to this specific part of the S-skeleton.

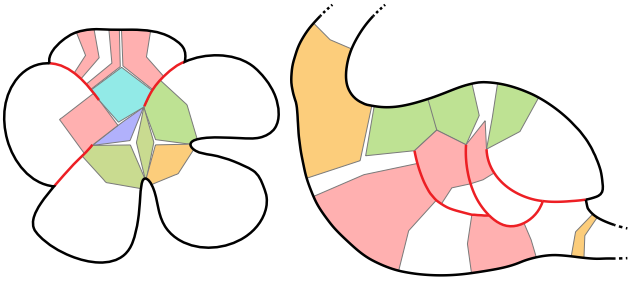

Fig. 9. Regions of possible junction: (SF, SF) **in dark blue,** (SB, SB) **in cyan,** (C, SF) **in green,** (C, C) **in orange, discarded regions in pink.**

Since they are defined by segments of the S-skeleton, each region of possible junction comes together with a pair of junction boundaries (the associated parts of the contours, possibly reduced to a point) found on each side of the skeleton. Regions of possible junction are classified as follows, based on the nature of this pair of junction boundaries (see Figure 9):

1. **Two segments of suggestive contours, that do not belong to the same tree of internal silhouettes:** The junction is either classified (SF, SF) and (SB, SB), depending on whether the suggestive contour's curves (a set of half edges) correspond to a front (occluding) or to a back (occluded) part of the shape. The occluding side is given by the T-junction properties.
2. **A pair formed by an external contour segment and a suggestive contour segment:** We only consider the case when the suggestive contour side corresponds to the front of the shape, denoted as (C, SF).
3. **Two portions of the external contour**: the junction is classified (C, C).

If a curve segment in a pair spans different types of contours, the associated region of possible junction is subdivided. Regions that do not fit into the categories described above (in pink in Figure 9) are discarded, since they have a bounding contour on one side and an occluding one on the other, and this case is not handled by our decomposition.

This classification is used to select the salient parts to be extracted at each stage of the recursive part decomposition algorithm described in Section 3.2: (SF, SF), (SB, SB), (C, SF) and

(C, C) regions of possible junction are respectively given highest to lowest priority. This enables us to give priority to parts that are unambiguously in front of their neighbors, such as for the bottom-right part of the flower in Figure 3, before processing partially occluded parts and those with weaker depth clues.

6.2. Processing complex suggestive contours

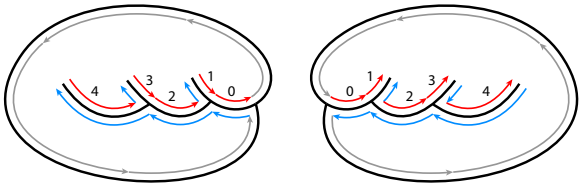

Fig. 10. Adding relative depth information along a suggestive contour, illustrated here for the case of a complex curve forming a tree. As shown here, the numbering must be made in forward or backward fashion depending on the root T-junction.

In addition to the sorting order we just defined, parts defined by (C, SF) junctions need to be given a priority order. This enables us to handle cases where the suggestive contour forms a tree of branching curves, such as the swan's wing in Figure 1.

Let us look at the similar shapes on Figure 10: parts with high label values on the edges need to be extracted first, since they embed the other ones. This enables us to extract the full wing, which can then be progressively decomposed into three consistent parts.

Given that suggestive contours represent internal silhouettes of volumetric parts of a shape that smoothly blend with the parent part, they should have a G^1 continuous junction to the silhouette they are attached to (see Figure 10). The side of this smooth junction indicates which part comes above the rest. Therefore:

1. If the connection point with the external contour is only C^0, the curve tree is re-classified as decoration.
2. If G^1 continuity is detected, we use a traversal of the suggestive contour from the connection point to the open end, on the side of the G^1 continuous curve in order to enumerate and prioritize these half edges for decomposition.
3. During the traversal of the tree, only suggestive contours that demarcate a part that is on the same side as the occluder at the root T-junction are allowed. Other contours are left-out as decorative strokes and are not processed by our algorithm.

Note that even in the case of complex suggestive contours that form a tree as in Figure 10, the suggested part is always on the same side of the curve, given that the organic shape hypothesis would otherwise be violated.

6.3. Part decomposition method

Decomposing a shape part at a given region of possible junction always involves generating two contours for closing the two resulting parts. We use the terms *front closure* and *back closure* to refer to the closure curve that closes the parts lying at the front and back, respectively, given the depth clues provided by the suggestive contours. To decompose a part, we first generate the two closure curves for all the junctions having the highest priority, using specific algorithms for each type of region of possible junction, as detailed below. If either of the resulting closure curves intersects the shape contour or if the front closure curve intersects a decoration curve, the current junction is discarded. Finally, the junctions that result in most plausible closure, as defined by the minimal sum of their closure curves' energy, are selected for the decomposition. The resulting parts are created and included in the partial depth set $\mathcal{P}$ according to their classification as a front or back closure curve.

Note that the two newly created parts may have overlaps between their respective closure curves, but this is valid since they are each assigned a different depth layer. To assign such depth in the case of (C, C) region of possible junction without a relative depth cue, e.g., the bottom-right petal of the flower in Figure 3, we use the convention that the largest shape part should be in front, which is often the best choice when the resulting parts are to be animated.

While inferring the closure of a part given two endpoints and the associated tangent vectors is easy (Section 4), and can be done for connecting two suggestive contours (SF, SF) and (SB, SB) regions of possible junction, the connections in the (C, SF) and (C, C) cases are much more challenging. Indeed, the best pairs of contour points in the region of possible junction zone should be computed for the front and back closure curves. Our methods for solving these two cases are presented next.

6.4. Contour / Suggestive contour (C,SF) closure

Given that we are in the case where the suggested part is on top, we compute all the possible closure curves that join the tip of the suggestive contour to the sample points on the facing contour segment in order to generate the front closure (Figure 11 (b)). We also generate all the possible closure curves joining the contour segment with the T-junction at the base of the suggestive contour tree (Figure 11 (c)). Keeping only pairs of closure curves whose tips on the contour are not farther from each other than the blending radius, we select the most plausible pair of closure curves of minimal energy using the sum of their SIMVC energies (Equation 2). This enables us to efficiently select the best pair of closure curves among the n^2 possible choices.

For this task we must define an adapted sampling rate to explore the space of possible closure curves. This is done by first computing a *blending radius* defined as the average of the radii at the two ends of the region of possible junctions along the S-skeleton. Based on the blending radius, *corner segments* (part of the curve where the curvature is larger than the blending radius) are identified (orange curve segments in Figure 11(a)). For each corner segment, sampling points are assigned at the beginning and end of the segment, and two possible tangents are associated with each curve depending on which of the two closure curves is being computed (Figure 11 (b) and (c)). Other contour parts are regularly sampled with a distance between consecutive samples equal to the half of the blending radius.

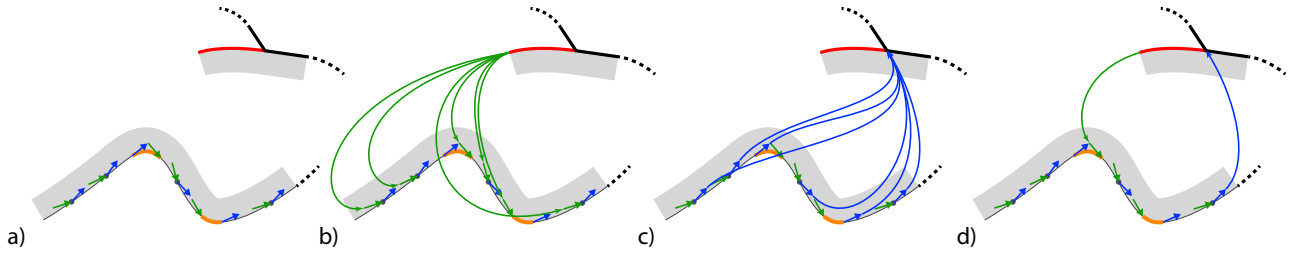

Fig. 11. Part decomposition at a region of possible junction. a) junction boundaries in this case are described by an external contour at the bottom and a red suggestive contour at the top. The external contour is uniformly sampled into a set of points and their associated pair of tangents. Corner segments (orange) samples are given two points instead of one to handle proper connection where curvature is the highest. b) all the closure curves corresponding to pairs of one sample point from the contour and the suggestive contour extremity are computed and their plausibility is evaluated for closing the rightmost part. Curves that intersect contours are eliminated. c) the same closing procedure is applied to the other region, therefore the suggestive contour's T-junction is used instead of its extremity; d) resulting closure curves.

For each such sample point we store the incoming (or outgoing) tangent vectors, with a small tilt outwards (or inwards) in order to avoid undesired intersection.

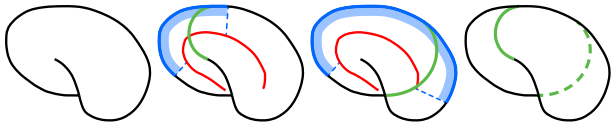

Fig. 12. (a) Example of a suggestive contour comparable to a cusp requiring a (C,SF) closure. (b) The front closure unknown end-point is searched along the blue contour part defined by the identified region of possible junction. (c) The transition is extended for the back closure. (d) Result of the decomposition with parts sharing a wide part of contour.

When dealing with suggestive contours that could be considered as big cusps (Figure 12 (a)), the front closure is processed normally (Figure 12 (b)) while for the back closure, we extend the relevant transition contour (Figure 12 (c)). The transition contour is extended by following its associated S-skeleton branch until either an extra branch is encountered, or the facing contour is a neighbor of the T-junction. This produces the result exposed in Figure 12 (d).

6.5. Contour / Contour (C,C) closure

We sample the boundaries of the regions of possible junction in the same way as for the (C, SF) case. The two closure curves' extremities may be located anywhere on these segments. With n sample points on both contours, a naive method would lead to n^2 possible curves to generate for each of the parts, and thus to n^4 pairs of closure curves to evaluate. To reduce the complexity back to n^2, we only consider the pairs of closure curves between a given pair of points.

In this case, we use a variation on the energy for selecting the most plausible pair of curves. Rather than selecting the shortest closures, we wish to favor a decomposition with an overlap that is close to the middle of the region of possible junction. Doing so ensures that curved parts such as a cat's tail is regularly decomposed regardless of the small variations in thickness. Therefore, we use the energy $\hat{E}$ of the two curves to select

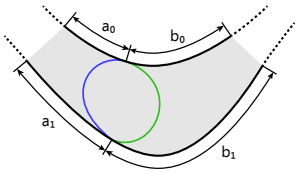

Fig. 13. For (C, C) closures we define a new coefficient for $\bar{E}$ based on the samples' positions relative to their respective sampling contours as defined in Section 6.5.

the best closing pair, defined as:

$$\hat{E} = \left(1 + \frac{(1 - \frac{a_0}{0.5*l_0})^2 l_0 + (1 - \frac{a_1}{0.5*l_1})^2 l_1}{l_0 + l_1}\right)(E^0_{\text{SIMVC}} + E^1_{\text{SIMVC}})$$

with $l_i = a_i + b_i$ and a_i, b_i the junction boundary segments arc lengths shown in Figure 13, E^0_{SIMVC} and E^1_{SIMVC} the energies of the closure curves.

We also wish to reward the use of corners over a solution with two circular arcs forming a circle since it has an energy close to zero. Thus valid closures that use a sample at a corner as an implicit end point are given priority.

7. 2D decomposition results

We now present our decomposition results and illustrate some applications in Section 8. Our method, its results and applications are also presented in the accompanying video.

7.1. Qualitative results

Figures 1, 3, 15, 22, 25 and 26 show a variety of shape decomposition and layering results that are automatically computed by our method. In Figure 25, we reused drawings from a recent paper [32], showing that our method achieves the structuring and layering of such drawings without the need of any extra information, whereas a user-defined 3D skeleton was used in the original paper. Figures 1 and 15 (top-right) show even more challenging cases where some of the suggestive contours form a chain of T-junctions, requiring the labeling method of Section 6.2 in order to be properly processed. Figure 26 shows results computed on cat drawings found on the web using a simple query and vectorized using Adobe Illustrator.

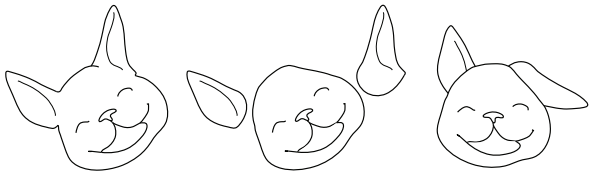

Fig. 14. Dog head example, and some editing by manipulating the extracted parts.

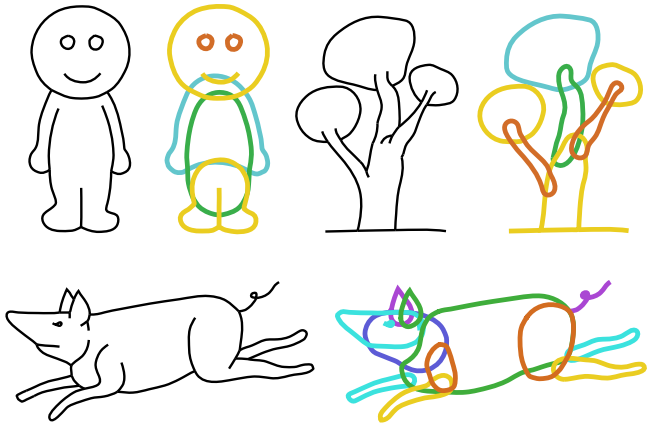

Fig. 15. Results on drawings of a cartoonish man, a tree and a pig. The two legs of the man are seen from a special view, thus the surface contact is classified as a decorative element. Warmer colors are in the foreground.

7.2. Quantitative results

Computational times range from a few seconds for the dog head of Figure 14, which includes a number of decorative curves and only three overlapping, structural regions, to a few minutes for complex drawings such as the swan of Figure 1. See Table 1. The flower is the one of Figure 3, which shows all the decomposition steps plus the final results, and the cat is the one on Figure 22.

Figure	Dog	Flower	Swan	Cat
T1 (in seconds)	19.1	26.4	708.1	520.1
T2 (in seconds)	7.2	9.8	134.0	85.0

Table 1. Computational times without (T1) versus with (T2) the optimization of contour completion described in Section 4.2. It can be seen that our optimization method enables us to reduce computational time by a factor of 3 to 6.

This table enables us to emphasizes the benefits of the optimization method we proposed for contour completion (Section 4.2), enabling to reduce computational time by a factor 5 or 6 in challenging cases such as the swan and the cat.

Validation

Segmentation in depth ordered structural parts is a fundamental first step for many further applications, from editing vector drawings with robust completion of partially hidden parts, to 2D animation and sketch-based 3D modeling.

We first tested our method with two applications in mind: the conversion of the input drawing into a Vector Graphics Complex [38] that then enables easy and meaningful 2D editing, and 2D vector animation. Figure 14 shows an example of meaningful editing of a drawing. Posing and animation results are shown respectively in Figure 1 and in the supplemental video. The decomposition of the wing of the swan may not fit the perceived structure for every viewer, but wings are not known to be easily animatable in 2D.

8. Applications

In addition of helping in animating static sketches, our method can be used in a variety of applications. In this section, we present two of them: cardboard puppetery and part-based 3D modeling.

8.1. Cardboard puppetry

Puppetry is an art-form that has existed for a very long time. Given a set of shapes, characters can be crafted by taking the range of possible motion into account [42]. Puppet manipulation can then either be made by manually posing puppets or by using rod-like structures as is done in traditional shadow-puppetry. However this type of puppetry requires the input parts to be simple, non-intersecting shapes. If the input itself is a complicated single sketch, then it has to be converted into small pieces to be handled by such approaches.

In our application, puppets are made of articulated parts that can be posed by a user. Once the input sketch is automatically divided into parts using our method, it can be appropriately connected and manipulated. However, though our technique provides structuring and layering information of the decomposed parts, it must be extended to provide the appropriate hinge locations in order to generate a fully connected puppet.

We propose to connect two overlapping structures by placing a hinge enabling the appropriate flawless rotation. We propose a simple heuristic to identify a consistent hinge position. For given intersecting structures, the area of intersection is computed and the maximal radius circle fitting inside this intersection is found. The hinge is then placed at the center of this circle. Figure 17 shows two intersecting shapes, the intersection region and the computed maximal radius circle.

Since our objective is to create fully-connected puppets, each extracted part should be connected to the part on which it rests. After each step of recursive part decomposition, the maximal radius circle lying in the intersection area of the extracted part along with the parent part is computed and shown as a suggested hinge position for the user. Various puppets fabricated and connected with our technique are shown in Figure 16. Maximal radius circles (black colored circles) along with hinge positions (red colored dots) are shown in the second column of Figure 16.

8.2. Part-based 3D modeling

From a given 2D sketch, several methods are able to reconstruct a corresponding 3D model. These algorithms are generally designed to support only simple curves as input. In addition to complicating the interpretation of the 3D shapes, the

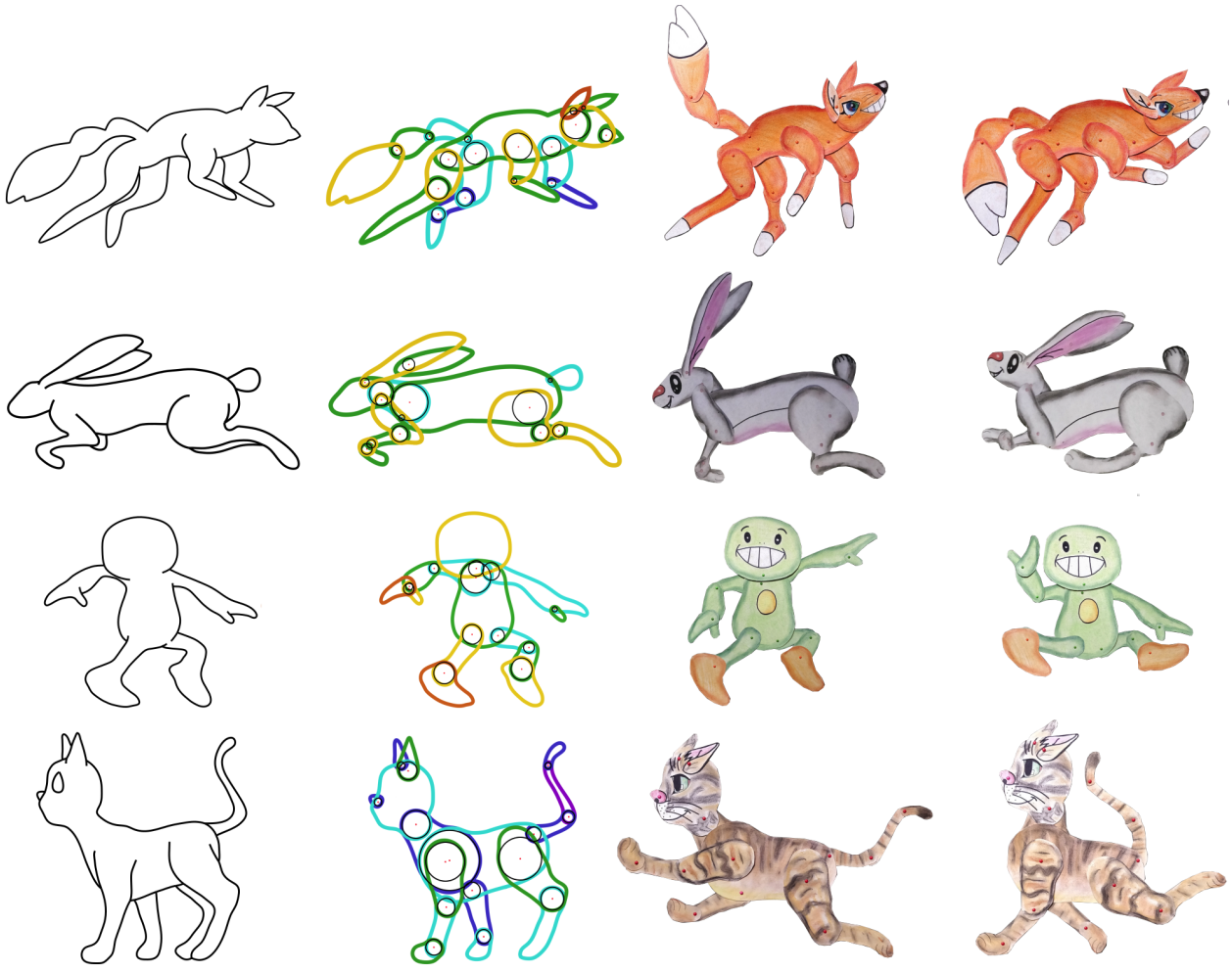

Fig. 16. Fabrication of articulated puppets from a single sketch: With a simple extension, we can use our system to suggest hinge positions between appropriate parts in order to fabricate cardboard puppets. This figure shows the input sketches and hinge positions along with various poses of puppets we fabricated from them.

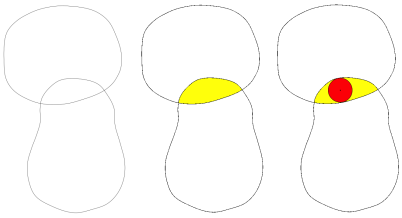

Fig. 17. (left) Two sample intersecting curves, (middle) intersection region and (right) maximum radius circle lying inside the area of intersection.

use of complex sketches with internal contours as input also requires special techniques to infer depth information between the model parts. We propose a method to automate this 3D modeling procedure by decomposing sketches with connected inner contours into inflatable parts that are positioned in depth afterward. Our current implementation makes use of the recent algorithm of Parakkat et al. [43] for seeks of simplicity. Indeed implicit surfaces [31] would have been an excellent choice too, and would have led to smoother results. The inflation algorithm from [43] is modified as follows:

1. The pixels lying on the outer boundary of the sketch are extracted and used to create a point set S.
2. A Delaunay triangulation of S is computed.
3. The Delaunay triangulation is restricted inside the sketch, similarly to [44]. This procedure removes Delaunay triangles located outside the shape.
4. Each non-boundary triangle edge is converted to a circle and is stitched with its neighbours [43] to generate a 3D inflation of the part.
5. A uniform sampling is applied to smooth the reconstructed model.

Figure 18 illustrates the various steps of the inflation algorithm. Figures (a) to (e) respectively show an input sketch, its Delaunay triangulation, the removal of the outside Delaunay triangles, the mesh inflation and the result after smoothing.

The inflation algorithm is separately applied to each individual, decomposed part. In our implementation, a fast user intervention is then required to set the relative depth value of each

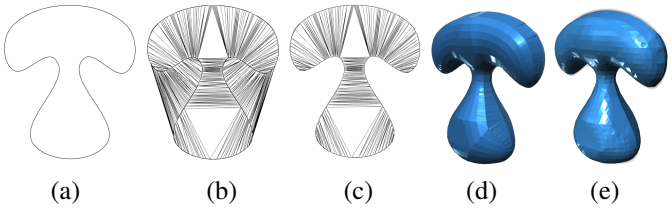

Fig. 18. (a) Input sketch, (b) Delaunay triangulation, (c) Result after restricted Delaunay, (d) triangulation inflation, (e) final 3D model after uniform sampling.

inflated part. This simple user interaction could be automated using the depth assignment procedure presented by Entem et al. [31]. Figure 19 shows several sketches with their 3D reconstructions. It has to be noted that, since the occluded legs (cat and rabbit) are anatomically incorrect, the symmetric copies of the foreground legs have been used to create the 3D models. Table 2 shows the time taken for creating 3D models and assigning depths. The generated shapes assume circular limbs and do not include more complex anatomic details. Additional user interaction would be required to improve the quality of the final shapes.

Figure	Fox	Bunny	Character	Cat
Modeling time(secs)	4.31	4.09	3.997	5.162
Ordering time(secs)	75	96	20	92

Table 2. Time taken for 3D modeling of parts and their depth arrangement.

9. Discussion

Our method for structuring complex drawings performs as expected in most cases. We point out that once a drawing is processed, the union of the extracted parts does not necessarily exactly correspond to the initial outline since the blending between parts is not conserved. However, it is very similar and it would be possible to retrieve a similar outline by representing the parts' contours as iso-contours of 2D scalar fields and blending them together. Scalar fields with skeleton could also allow for easy directional rescaling of structural parts to allow for illusory 3D rotations. For instance, this extension could be used to improve the animation of our swan's wings.

In some cases, a valid drawing may be ambiguous and gives rise to several different interpretations. This is the case for the example in Figure 20, where the shape could either be interpreted as a boxing glove (b) or as a snail head protruding out of the shell (c). Our method will output a single result in such cases, the one in (c), because of the way we process complex suggestive contours. Some curves that would be processed naturally by a human as contours are not processed when there is no T-junction such as for the beak of the bird in Figure 1.

Lastly, similarly to the metric in [1], our d_{SSI} metric could also be used to define a distance between two points of the contour, which we would define as the d_{SSI} distance between their two corresponding S-skeleton's vertices. In future work we wish to explore the possible applications of this new metric to contour drawings.

9.1. Comparison with previous work

The closest work we can compare with is SmoothSketch [25]. While they look for a plausible completion of the hidden contour hinted by cusps, we instead use the facing contour to smoothly wrap a curve around the hypothetical 3D shape as shown in Figure 21. We emphasize that the segment of our hidden closure that is near the T-junction is a plausible hidden contour. Thus, our method could benefit figural completion of cusps in SmoothSketch's failure cases. However, we think that both this algorithm and ours should be used in a complete application since hidden contour completion is more meaningful than our decomposition for the case of cusps that are small relative to the local thickness of the shape, and specific cases such as the swan's wing.

The recursivity of our decomposition hides some perceptual information such as similarity, grouping, symmetry. In the paw example in Figure 21(c,d), we perceive the similarity and symmetry of the fingers. However our algorithm first decomposes the foreground finger, and considers the two others as a whole, thus the background finger is eventually perceived as big as the middle one once the first is extracted. Designing a global method from our recursive one is a non trivial problem since the closures are interdependent in many cases.

We also show results on inputs from [31] in Figure 22 (top, middle). Though the segmentation is similar, our initialization step cannot complete complex hidden contours. This limitation would locally require a completion similar to the one used in [34] but it is non trivial to combine this method with figural completion to be able to complete parts with distinct visible regions in the absence of distinct similarities between these regions such as color or grouping annotations. However our algorithm tackles the case of parts hinted by single suggestive contours as shown in Figure 22 (bottom).

9.2. Limitations

Even though our method gives good results in most cases, we have a few limitations as well. One main problem is the cyclic arrangement of parts over one another. Figure 23(a) shows an example in which the petals are overlapping in a cyclic fashion. In this case, layering cannot be done using the proposed method and our part decomposition phase fails. Since we are assuming that the occlusion results in either T-junctions or cusps, the more complex junctions are not processed in the current system.

Many limitations are found in the initialization step, when drawings carry ambiguities at curve intersections, notably when T-junctions are not well defined either due to special view or surface contact as in Figure 23 (b,c) and Figure 26 (2,3), or misleading because the hypothetical occluded contour is in fact a texture change as in Figure 23 (e, left). The latter limitation can be manually worked around by erasing a part of the stroke as in Figure 23 (e, right).

9.3. Future Work

Our current method can be improved in various ways. One of them is to include user interventions to solve ambiguous cases (recognize whether two concentric circles represent two spheres or a torus), and to do assisted segmentation as in [34]. Another

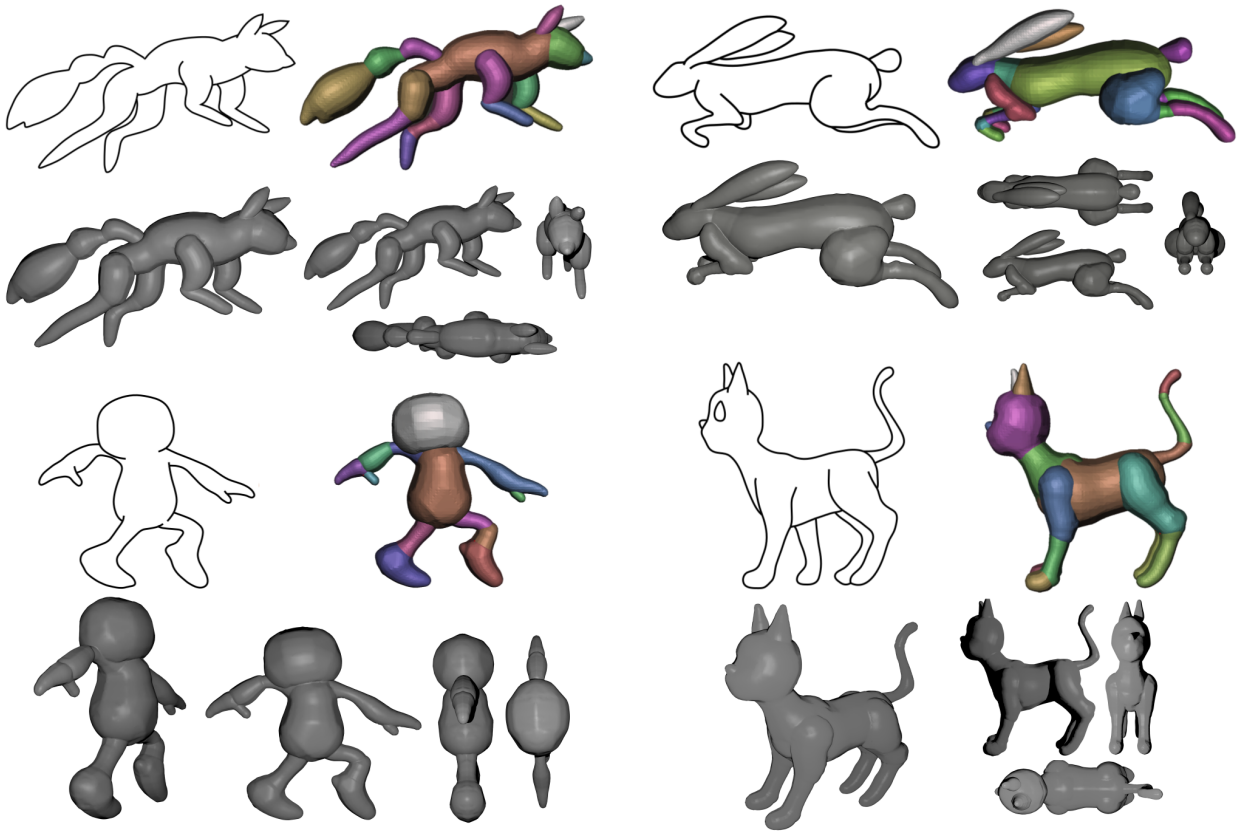

Fig. 19. 3D modeling from a single, complex organic sketch. the resulting model is first depicted from the drawing viewpoint and then from various views.

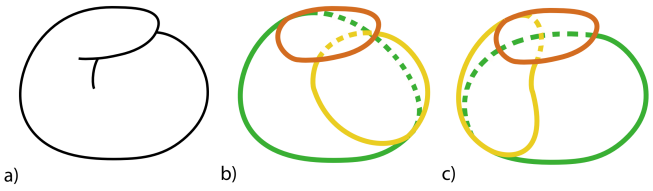

Fig. 20. An ambiguity of interpretation. Our algorithm can only produce the result (c) while (b) would also be a perceptually plausible result.

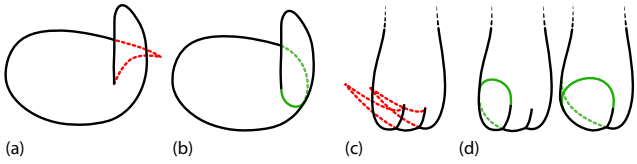

Fig. 21. Comparison of our results (b,d) with SmoothSketch's failure cases (a,c).

interesting direction is to use decorative curves as suggestive contours. Figure 24 shows a case in which the decorative curve represents a suggestive curve and our method would ignore it.

In terms of applications, a first interesting avenue for future work would be to extend our work from animated puppets to 2D vector animation, in order to produce animated results such as the manually created flying swam example at the bottom of Figure 1 (c). To achieve this, in addition to the method for computing hinges we already discussed, an automatic mechanism to clean and blend 2D contours of animated subshapes should be provided. This could be done with the help of 2D implicit contours and specially designed combination operators for curves. Note that beyond local blending, the main challenge here is to automatically remove the non-salient parts of the contours of overlapping parts, while ensuring temporal coherence. In this context, the 3D reconstructions we already provided could serve as a 3D intermediate representation. It could also allow us to apply out-of-plane rotations to limbs and to change the viewpoint in animated sketches, two cases in which the silhouettes and occlusions between shape parts need to be recomputed.

Lastly, 3D shape reconstruction from a 2D sketch would benefit from automatic depth inference, which we could do using the amount of overlap between shape parts. Decoration curves could then be projected back onto the 3D model. Moreover, our current 3D reconstruction method could be improved by smoothly blending parts thanks to implicit modeling, similarly to what was done in [31, 32] in a more constrained case.

10. Conclusion

We presented the first automatic method able to use complex inner contours in the analysis and recursive decomposition of drawings that represent smooth shapes. Our decomposition method outputs a structure of closed 2D shapes layered in depth. It relies on the inference and progressive refinement of a partial depth tree to store depth information. A new metric

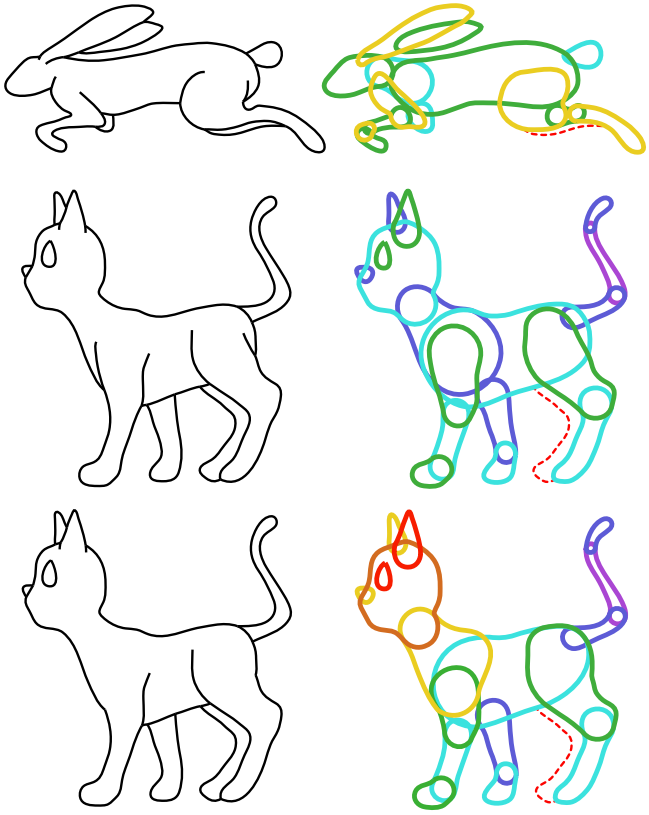

Fig. 22. Results on two drawings from [31] and the third is the second drawing minus two suggestive contours, a case that their algorithm could not handle. Red dotted curves are contours that make our initialization stage fail (they are manually removed before further processing).

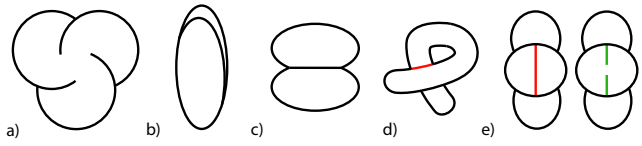

Fig. 23. Different limitation cases of the initialization step (b, c, d, e) and recursive decomposition algorithm (a).

computed along a skeleton was proposed to detect salient parts of complex drawings including internal silhouette curves. We introduced a new, perception-based criterion for selecting the most salient possible junction regions, prioritizing them, and using them to recursively segment a shape into parts. An efficient implementation of curve closures using a variation of the Scale-Invariant MVC functional was defined for closing the extracted parts, hidden or salient. Lastly, we managed to keep most parameters scale-invariant, enabling us to achieve structural decomposition of drawings with different resolutions of features.

Many applications can benefit from our method. As we have illustrated, it enables organic sketches to be easily edited in a meaningful way. Subdivision and depth layering makes the model ready for simple 2D animations. As discussed in Section 8, the decomposition algorithm aids in part-based 3D shape modeling, a domain where complex sketches with a variety of

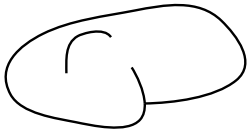

Fig. 24. An input where a suggestive contour is not connected to the outer contour and thus misclassified as a decorative element in our initializaton step. We would like to use it as a possible segment of a foreground closure in future works.

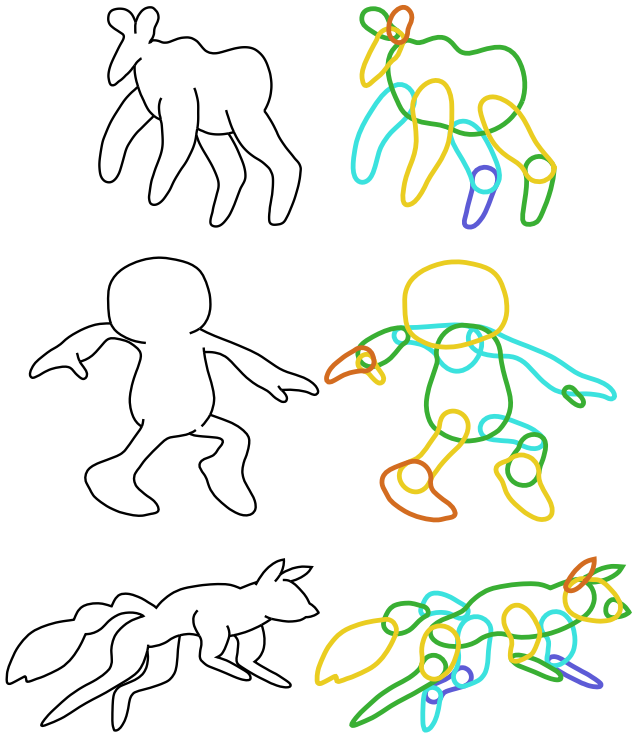

Fig. 25. Results on three drawings also used in [32], except that we removed the hat of the character. Warmer colors are in the foreground.

internal contours were never considered so far. As future work, we also plan to further ease 2D animation of complex sketches by developing an automatic mechanism to locally clean and blend part contours at each animation step, in order to enable smooth transitions between silhouette curves where and when needed.

Acknowledgments

A preliminary version of this paper was published at the Expressive 2018 conference. We wish to thank all the anonymous reviewers of our work for their valuable comments. Many thanks to Michiel van de Panne and to John Hughes for proofreading previous versions of ths paper. This work was partly funded by the ERC advanced grant EXPRESSIVE.

References

[1] Liu, R, Zhang, H, Shamir, A, Cohen-Or, D. A part-aware surface metric for shape analysis. Computer Graphics Forum 2009;28(2):397–406.

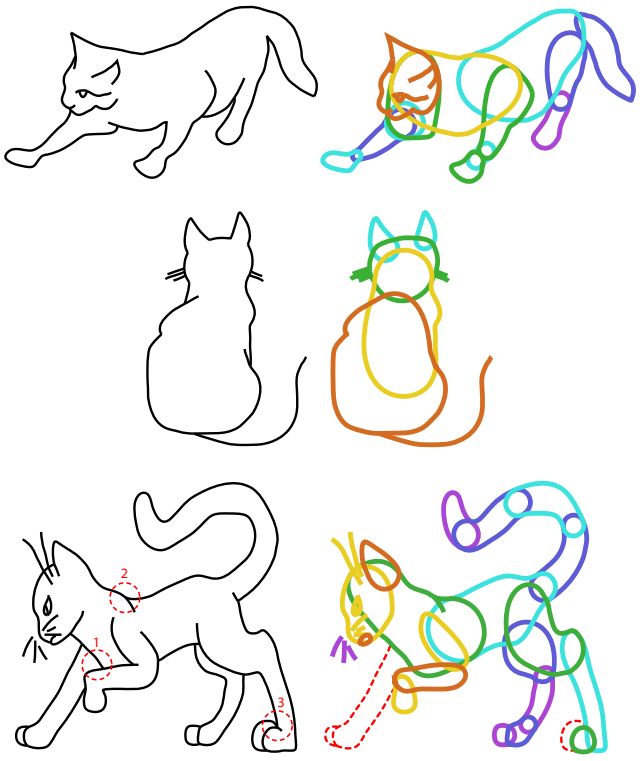

Fig. 26. Results for three drawings found on the first results page of Google Images with the following search terms: "cat line drawing". The second result has been produced with salience threshold parameter that is more sensitive than the one used for all the other examples. The third example is subject to three limitations: 1) non trivial hidden part completion; 2) badly defined T-junction; 3) 4-valence vertex.

[2] Singh, M. Visual representation of contour and shape. In: Oxford handbook of perceptual organization. Oxford: Oxford University Press; 2015,.

[3] Ullman, S. Filling-in the gaps: The shape of subjective contours and a model for their generation. Biological Cybernetics 1976;25(1):1–6. URL: http://dx.doi.org/10.1007/BF00337043. doi:10.1007/BF00337043.

[4] Nitzberg, M, Mumford, D. The 2.1-D sketch. In: ICCV. IEEE. ISBN 0-8186-2057-9; 1990, p. 138–144.

[5] Williams, LR, Hanson, AR. Perceptual completion of occluded surfaces. Computer Vision and Image Understanding 1996;64(1):1–20.

[6] Horn, BKP. The curve of least energy. ACM Transactions on Mathematical Software 1983;9(4):441–460. URL: http://doi.acm.org/10.1145/356056.356061. doi:10.1145/356056.356061.

[7] Moreton, HP. Minimum curvature variation curves, networks, and surfaces for fair free-form shape design. Ph.D. thesis; Berkeley, CA, USA; 1992. UMI Order No. GAX93-30652.

[8] Blum, H, Nagel, R. Shape description using weighted symmetric axis features. PR 1978;10:167–180.

[9] Richards, WA, Koenderink, JJ, Hoffman, DD. Inferring three-dimensional shapes from two-dimensional silhouettes. Journal of the Optical Society of America A 1987;4(7):1168–1175. URL: http://josaa.osa.org/abstract.cfm?URI=josaa-4-7-1168. doi:10.1364/JOSAA.4.001168.

[10] Latecki, LJ, Lakamper, R. Convexity rule for shape decomposition based on discrete contour evolution. Computer Vision Image Understanding 1999;73(3):441–454.

[11] Yang, M, Kpalma K., I, Ronsin, J. A Survey of Shape Feature Extraction Techniques. Pattern Recognition, Peng-Yeng Yin (Ed) (2008) 43-90 2008;:43–90URL: http://hal.archives-ouvertes.fr/docs/00/44/60/37/PDF/ARS-Journal-SurveyPatternRecognition.pdf.

[12] Shamir, A. A survey on mesh segmentation techniques. Computer Graphics Forum 2008;27(6):1539–1556. URL: http://dx.doi.org/10.1111/j.1467-8659.2007.01103.x. doi:10.1111/j.1467-8659.2007.01103.x.

[13] Larsson, L, Morin, G, Begault, A, Chaine, R, Abiva, J, Hubert, E, et al. Identifying perceptually salient features on 2d shapes. Research in Shape Modeling 2015;doi:10.1007/978-3-319-16348-2_9.

[14] Carlier, A, Leonard, K, Hahmann, S, Morin, G. The 2d shape structure dataset: A user annotated open access database. Computer and Graphics 2016;58.

[15] Leonard, K, Morin, G, Hahmann, S, Carlier, A. A 2d shape structure for decomposition and part similarity. In: Proceedings of ICPR. Cancun, Mexico; 2016,.

[16] Pessoa, L, Weerd, PD. Filling-in: From perceptual completion to cortical reorganization. 2003.

[17] Igarashi, T, Mitani, J. Apparent layer operations for the manipulation of deformable objects. ACM Trans Graph 2010;29(4):110:1–110:7. URL: http://doi.acm.org/10.1145/1778765.1778847. doi:10.1145/1778765.1778847.

[18] Noris, G, Sykora, D, Shamir, A, Coros, S, Whited, B, Simmons, M, et al. Smart scribbles for sketch segmentation. Computer Graphics Forum 2012;31(8):2516–2527. URL: http://dx.doi.org/10.1111/j.1467-8659.2012.03224.x. doi:10.1111/j.1467-8659.2012.03224.x.

[19] Mi, X, DeCarlo, D, Stone, M. Abstraction of 2d shapes in terms of parts. In: Proceedings of the 7th International Symposium on Non-Photorealistic Animation and Rendering. NPAR '09; New York, NY, USA: ACM. ISBN 978-1-60558-604-5; 2009, p. 15–24. URL: http://doi.acm.org/10.1145/1572614.1572617. doi:10.1145/1572614.1572617.

[20] Liu, X, Wong, TT, Heng, PA. Closure-aware sketch simplification. ACM Trans Graph 2015;34(6):168:1–168:10. URL: http://doi.acm.org/10.1145/2816795.2818067. doi:10.1145/2816795.2818067.

[21] Giesen, J, Miklos, B, Pauly, M, Wormser, C. The scale axis transform. In: Proceedings of the Twenty-fifth Annual Symposium on Computational Geometry. SCG '09; ACM. ISBN 978-1-60558-501-7; 2009, p. 106–115.

[22] DeCarlo, D, Finkelstein, A, Rusinkiewicz, S, Santella, A. Suggestive contours for conveying shape. In: ACM SIGGRAPH 2003 Papers. SIGGRAPH '03; New York, NY, USA: ACM. ISBN 1-58113-709-5; 2003, p. 848–855. URL: http://doi.acm.org/10.1145/1201775.882354. doi:10.1145/1201775.882354.

[23] Eisemann, E, Paris, S, Durand, F. A visibility algorithm for converting 3d meshes into editable 2d vector graphics. ACM Trans Graph 2009;28(3):83:1–83:8. URL: http://doi.acm.org/10.1145/1531326.1531389. doi:10.1145/1531326.1531389.

[24] Kunsberg, BS, Zucker, SW. Critical contours: An invariant linking image flow with salient surface organization. CoRR 2017;abs/1705.07329. URL: http://arxiv.org/abs/1705.07329. arXiv:1705.07329.

[25] Karpenko, OA, Hughes, JF. Smoothsketch: 3d free-form shapes from complex sketches. ACM Trans Graph 2006;25(3):589–598. URL: http://doi.acm.org/10.1145/1141911.1141928. doi:10.1145/1141911.1141928.

[26] Cordier, F, Seo, H. Free-form sketching of self-occluding objects. IEEE Comput Graph Appl 2007;27(1):50–59. URL: http://dx.doi.org/10.1109/MCG.2007.8. doi:10.1109/MCG.2007.8.

[27] Xu, B, Chang, W, Sheffer, A, Bousseau, A, McCrae, J, Singh, K. True2form: 3D curve networks from 2D sketches via selective regularization. Transactions on Graphics (Proc SIGGRAPH 2014) 2014;33(4).

[28] Yeh, CK, Huang, SY, Jayaraman, PK, Fu, CW, Lee, TY. Interactive high-relief reconstruction for organic and double-sided objects from a photo. IEEE Transactions on Visualization and Computer Graphics 2017;23(7):1796–1808. doi:10.1109/TVCG.2016.2574705.

[29] Cordier, F, Seo, H, Park, J, Noh, JY. Sketching of mirror-symmetric shapes. IEEE Transactions on Visualization and Computer Graphics 2011;17(11):1650–1662.

[30] Turquin, E, Wither, J, Boissieux, L, Cani, MP, Hughes, JF. A sketch-based interface for clothing virtual characters. IEEE Computer Graphics and Applications 2007;27(1):72–81.

[31] Entem, E, Barthe, L, Cani, MP, Cordier, F, Van De Panne, M. Modeling 3d animals from a side-view sketch. Computers & Graphics 2014;URL: https://hal.inria.fr/hal-01073059. doi:10.1016/j.cag.2014.09.037.

[32] Bessmeltsev, M, Chang, W, Vining, N, Sheffer, A, Singh, K.

Modeling character canvases from cartoon drawings. ACM Trans Graph 2015;34(5):162:1–162:16. URL: http://doi.acm.org/10.1145/2801134. doi:10.1145/2801134.

[33] Rivers, A, Igarashi, T, Durand, F. 2.5d cartoon models. ACM Trans Graph 2010;29(4):59:1–59:7. URL: http://doi.acm.org/10.1145/1778765.1778796. doi:10.1145/1778765.1778796.

[34] Sýkora, D, Kavan, L, Čadík, M, Jamriška, O, Jacobson, A, Whited, B, et al. Ink-and-ray: Bas-relief meshes for adding global illumination effects to hand-drawn characters. ACM Trans Graph 2014;33(2):16:1–16:15. URL: http://doi.acm.org/10.1145/2591011. doi:10.1145/2591011.

[35] Bui, MT, Kim, J, Lee, Y. 3d-look shading from contours and hatching strokes. Computers & Graphics 2015;51:167 – 176. URL: http://www.sciencedirect.com/science/article/pii/S0097849315000734. doi:https://doi.org/10.1016/j.cag.2015.05.026; international Conference Shape Modeling International.

[36] Wiley, KB, Williams, LR. Representation of interwoven surfaces in $2^1/_2$d drawing. IEEE Computer Graphics and Applications 2006;27(4):70–83.

[37] Wiley, KB. Druid: Use of crossing-state equivalence classes for rapid relabeling of knot-diagrams representing $2^1/_2$D scenes. 2006.

[38] Dalstein, B, Ronfard, R, van de Panne, M. Vector graphics complexes. ACM Trans Graph 2014;33(4):133.

[39] Noris, G, Hornung, A, Sumner, RW, Simmons, M, Gross, M. Topology-driven vectorization of clean line drawings. ACM Trans Graph 2013;32(1):4:1–4:11.

[40] Kronrod, AS. Nodes and weights of quadrature formulas: sixteen-place tables. Consultants Bureau 1965;.

[41] Powell, MJD. An efficient method for finding the minimum of a function of several variables without calculating derivatives. Computer Journal 1964;7(2):155–16.

[42] Megaro, V, Thomaszewski, B, Gauge, D, Grinspun, E, Coros, S, Gross, M. Chacra: An interactive design system for rapid character crafting. In: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation. SCA '14; Aire-la-Ville, Switzerland, Switzerland: Eurographics Association; 2014, p. 123–130. URL: http://dl.acm.org/citation.cfm?id=2849517.2849538.

[43] Parakkat, AD, Joshi, SA, Pundarikaksha, UB, Muthuganapathy, R. Sketch and shade: An interactive assistant for sketching and shading. In: Proceedings of the Symposium on Sketch-Based Interfaces and Modeling. SBIM '17; New York, NY, USA: ACM. ISBN 978-1-4503-5079-2; 2017, p. 11:1–11:2. URL: http://doi.acm.org/10.1145/3092907.3122799. doi:10.1145/3092907.3122799.

[44] Methirumangalath, S, Parakkat, AD, Muthuganapathy, R. A unified approach towards reconstruction of a planar point set. Comput Graph 2015;51(C):90–97. URL: http://dx.doi.org/10.1016/j.cag.2015.05.025. doi:10.1016/j.cag.2015.05.025.