Computer-Aided Design 45 (2013) 301–311

Contents lists available at SciVerse ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

Inferring mirror symmetric 3D shapes from sketches[☆]

Frederic Cordier[a,*], Hyewon Seo[b], Mahmoud Melkemi[a], Nickolas S. Sapidis[c]

[a] *LMIA, Université de Haute Alsace, France*
[b] *Université de Strasbourg (UMR 7005 CNRS), France*
[c] *University of Western Macedonia, Greece*

ARTICLE INFO

Keywords:
3D reconstruction
Sketch-based modeling

ABSTRACT

We describe a system for taking a 2D sketch of a mirror-symmetric 3D shape and lifting the curves to 3D, inferring the symmetry relationship from the original hand-drawn curves. The system takes as input a hand-drawn sketch and generates a set of 3D curves such that their orthogonal projection matches the input sketch. The main contribution is a method which is able to identify the symmetry relationship among the hand-drawn curves even in the presence of ambiguity in the sketch.

1. Introduction

Freehand sketching is one of the most common ways to communicate ideas. Sketches have been used since ancient times in the form of hieroglyphs. Nowadays, sketches are used for many purposes such as the creation of computer games, movie script, industrial product design, etc. Sketching and understanding sketches is an inherent part of human comprehension [1].

However, 3D reconstruction from sketches is particularly difficult. The main issues of reconstruction of 3D shapes from sketches are the interpretation of the sketch, the reconstruction of the occluded parts and the computation of the 3D shape using the 2D data.

In this paper, we present a method for the 3D reconstruction of a particular class of objects, which are sets of mirror-symmetric curves. Mirror symmetric shapes are symmetric with respect to a plane (also known as a symmetry plane). Many, if not most, human-made objects are mirror symmetric. We believe that a sketching interface for symmetric shapes would be useful.

The choice of reconstructing 3D curves rather than 3D surfaces is motivated by the fact that the problem of occluded surface does not exist for curves, contrary to surface sketching. Also, another advantage is the possibility to create much more complex models. For instance, by sketching the curves instead of the surfaces, the user is able to draw all the details of a car model using a single sketch. If the user draws the silhouette of the car body, the occluded parts from the backside are missing; the user has to provide another sketch showing the back view. Thus, wireframe sketching is often used in Computer Aided Design where the visible and occluded parts of the model have to be carefully designed. If the modeling of surfaces is required, our approach can be combined with existing techniques that generate 3D surfaces from 3D curves.

Our contribution is a method to identify automatically the symmetry relationships between the input 2D curves. In particular, our method identifies the pairs of symmetric curves, the self-symmetric curves and the curves with no symmetry. Our method is able to process sketches with inherent ambiguity. It always finds a unique solution by using the curve connectivity and maximizing the compactness of the reconstructed models. Using the symmetry relationship, we then compute the 3D reconstruction of the curves.

2. Related work

The most common approach to 3D modeling with a sketching interface is to require the user to draw the visible and hidden contours of the rectilinear shape to be modeled. The reconstruction is usually formulated as an optimization problem. The variables of the objective functions are the missing depth of the vertices of the drawing (and possibly other parameters). Different objective functions have been proposed, such as minimizing the standard deviation of the segment magnitudes [2] or minimizing the entropy of the angle distribution [3]. Liu et al. [4] proposed a different approach for which the variables of the objective function are the parameters of the planes that pass through the planar faces of the model to reconstruct. All of these reconstruction techniques are particularly suitable for the design of CAD-like geometric shapes. However, the hypothesis they use allows modeling of rectilinear shapes only and is not suitable for free-form modeling.

Another group [5] presented a sketching interface for free-form modeling of surfaces. In their system, the user creates a shape by

☆ The research of N. Sapidis has been co-financed by the European Union (European Social Fund — ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Program: THALIS.

* Corresponding author.
E-mail address: fredcord@gmail.com (F. Cordier).

drawing its 2D silhouette; a 3D mesh is generated by inflating the region surrounded by the silhouette, making wide areas fat and narrow areas thin. The created model can be then modified interactively with a set of tools that cuts, extrudes, bends, or draws on the mesh. Others [6–9] have also proposed methods to create 3D models from 2D silhouette curves. Unlike the system proposed by Igarashi et al. [5], the user can create self-occluding objects. Another difference is that the curves of the 2D drawing are processed in conjunction, and no modification is allowed after the creation of the 3D model. The purpose of these works is the modeling of surfaces. In this paper, we focus on the modeling of 3D curves.

Cohen et al. [9] proposed a method for 3D curve modeling with which the user creates non-planar curves. The main disadvantage is that the user must draw two curves: the curve itself and its shadow on the floor plane. Research has been done for skewed mirror symmetry, which is a particular case of 3D curves. Skewed-mirror symmetry depicts a mirror-symmetric planar curve viewed from some (unknown) viewing direction. Researchers [10] have proposed methods to detect and compute the symmetry correspondence of skewed mirror symmetric curves. In this paper, we consider the more general case, which is the reconstruction of symmetric curves that are not necessarily planar. Schmidt et al. [11] proposed an interactive design tool to create 3D models composed of curves. The user defines 3D constraints and uses these constraints to create complex curve networks. The system is designed for aiding single view sketching. Bae et al. [12] proposed another system for sketching 3D curves. Their contribution is a user-friendly interface with new features such as a widget for curve manipulation and the automatic view rotation. For these two systems, the modeling is done incrementally using different viewpoints and different types of widgets. Our system is more faithful to the principle of sketching. The user draws models with our sketching interface as they would do with a paper and a pen.

Li et al. [13] proposed a computational model that uses planarity and compactness constraints to recover 3D symmetric objects from 2D images. They assume known correspondence of symmetric points. Their method has been successfully applied to the reconstruction of models composed of curves. Compared to their work, our contribution is a method to compute the symmetry correspondence. Melkemi et al. [14] [15] have proposed a method to compute the symmetry relationship between two curves. They did not consider the case of multiple curves.

The closest work to ours is that of Öztireli et al. [16]. They proposed an algorithm to reconstruct a 3D model from a set of planar curves using the symmetry assumption and the orthogonal projection. The user has to select a pair of symmetric curves to define the orientation of the symmetry. The symmetry relationship is then computed. If the reconstruction is not possible because of the ambiguity of the sketch, more user intervention is required. In contrary to this previous work, our method does not require user intervention and is able to process ambiguous sketches automatically.

3. Overview

Our system takes a user's sketch composed of a set of 2D curves and determines the 3D curves whose orthogonal projection matches the input sketch. The set of reconstructed curves may be composed of pairs of symmetric curves (C_4 and C_5 in Fig. 1(a)), self-symmetric curves (C_2 in Fig. 1(a)) and curves with no symmetry (C_1 in Fig. 1(a)). Mirror-symmetry is defined as invariant under reflection with respect to a plane (referred as symmetry plane); C_i is mirror-symmetric to a curve C_j, if the curve C_j has the same position as C_j upon undergoing a reflection. Similarly, a curve C_i which does not change upon undergoing a reflection is self-symmetric. Non-symmetric curves are located in the symmetry plane.

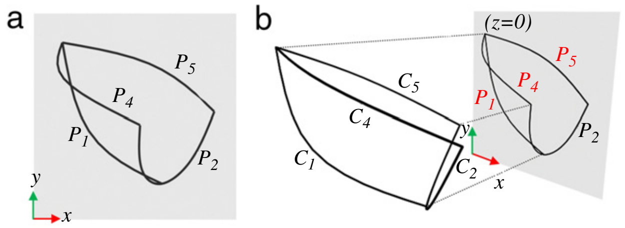


Fig. 1. In (a), the 2D curves are drawn by the user on the sketching plane ($z = 0$); P_2 is the projection of a self-symmetric curves; P_4 and P_5 are the projection of a pair of symmetric curves; P_1 is the projection of a non-symmetric curve. In (b), the 3D curves C_1, C_2, C_3, C_4 and C_5 are reconstructed such that their orthogonal projection matches the input 2D curves.

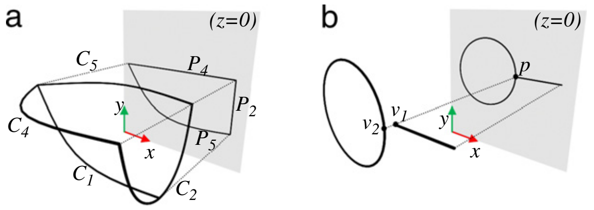


Fig. 2. Orthogonal projections that do not comply with the generic viewpoint assumption: in (a), the 3D curves C_4 and C_5 are projected onto the same 2D curve P_4. In (b), the vertices v_1 and v_2 are projected onto the same vertex p.

Other types of symmetry are defined as invariant under other transformations such as translation, rotation, etc. These types of symmetry are not considered in our approach.

3.1. *Assumptions*

The user draws the 2D curves on the plane ($z = 0$) that we call the sketching plane. These curves are polygonal curves; a polygonal curve is specified by a sequence of vertices so that the curve is composed of the line segments connecting the consecutive vertices. These curves are the orthogonal projection of the 3D polygonal curves onto the sketching plane ($z = 0$) (Fig. 1(b)). This implies that the x and y coordinates of the 3D vertices of the curves are known. The z-coordinates have to be computed.

We assume the view of the sketch to be generic. The "generic view" assumption states that the view is not accidental. The idea is that accidental views are unstable, that is, small changes in the orientation of the projection plane would generate large changes in the projected image. The generic viewpoint assumption favors the sketch interpretations which are stable with respect to small changes of the orientation of the projection plane. This assumption is commonly used in the domain of 3D reconstruction and computer vision [17].

Let C_i and C_j be two 3D curves with different shape and P_i and P_j be their orthogonal projection respectively. If the orthogonal projection complies with the generic viewpoint assumption, then the projected curves P_i and P_j must have different shape (Fig. 2(a)). The generic viewpoint assumption applies to curve points as well. Let v_i be an endpoint of a 3D curve and v_j be a point along a 3D curve. If the viewpoint is generic and v_i and v_j have different location, their projection must have different location as well (Fig. 2(b)).

The generic viewpoint assumption implies that 2D curves connected at their endpoint are the projection of 3D curves that are also connected at their endpoints. This property on curve connectivity is used in our algorithm to resolve ambiguities in the sketch.

Another assumption is that the set of 3D curves reconstructed from the input sketch is mirror-symmetric. We also assume that the symmetry relationship can be found without changing the topology of the curves. This means that the process of finding pairs of symmetric curves does not require cutting or gluing curves together.

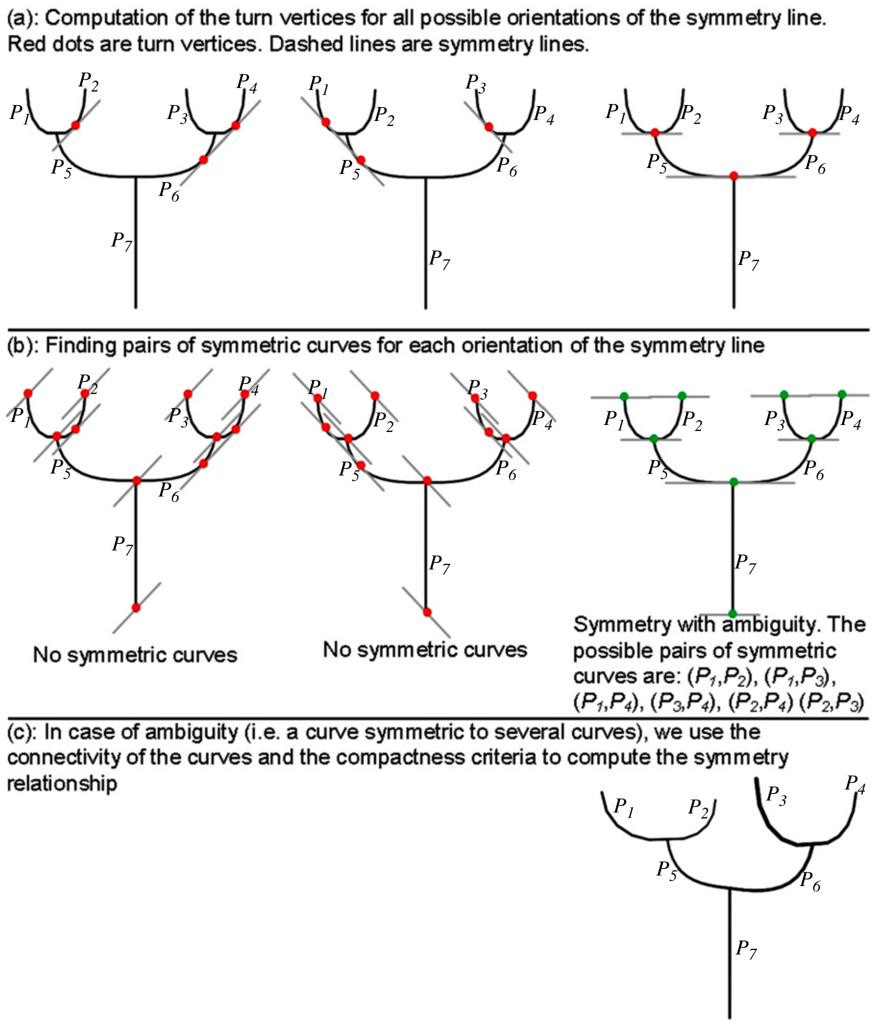

(a): Computation of the turn vertices for all possible orientations of the symmetry line. Red dots are turn vertices. Dashed lines are symmetry lines.
P1 P2 P3 P4 P5 P6 P7
(b): Finding pairs of symmetric curves for each orientation of the symmetry line
No symmetric curves
No symmetric curves
Symmetry with ambiguity. The possible pairs of symmetric curves are: (P1,P2), (P1,P3), (P1,P4), (P3,P4), (P2,P4) (P2,P3)
(c): In case of ambiguity (i.e. a curve symmetric to several curves), we use the connectivity of the curves and the compactness criteria to compute the symmetry relationship

Fig. 3. Overview of the approach.

3.2. Overview of the approach

As we will explain in Section 4, the computation of the 3D positions of a point and its mirror image is possible if we know the 2D position of their orthogonal projection onto the sketching plane ($z = 0$). Similarly, it is possible to compute whether two polygonal curves are the projection of a pair of 3D polygonal curves that are symmetric to one another.

In case of a sketch composed of multiple curves, the computation of the symmetry relationship becomes more complicated. One difficulty is to identify the orientation of the symmetry from the sketch. Another difficulty is the ambiguity that may arise in some sketches. Ambiguity appears when the symmetry relationship is not uniquely defined (see Fig. 3).

Our approach is composed of the following steps. First, we compute all the possible orientations of the symmetry lines. Symmetry lines are lines that connect pairs of mirror-symmetric vertices. Two vertices are mirror symmetric if they are invariant under reflection. The properties of mirror-symmetric vertices are described in Section 4. Then, we compute the turn vertices of the polygonal curves for each possible orientation of the symmetry line. A vertex v is a turn vertex if its two neighbors are in the same half-plane delimited by the symmetry line passing through v (see Fig. 3(a)). The computation of the turn vertices is described in Section 5.1.

The next step is to compute the symmetry relationship for each candidate of the symmetry orientation. Two curves are the projection of symmetric curves if they have the same number of turn vertices and there exists a one-to-one matching between the vertices and line segments of the two curves (see Fig. 3(a)). A method to compute the symmetry relationship using turn vertices is given in Section 5.2.

For some sketches, the symmetry relationship is not uniquely defined (Fig. 3(c)). A one-to-one symmetry correspondence could not be found using the turn vertices; a curve has more than one candidate for the symmetry relationship. The last step of our algorithm is composed of two methods to overcome the ambiguity problem. First, we use the connectivity among the curves to identify the symmetry relationships that are not valid. This method is described in Section 6.1. If the ambiguity remains, the symmetry relationship is defined such that the compactness of the reconstructed model is maximized (Section 6.2).

4. Mirror symmetry

Let v_i be a vertex with coordinates (x_i, y_i, z_i) and v'_i be a vertex with coordinates (x'_i, y'_i, z'_i). v'_i is the mirror symmetric of v_i. We assume that v_i and v'_i do not have same coordinates. Let M be the symmetry plane whose normal vector $\vec{N}$ is a unit vector with components (x_N, y_N, z_N). Without loss of generality, we assume that the symmetry plane passes through the origin of the coordinate system. p_i and p'_i are the orthogonal projection of v_i and v'_i onto the plane ($z = 0$). Their coordinates are $(x_i, y_i, 0)$ and $(x'_i, y'_i, 0)$ respectively. Thus, the 3D reconstruction comes down to computing the z-coordinates of the two vertices. As explained in [8], if the

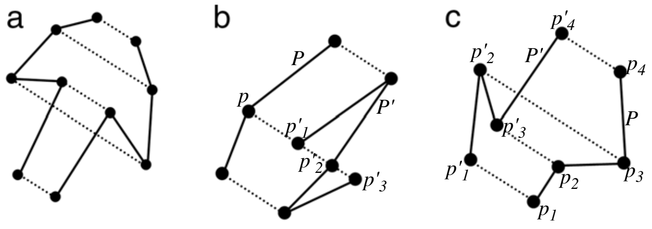


Fig. 4. In (a), the two polygonal curves comply with Proposition 2; they are the orthogonal projection of mirror-symmetric curves. In (b), the vertex $p \in P$ has more than one symmetric counterpart on P'. In (c), $p'_1 \in P'$ is the symmetric vertex of $p_1 \in P$. The vertices p_2 and p'_2 which are adjacent to p_1 and p'_1 respectively, are not symmetric to each other. Dashed lines are symmetry lines and black dots are the curve vertices.

components (x_N, y_N, z_N) of the normal vector of M are known, the z-coordinates of the two vertices and are given by Eqs. (1) and (2):

$$z_i = -\frac{1}{2}\left(\frac{x_N(x'_i + x_i)}{z_N} + \frac{y_N(y'_i + y_i)}{z_N} + \frac{z_N(y'_i - y_i)}{y_N}\right) \tag{1}$$

$$z'_i = -\frac{1}{2}\left(\frac{x_N(x'_i + x_i)}{z_N} + \frac{y_N(y'_i + y_i)}{z_N} - \frac{z_N(y'_i - y_i)}{y_N}\right). \tag{2}$$

The z-coordinates are also given by Eqs. (3) and (4):

$$z_i = -\frac{1}{2}\left(\frac{x_N(x'_i + x_i)}{z_N} + \frac{y_N(y'_i + y_i)}{z_N} + \frac{z_N(y'_i - y_i)}{x_N}\right) \tag{3}$$

$$z'_i = -\frac{1}{2}\left(\frac{x_N(x'_i + x_i)}{z_N} + \frac{y_N(y'_i + y_i)}{z_N} - \frac{z_N(y'_i - y_i)}{x_N}\right). \tag{4}$$

If a point v_i has no mirror image (v_i and v'_i have the same location), it is located on the symmetry plane. Its z-coordinate is given as follows:

$$z_i = -\left(\frac{x_N x_i}{z_N} + \frac{y_N y_i}{z_N}\right). \tag{5}$$

Given that the value of z_i is computed with the two Eqs. (1) and (3) and z'_i with Eqs. (2) and (4), the computation of these values is possible if and only if z_N differs from 0 and the coordinates of v_i and v'_i satisfy the following equality:

$$(x'_i - x_i) \cdot y_N = (y'_i - y_i) \cdot x_N. \tag{6}$$

For a set of vertices being symmetric to each other with respect to the same symmetry plane, the value $\frac{x_N}{y_N}$ is constant. Therefore, Eq. (6) implies that the lines that connect pairs of symmetric vertices must be parallel to each other. These lines are referred to as symmetry lines. This gives us the following proposition.

Proposition 1. *Let there be two sets of 2D vertices $P = p_0, \ldots, p_i, p_{n-1}$ and $P' = p'_0, \ldots, p'_i, p'_{n-1}$, each vertex p'_i being the mirror image of p_i. These two sets are the orthogonal projection of the two sets of vertices V and V′, which are mirror-symmetric to each other if and only if all the symmetry lines (lines that pass through p_i and their mirror image p'_i) are parallel to each other.*

Using Proposition 1, we define a set of properties for a pair of polygonal curves P and P' which are the orthogonal projection of a pair of symmetric 3D curves.

Proposition 2. *Given a straight line l, we say that two polygonal curves P and P′ are the orthogonal projections of a pair of symmetric 3D polygonal curves if it complies with the following conditions:*

- ***Bijection.*** *The vertex p on P has a unique symmetric vertex $p' \in P'$ which is on the line parallel to l and passing through p (Fig. 4(b)).*
- ***Continuity.*** *Let p_1 and p_1 be two symmetric vertices located on P and P′ respectively. Let p_2 be a vertex adjacent to p_1 along P. Then, p_2 must have a symmetric vertex adjacent to p_1 along P (Fig. 4(c)).*

A vertex p_i is adjacent to p_j if p_i and p_j are consecutive vertices of the polygonal curve, that is, they are connected with a line segment of the polygonal curve. Note that Proposition 2 holds only with the generic viewpoint assumption. For the sake of simplicity, we say that two 2D polygonal curves P and P' are symmetric to one another if they are the orthogonal projection of two 3D curves which are symmetric to one another.

The continuity condition is explained by the fact that a one-to-one correspondence must exist between the line segments of the two polygonal curves P and P', that is, every segment of P should correspond to a segment of P'. In Fig. 4(c), the segment (p_1, p_2) of P does not have any corresponding segment on P'. Although a one-to-one correspondence exists between the vertices of P and P', the two polygonal curves are not symmetric to one another.

5. Finding the candidates for the symmetry orientation

The simplest way to find the symmetry relationship would be to search for all the curves its symmetric counterpart among the other curves. We propose a more efficient approach. We first compute all the possible orientations of the symmetry lines and sort them according to how much they may result in a valid symmetry relationship. The first orientation is the one which is most likely to be the valid one.

Our approach is based on the following observation. Let there be two curves which are symmetric to one another with respect to a symmetry line l; it follows that the lines that pass through their endpoints are parallel to l. For example, in Fig. 5, there are two lines parallel to l and passing through the two endpoints $p_{1,1}$ and $p_{2,1}$ and the two other endpoints $p_{1,4}$ and $p_{2,4}$. In case the two curves P_i and P_j are not monotone with respect to the line l orthogonal to l, these polygonal curves are decomposed into monotone polygons. Let $P_{i,1}, P_{i,2}, , P_{i,n}$ and $P_{j,1}, P_{j,2}, P_{j,n}$ be the set of the monotone polygons of P_i and P_j respectively. Since P_i and P_j are symmetric to one another, this symmetry relationship exists also between the monotone polygons of P_i and P_j: $P_{i,1}$ is symmetric to $P_{j,1}$, $P_{i,2}$ symmetric to $P_{j,2}$, etc. It follows that the lines that pass through the endpoints of a pair of symmetric monotone polygons are parallel to l. In Fig. 5, the lines passing through $(p_{1,3}, p_{2,3})$ and $(p_{1,2}, p_{2,2})$ are parallel to l. Given a line l, we first compute the endpoints of the monotone pieces of the curves. We name these endpoints "turn vertices". Next, we find how many of these turn vertices can be connected with lines parallel to l. We also find how many of the curve endpoints can be connected with lines parallel to l. We use this information to compute the maximum number of possible symmetric curves for the symmetry line l.

5.1. Finding the turn vertices

A turn vertex is a vertex such that the two adjacent vertices are located in the same half-plane delimited by l. Let p_i be a vertex and $\alpha_{i-1,1}$ and $\alpha_{i,1}$ be the two angles between $\vec{x}$ and the two segments

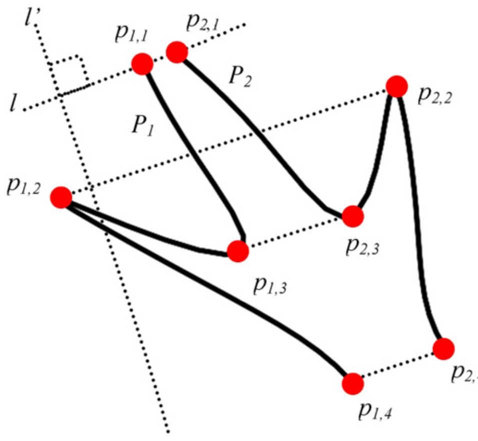


Fig. 5. The polygonal curves P_1 and P_2 are composed of three pieces, which are monotone with respect to l perpendicular to l. The turn vertices $v_{1,1}$ and $v_{1,2}$ of P_1 are connected to the turn vertices $v_{2,1}$ and $v_{2,2}$ of P_2 with lines parallel to l. Similarly, the endpoints of P_1 and P_2 are connected with lines parallel to l.

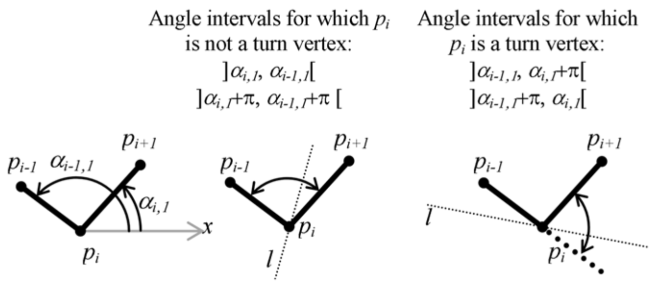

Angle intervals for which p_i is not a turn vertex:
$]\alpha_{i,1}, \alpha_{i-1,1}[$
$]\alpha_{i,1}+\pi, \alpha_{i-1,1}+\pi\,[$
Angle intervals for which p_i is a turn vertex:
$]\alpha_{i-1,1}, \alpha_{i,1}+\pi[$
$]\alpha_{i-1,1}+\pi, \alpha_{i,1}[$
p_{i-1}
$\alpha_{i-1,1}$
p_{i+1}
$\alpha_{i,1}$
x
p_i
p_{i-1}
p_{i+1}
p_i
l
p_{i-1}
p_{i+1}
l
p_i

Fig. 6. Angle intervals for which pi is a turn vertex.

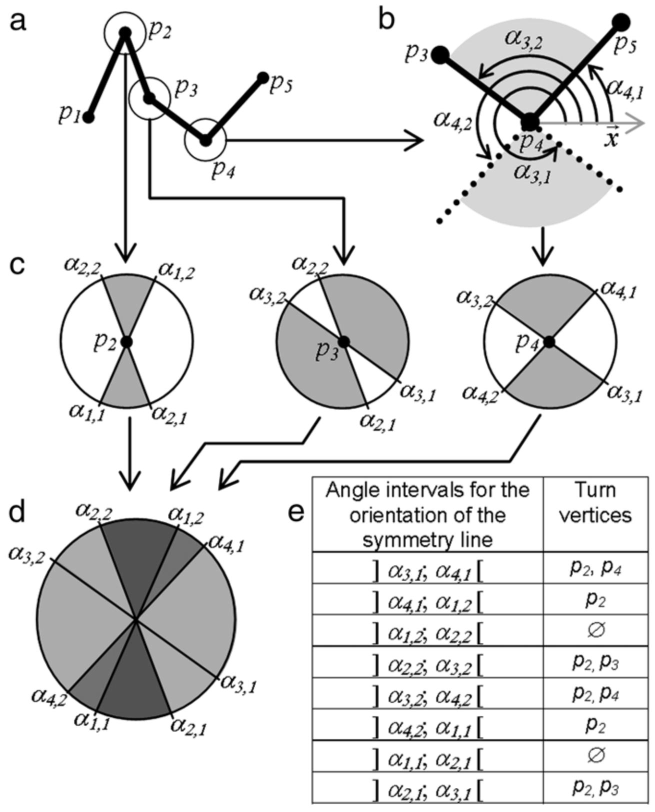
(p_{i-1}, p_i) and (p_i, p_{i+1}) connected to p_i (See Fig. 6.). Let θ be the angle between l and $\vec{x}$. p_i is a turn vertex if one of the two following inequalities is true:

$$\begin{aligned} &\alpha_{i-1,1} < \theta < \alpha_{i,2}, \quad \alpha_{i,2} = \alpha_{i,1} + \pi \\ &\alpha_{i-1,2} < \theta < \alpha_{i,1}, \quad \alpha_{i-1,2} = \alpha_{i-1,1} + \pi\,. \end{aligned} \tag{7}$$

Note that these two inequalities are equivalent, since l is not oriented. The inequalities (7) define the angle intervals for which p_i is a turn vertex (see Fig. 7(b)). Now, we consider the case of a set of vertices. For each vertex v_i, we compute the four angles $\alpha_{i-1,1}$, $\alpha_{i-1,2}$, $\alpha_{i,1}$ and $\alpha_{i,2}$, and write the corresponding inequalities (7). If all the vertices of the set are turn vertices with respect to the same line l, the inequalities (7) must be true for all the vertices. In fact, this is equivalent to computing if the intersection of the angle intervals is not empty (See Fig. 8.).

Our algorithm works as follows. We have a set of the polygonal curves as input. First, we compute the two angles $\alpha_{i,1}$ and $\alpha_{i,2}$ for each vertex v_i of the curves. Then, we sort these angle values by increasing order. Each angle value of this sorted list defines the orientation of the symmetry line for which a vertex becomes a turn vertex. It follows that the set of vertices that are turn vertices do not change when the orientation of the symmetry line is between two consecutive angle values of this sorted list. For each pair of consecutive angle values of the sorted list, we compute the corresponding turn vertices (Fig. 7(e)). As a result, we obtain a set of angle intervals with the corresponding turn vertices.

In our algorithm, we make the two following assumptions. First, we suppose that all segments of the curves have different orientation (different angle values $\alpha_{i,1}$). In other words, we suppose that the polygonal curves do not have any segment parallel to each other. If a polygonal curve is composed of a set of adjacent



Angle intervals for the orientation of the symmetry line	Turn vertices
$]\,\alpha_{3,1};\ \alpha_{4,1}\,[$	p_2, p_4
$]\,\alpha_{4,1};\ \alpha_{1,2}\,[$	p_2
$]\,\alpha_{1,2};\ \alpha_{2,2}\,[$	∅
$]\,\alpha_{2,2};\ \alpha_{3,2}\,[$	p_2, p_3
$]\,\alpha_{3,2};\ \alpha_{4,2}\,[$	p_2, p_4
$]\,\alpha_{4,2};\ \alpha_{1,1}\,[$	p_2
$]\,\alpha_{1,1};\ \alpha_{2,1}\,[$	∅
$]\,\alpha_{2,1};\ \alpha_{3,1}\,[$	p_2, p_3

Fig. 7. In (a), the polygonal curve with the three vertices. In (b), p_4 is a turn vertex if the symmetry line l that goes through p_4 does not enter the sectors shown in grey. In (c), the angle intervals for which the three vertices are turn vertices. In (d), the sorted list of angles for the three vertices is shown. The table in (e) contains the angle intervals with the corresponding turn vertices.

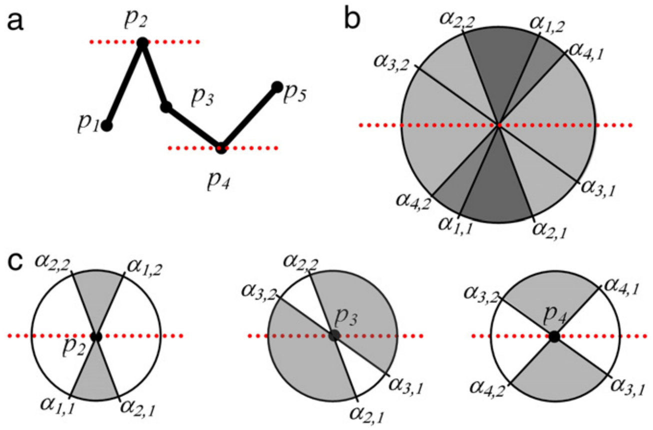


Fig. 8. Computing the turn vertices of the polygon in (a) for the horizontal symmetry line. The symmetry line is shown in a red dashed line. As shown in (b), the angle value of the symmetry line is in the interval $]\alpha_{3,1};\ \alpha_{4,1}[$. This implies that p_2 and p_4 are turn vertices. The angle intervals for p_2, p_3 and p_4 with the symmetry line in shown in (c).

segments with same orientation, we replace these segments with another whose extremities are the first endpoint of the first segment and the last endpoint of the last segment. If the curves contain segments with the same orientation and which are not adjacent, we slightly modify the position of the corresponding vertices to change the orientation of the segments.

Second, we assume that our sketch is composed of smooth polygonal curves. This implies that the angle intervals $]\alpha_{i+1,2};\ \alpha_{i,1}[$

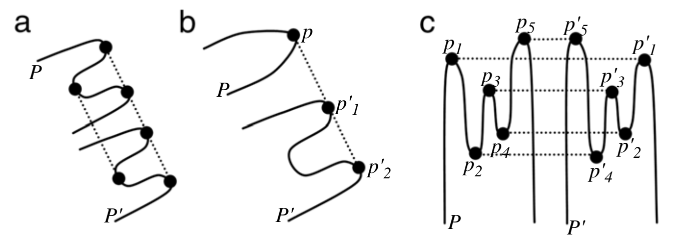


Fig. 9. In (a), the two polygonal curves comply with Proposition 3; they are the orthogonal projection of mirror-symmetric curves. In (b), the turn vertex $p \in P$ has more than one symmetric counterpart on P. In (c), $p_1 \in P$ is the symmetric vertex of $p_1 \in P$. The vertices p_2 and p_2' which are adjacent to p_1 and p_1' respectively, are not symmetric to each other. Dashed lines are the symmetry lines and black dots and the turn vertices and endpoints.

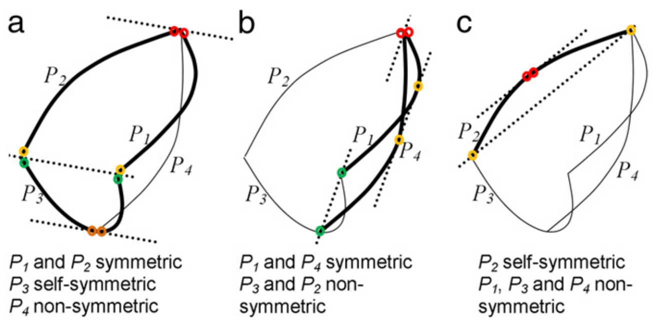


Fig. 10. Dashed lines are symmetry lines that pass through pairs of turn vertices or pairs of endpoints. Symmetric curves are shown in thick lines. Matching endpoints and turn vertices are shown with the same color. In (a), there are three symmetric curves corresponding to the orientation of the symmetry line, only two in (b), and only one in (c).

and $]\alpha_{i+1,1}; \alpha_{i,2}[$ in the inequalities (7) are small enough so that we can approximate each angle interval with one angle value (we take median value). This angle value is the orientation of symmetry line l_i corresponding to the angle interval.

As a result, we have a list of angle intervals; each angle interval is associated with a line (whose orientation is the median value of the angle interval) and the corresponding turn vertices. The list of symmetry lines corresponding to the angle intervals is denoted $L = \{l_0, l_1, , l_1, , l_n\}$. Each line l_i is associated with the corresponding turn vertices $t_{i,0}, t_{i,1}, , t_{i,m-1}$. This list contains all the possible candidates for the orientation of the symmetry lines.

5.2. Computing the maximum number of possible symmetric curves for each candidate of the symmetry orientation

Now that we have a list of candidates for the symmetry orientations, the next step is to identify those that have the highest chance of resulting in a valid symmetry relationship. For each candidate l_i, we compute the number of curves that are symmetric according to the Proposition 2. To find if a polygonal curve is symmetric to another one or self-symmetric, we write:

Proposition 3. *A polygonal curve P is symmetric to another curve P with respect to a symmetry line li if it complies with the following conditions:*

- ***Bijection.*** *Each turn vertex p on P, has a unique symmetric turn vertex $p \in P$ which is on the line parallel to l_i and passing through p. This proposition applies to endpoints as well.*
- ***Continuity.*** *Let p_1 and p_2' be two symmetric turn vertices belonging to P and P respectively. Let p_2 be a turn vertex adjacent to p_1 along P. Then, p_2 must have a symmetric turn vertex p_2' adjacent to p_1' along P.*

Similar to Proposition 2, Proposition 3 requires the generic viewpoint assumption. Note that Proposition 3 is also valid for self-symmetric curves. In this case, P and P are the same polygonal curve. An important result of Proposition 3 is that two symmetric polygonal curves must have the same number of turn vertices and there must exist a one-to-one correspondence between their turn vertices and their end-points (Fig. 9(a)). In addition, the turn vertices should match to each other in the same order along the two polygonal curves. If p_1 is symmetric to p_1', then their adjacent vertices p_2 and p_2' must be symmetric as well (see Fig. 9(c)).

Given a candidate l_i for the symmetry line and the corresponding turn vertices $\{t_{i,0}, t_{i,1}, , t_{i,m-1}\}$, we compute the number of curves that are symmetric and self-symmetric (Fig. 10). We first compute the orthogonal projection of the turn vertices and endpoints of the polygonal curves onto a line l_i which is perpendicular to l_i. We sort the projected vertices by increasing order of their coordinates. If the two vertices p_j and p_k project onto the same

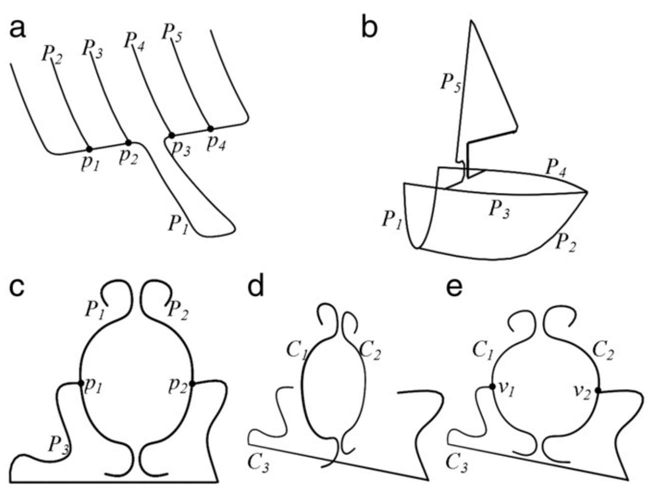


Fig. 11. In (a), P_2 to P_5 are symmetric to each other. In (b), the sketch does not correspond to a symmetric model. Its reconstruction is not possible. In (c), P_1 and P_2 seem to be symmetric to each other. C_1 and C_2 are the reconstructed curves of P_1 and P_2 respectively. If this symmetry is taken into account, the reconstruction is not correct (d). In (e), the reconstruction is made with C_1 and C_2 as non-symmetric. These two curves lie in the symmetry plane with C_3 which also not symmetric.

point on l_i', these two vertices may be symmetric. Two vertices are considered to project onto the same point if the distance between their projections is smaller than a user-defined value. By default, this value is equal to the length of the smallest line segment of the polygonal curves; the user can increase this value to handle inaccurate sketches. If we find a pair of curves whose endpoints and turn vertices match to each other in a one-to-one manner, we check if these two curves comply with Proposition 3. Once a curve has been found to match with another one, it is no longer considered for matching with other curves. We apply the same process to find self-symmetric curves.

Note that we compute the maximum number of possible symmetric curves, not the symmetry relationship. The curves that are considered as symmetric according to Proposition 3 may violate the generic viewpoint assumption (Fig. 11(c)). In some cases, a curve is found symmetric to several other curves (Fig. 11(a)). In this case, the maximum number of possible symmetric curves is the number of pairs that can be constructed from this set of curves that are symmetric to each other.

This process of computing the maximum number of possible symmetric and self-symmetric curves is calculated for all symmetry line candidates of the list $L = \{l_0, l_1, , l_1, , l_{n-1}\}$. The list L is then sorted by order of decreasing number of symmetric curves.

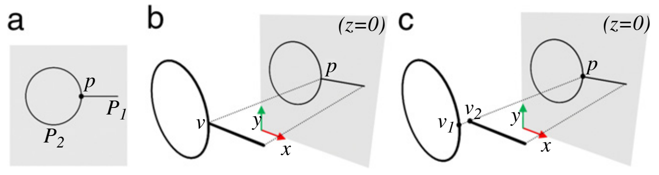


Fig. 12. The input sketch is composed of two curves P_1 and P_2; the endpoint p of P_1 is located on P_2. One possible 3D reconstruction that complies with the generic viewpoint assumption is shown in (b). The 3D reconstruction in (c) does not comply with the generic viewpoint assumption.

Symmetry lines with the highest number of possible symmetric curves will be processed first. This is because we want to avoid 3D reconstruction with high number of non-symmetric curves. Non-symmetric curves are located in the symmetry plane; 3D reconstruction with only non-symmetric curves would result into a set of curves that are coplanar with each other.

6. Computing the symmetry relationship

As mentioned in Section 5.2, the Proposition 3 is usually not sufficient to define the symmetry relationship. The first problem is that we rarely obtain exactly a one-to-one correspondence among the symmetric curves because of the ambiguity of the sketch (Fig. 11(a)). In such case, we have a set of more than two curves which are all symmetric to each other according to Proposition 3. Another problem is that two curves that are symmetric according to Proposition 3 are not necessarily symmetric (Fig. 11(c)). There are also sketches that do not represent symmetric models and therefore, cannot be reconstructed using the symmetry relationship (Fig. 11(b)).

To solve this problem, we propose an algorithm to define the symmetry relationship which exploits the curve connectivity. For cases where the ambiguity has not been fully resolved, we propose another algorithm which maximizes the compactness of the reconstructed curves.

These algorithms are applied for each candidate of the symmetry orientation. We start with l_0, the first candidate for the orientation of the symmetry line and iterate over the elements of $L = \{l_0, l_1, , l_1, , l_{n-1}\}$ until we have computed the symmetry relationship with the highest number of symmetric curves.

In the remaining of the paper, we denote P, the set of all the curves of the sketch. Given a candidate l_i of the symmetry orientation, we compute two sets. P_C is the set of polygonal curves for which the symmetry relation has been uniquely defined using Proposition 3. This includes the curves for which no symmetry has been found, the curves that are symmetric to only one curve, and the curves that are symmetric only to themselves. We compute P_N, the set of the remaining curves (i.e. $P = P_C \cup P_N$); these are the curves that are symmetric to several other curves according to Proposition 3.

6.1. Exploiting the curve connectivity

We assume the generic viewpoint. This implies that the reconstruction of 3D curves is computed in a way that the viewpoint is generic (Fig. 12). This assumption enables us to use the curve connectivity to resolve ambiguities in the sketch.

We now have the following proposition.

Proposition 4. *Let two polygonal curves P_i and P_j be in the sketching plane. C_i and C_j are the reconstructed curves of P_i and P_j respectively. If an endpoint of P_i is located along P_j, then the corresponding endpoint of C_i is also located along C_j.*

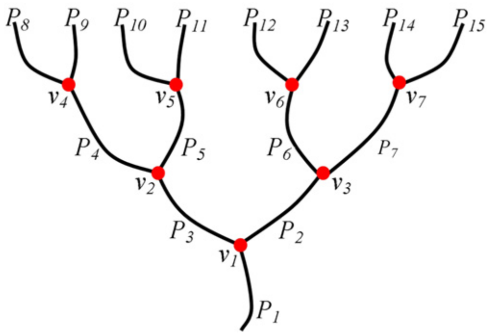


Fig. 13. The polygonal curves P_4 to P_7 are all symmetric to each other according to Proposition 3; the same problem occurs for curves P_8 to P_{15}. We use their connectivity (red dots) to compute the symmetry relationship.

Proposition 4 is used for two purposes. First, we check if the symmetry relationship of the curves of P_C calculated with Proposition 3 is valid. If not, the algorithm tries to find a valid symmetry relationship by changing symmetric curves into non-symmetric ones. Second, we use Proposition 4 to find the one-to-one symmetry relationship among the curves of P_N. P_N are the set of curves that have several possible symmetric counterparts.

Checking the validity of the symmetry relationship of the curves of P_C. Two curves which are symmetric according to Proposition 3 may violate Proposition 4 (Fig. 11(c)). To check the validity of the symmetry relationship of the curves of P_C, we find all the connected curves of P_C (curves whose endpoint is located onto another curve). We then compute the 3D reconstruction of the connected curves of P_C using Eqs. (1)–(5) and check if the Proposition 4 is violated. If so, we change one of the symmetric curves into a non-symmetric curve. This process is repeated for all the connected curves of P_C. In the worst case, all the symmetric curves become non-symmetric and the number of symmetric curves becomes zero.

Finding of the symmetry relationship of the curves of P_N. P_N is the set of curves which have more than one symmetric counterpart; the symmetry relationship could not be uniquely defined. We use Proposition 4 to compute the one-to-one symmetry correspondence among these curves.

Our algorithm finds the symmetry relationship iteratively by checking all possible combinations of pairs of curves and non-symmetric curves from the set P_N. At each iteration, we arbitrarily define which curves of P_N are symmetric or self-symmetric and those which are not symmetric. Then, we compute their 3D positions using Eqs. (1)–(5) and we check if they comply with Proposition 4. Since a curve can be in three different states (symmetric, self-symmetric, not symmetric), the complexity of the algorithm is 3^n in the worst case, n being the number of curves of P_N. Although the time complexity is high, the computation time remains low because each iteration involves computing a small number of equations.

6.2. Maximizing the compactness of the reconstructed curves

For some sketches, Proposition 4 is not sufficient to uniquely define the symmetry relationship. Ambiguity occurs in three different cases. The first case concerns the curves whose both endpoints share the same position and that comply with Proposition 3 (Fig. 14(a)). The second case concerns the curves whose one endpoint shares the same position. The other endpoints are aligned along the symmetry line (see Fig. 14(b)). The third case is the curves which comply with the Proposition 3 and whose endpoints do not have the same position (see Fig. 14(c)).

Our algorithm is based on the work of Li et al. [13]. They have shown that human beings interpret ambiguous sketches in a way

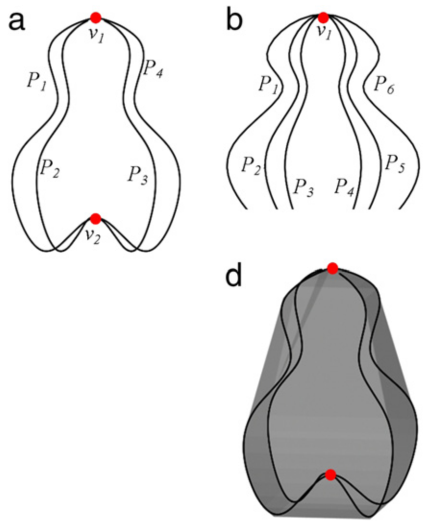


Fig. 14. In (a), (b) and (c), the three cases for which the symmetry relationship is computed using the compactness criteria. In (d), we compute the symmetry relationship such that convex hull of the reconstructed curves is the most compact.

to maximize the compactness of the reconstructed shape. The compactness of a shape O is defined as follows:

$$C(O) = \frac{V(O)^2}{S(O)^3}. \tag{8}$$

$V(O)$ and $S(O)$ are the volume and the surface of shape respectively. In our case, the reconstructed shape is composed of 3D curves; we compute the compactness ratio of the convex hull of the 3D curves; $V(O)$ and $S(O)$ are the volume and the surface of the convex hull respectively.

We first identify a subset of curves from the set P_N which are all symmetric to each other with respect to the Proposition 3. P_N is the set of curves for which the symmetry relationship could not be uniquely defined using Proposition 3 (see Section 6.1). Then, we arbitrarily define the symmetry relationship among these curves (by arbitrarily choosing which one is symmetric or self-symmetric and which one is not symmetric). This enables us to compute the 3D positions of the curves using Eqs. (1)–(5), and then the corresponding convex hull and its compactness. This process is repeated for all possible pairs of symmetric curves and non-symmetric curves from the set. In the end, we choose the symmetry relationship whose compactness is the highest.

In some cases, this method does not uniquely define the symmetry relationship for all the curves. This case happens when all the internal vertices of some curves (vertices other than the endpoints) are not part of the convex hull. The shape of the convex hull remains the same however the symmetry relationship is defined among these curves. The solution to this problem is to find these curves that are not part of the convex hull and repeat the whole process to compute their symmetry relationship.

6.3. The criterion to find the best symmetry relationship

The symmetry relationship is calculated (Sections 6.1 and 6.2) iteratively for each element of the list $L = \{l_0, l_1, , l_1, , l_{n-1}\}$ of the candidates for the symmetry orientation, starting from l_0. For each candidate l_i, we know the maximum number of possible symmetric curves (Section 5.2) and these candidates are sorted by decreasing order of this number. We stop the iterations over L when the number of symmetric curves that has been found for l_i while computing the symmetry relationship (Sections 6.1 and 6.2) is larger than the maximum number of possible symmetric curves for the next candidate l_{i+1}. The goal is to find the symmetry orientation with the highest number of symmetric curves. If there are several symmetry orientations with same number of symmetric curves, we select the one whose compactness is the largest.

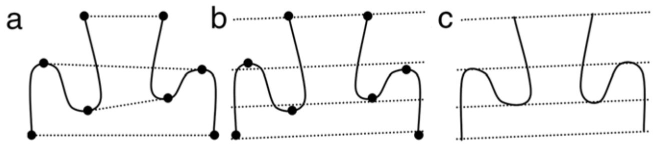


Fig. 15. Given a set of symmetric curves (a), we compute the symmetry lines that pass as close as possible through pairs of symmetric vertices (b). The coordinates of curve vertices are then modified such that turn vertices and endpoints are located on the symmetry lines (c). Dashed lines are symmetry lines and black dots are the turn vertices and endpoints.

7. 3D reconstruction

Once the symmetry relationship has been found, we compute the 3D reconstruction of the curves by using Eqs. (1)–(5). Prior to this step, we compute an accurate approximation of the orientation of the symmetry line. We also resample the curves such that the symmetry relationship is defined at the vertex level.

7.1. Orientation of the symmetry lines

Because sketches are drawn by hand, the symmetry lines that pass through pairs of symmetric vertices are never exactly parallel to each other. Up to this stage of the algorithm, the orientation of the symmetry line is taken as the median value of the corresponding angle interval (Section 5.1). Since the symmetry relationship is completely defined, it is now possible to compute the orientation of the symmetry lines more accurately. The idea is to compute the set of parallel lines that pass as close as possible through pairs of turn vertices and endpoints. We do this using the linear regression method. The equation of a straight line is:

$$y = ax + b. \tag{9}$$

The coefficient a is the slope of the line; this value is identical for all symmetry lines (all symmetry lines are parallel to each other). The coefficient b depends on the coordinates of the pairs of symmetric turn vertices and endpoints. Let $P = \{p_0, \ldots, p_i, \ldots, p_{n-1}\}$ and $P' = \{p'_0, \ldots, p'_i, \ldots, p'_{n-1}\}$ be two sets of turn vertices and endpoints, each vertex p'_i being symmetric to p_i. The coordinates of p_i and p'_i are (x_i, y_i) and (x'_i, y'_i) respectively. The lines that pass through pairs of symmetric vertices give the following set of equations:

$$\begin{bmatrix} x_0 & 1 & 0 & \ldots & 0 \\ x'_0 & 1 & 0 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ x_{n-1} & 0 & 0 & \ldots & 1 \\ x'_{n-1} & 0 & 0 & \ldots & 1 \end{bmatrix} \cdot \begin{bmatrix} a \\ b_0 \\ \ldots \\ \ldots \\ b_{n-1} \end{bmatrix} = \begin{bmatrix} y_0 \\ y'_0 \\ \ldots \\ y_{n-1} \\ y'_{n-1} \end{bmatrix}. \tag{10}$$

Since this system is over-determined (number of equations is larger than the number of unknowns), we compute an approximate solution with the least squares method. This enables us to compute the orientation of the symmetry line (Fig. 15(b)). Note that this method works best when the slope of the line is close to zero. Before estimating the orientation of the symmetry lines, we apply a rotation to the curve vertices so as to make the symmetry lines parallel to the x-axis.

Next, we modify the shape and position of the curves so that the turn vertices and endpoints are located on the symmetry lines (Fig. 15(c)). Turn vertices and endpoints are translated to their corresponding position on the symmetry line. The translation of the other vertices is defined as a linear interpolation of the translations of the two adjacent endpoints and/or turn vertices.

7.2. Resampling of the curves

Given two symmetric curves, we first decompose these curves into monotone pieces by cutting them at their turn vertices. The result is a set of pairs of monotone pieces which are symmetric to one another. Each pair is then resampled separately such that symmetry relationship is defined at the vertex level.

7.3. 3D reconstruction

Once the curve resampling is complete, the 3D positions of the curve vertices are calculated using Eqs. (1)–(5). These equations require the value z_N of the normal vector of the symmetry plane. This value is defined in a way that the compactness of the convex hull of the reconstructed model is maximal. As mentioned in Section 5, we measure the compactness of the reconstructed model with the function $V(O)^2/S(O)^3$; $V(O)$ and $S(O)$ are the volume and the surface of the convex hull of the curve points respectively. It can be shown from the Eqs. (1)–(5) that changing the value of z_N is equivalent to applying a scale transformation to the 3D curves along the axis orthogonal the symmetry plane. In addition, the vertices belonging to the convex hull do not change when a scale transformation is applied to these vertices. It follows that the compactness function is a rational function composed of polynomial functions and square root of polynomial functions, these functions having z_N as a variable. This compactness function has only one maximum, which is computed using the gradient descent method. The compactness maximization method always gives a solution. If the user is not satisfied with the result, this means that the scale of the reconstructed model along the line perpendicular to the symmetry plane is not correct. For instance, the user draws a rectangular parallelepiped and the reconstructed model looks more like a cube. In this case, the user can modify the z_N value.

8. Results, limitations and conclusion

Our sketch-based modeling tool has been implemented as a plug-in to Maya. The steps to create curves in Maya are as follows. The front view is first selected so that curves are created in the plane ($z = 0$). Then, the user draws the curves by placing a set of points in the front view. An alternative way is to use the pencil tool with which the user creates a curve by dragging the mouse. Once all the curves are created, the user clicks an icon to launch our plug-in. If the input curves are Bezier curves, they are automatically converted into polygonal curves. The 3D reconstruction is then computed and the resulting 3D polygonal curves are shown in the 3D viewer of Maya.

Our method is demonstrated with several examples corresponding to different cases of sketching, showing its versatility. Some examples have been created by users (Fig. 19). The teapot model has been created in several steps. The user draws some parts of the model; the 3D reconstruction is computed; then the user rotates the model and continues sketching. We have also tested our system with models created with existing software [12] (Fig. 20). To do this, we first compute the orthogonal projection of the 3D curves of an existing model onto a plane. These projected curves are then used as input to our system to recreate the original model. These sketches generated from existing models are composed of several dozens of curves; they are far more complex than what a user can draw. The purpose is to demonstrate the robustness of our approach. The computation time required to generate the 3D models ranges from a few seconds to about thirty seconds depending on the number of curves that compose the hand drawn sketch. This computation time becomes slow for models that require computing the convex hull (see Section 6.2).

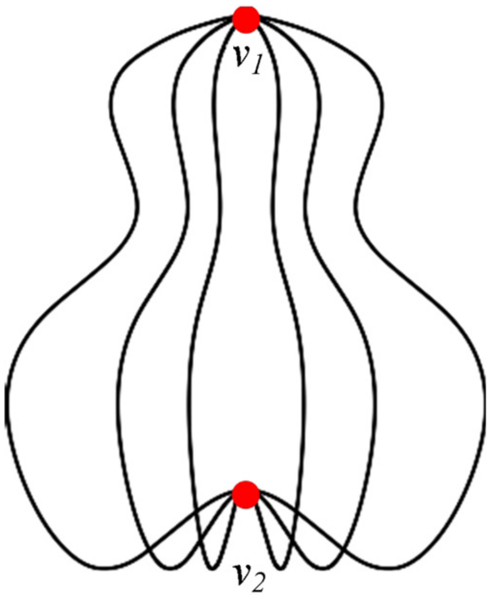


Fig. 16. All curves are connected at their endpoint. The fact that the user has selected two curves as being symmetric does not give any clue for finding the symmetry relationship of other curves.

One may think that the reconstruction process could be done simply by letting the user select two symmetric curves. The orientation of symmetry plane is then determined by using these two symmetric curves and all the curves are reconstructed in 3D. However, some models exhibit inherent ambiguity; the selection of two symmetric curves is not sufficient to find the symmetry relationship for other curves (see Figs. 13, 15 and 16). To reconstruct such model, the user would have to define all the pairs of symmetric curves. The main advantage of our method over existing approaches is that it finds the symmetry relationship without user intervention by using the curve connectivity and maximizing the compactness of the reconstructed model.

8.1. Comparison with existing methods

In the system presented by Bae et al. [12], the modeling process is done iteratively. To create a model such as those shown in Fig. 20, the user has to alternate between drawing the curves and changing the viewpoint. In addition, the snapping feature has been integrated; this helps the user to create accurate and complex sketches. In our approach, the reconstruction process is done in a single step. The user provides a set of planar curves and these curves are processed all together at once. Compared to the work of Bae et al. [12], our method is more faithful to the paradigm of sketching. Our method is quite sensitive to the quality of the sketches; this drawback could be overcome by integrating the snapping feature into our system.

The system presented by [8] aims at reconstructing 3D symmetric shapes from a sketch. Their system takes as input a hand-drawn sketch and automatically generates a surface whose silhouette matches the input sketch. Similarly to our approach, one important step of their approach is to detect the symmetry relationship among the 2D curves of the sketch. The major difference is that their method is limited to the detection of skewed mirror symmetry and translational symmetry. Skewed-mirror symmetry depicts a mirror-symmetric planar curve viewed from some (unknown) viewing direction. This limits the method to the reconstruction of shapes whose 3D silhouette curves are planar and lie in the same plane.

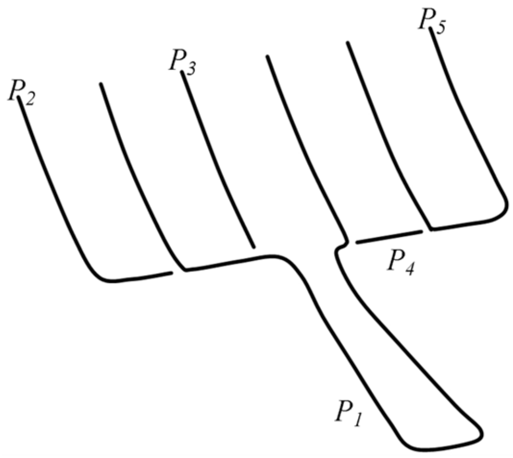


Fig. 17. The topology of the curves does not correspond to the symmetry of the model.

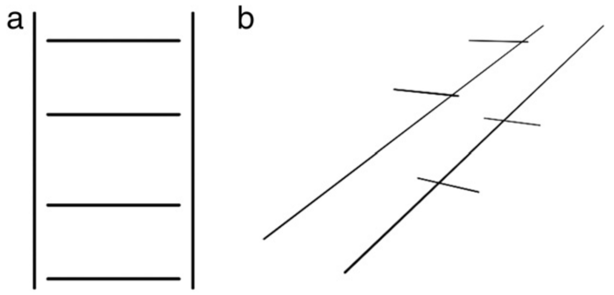


Fig. 18. (a) The ladder is composed of disconnected polygonal curves; (b), The 3D reconstruction generated by our algorithm.

8.2. Limitations

The main limitation is that we assume that the topology of the curves does not need to be modified. For instance, the reconstruction from the sketch in Fig. 17 is not possible because the topology of the curves does not reflect the symmetry of the model.

Another limitation is that our method heavily relies on the connectivity of the curves to find the symmetry relationship. In particular, our algorithm searches for curves whose endpoints share the same location and uses this information to compute the symmetry relationship. If the sketch is composed of disconnected polygonal curves, our method computes the symmetry relationship that maximizes the number of pairs of symmetric curves and the compactness of the reconstructed model. This may lead to unexpected results as shown in Fig. 18.

8.3. Future work

Our system can be extended in different ways. One extension could be to generate a smooth surface that passes through the 3D curves. An approach similar to that of [18] could be implemented. For a given set of curves, this approach automatically constructs a surface embedding. The shape of the surface is obtained by applying an optimization method that minimizes the curvature variation. Öztireli et al. [16] have also used a technique to generate a smooth surface from points. Given a set of points with depth, they compute a depth map function corresponding to the surface passing through the vertices. This depth map function consists of radial basis functions. Another way to extend our system would be to compute the 3D reconstruction with perspective projection. One main difference with respect to orthogonal projection is that the perspective projection of a set of parallel lines is a set of lines that meet at a point in the projection plane. It follows that the

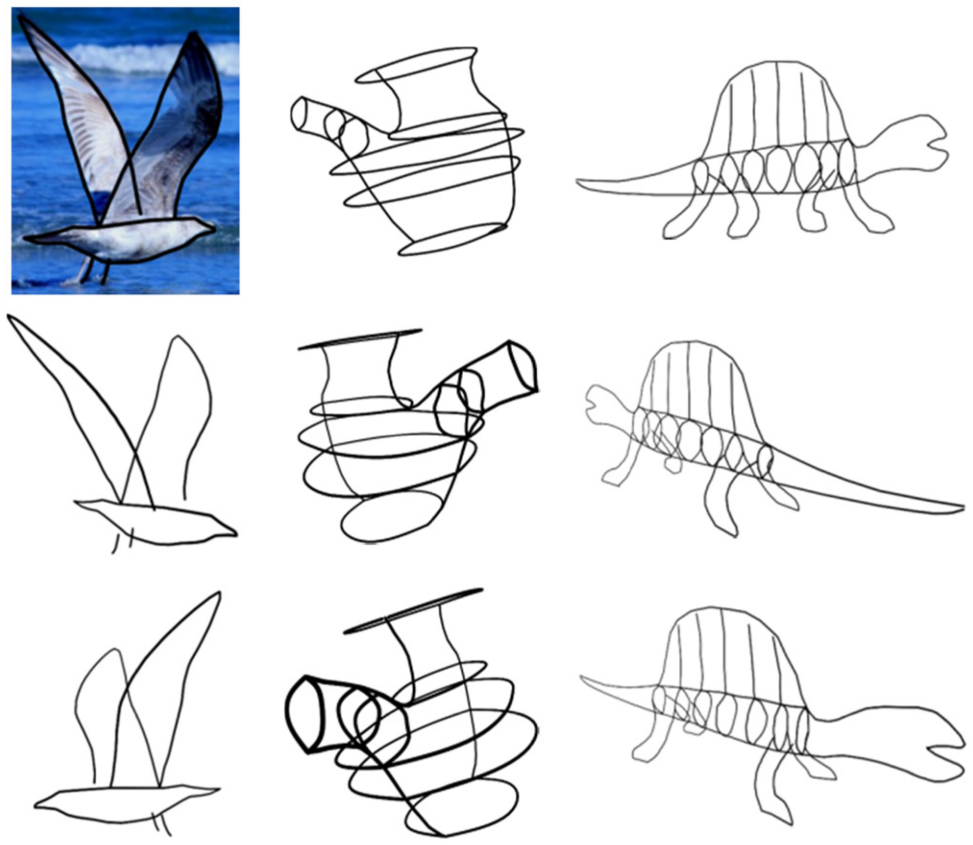


Fig. 19. Examples of sketches. The first row shows the input sketches. The two others rows show the 3D models from different viewpoints.

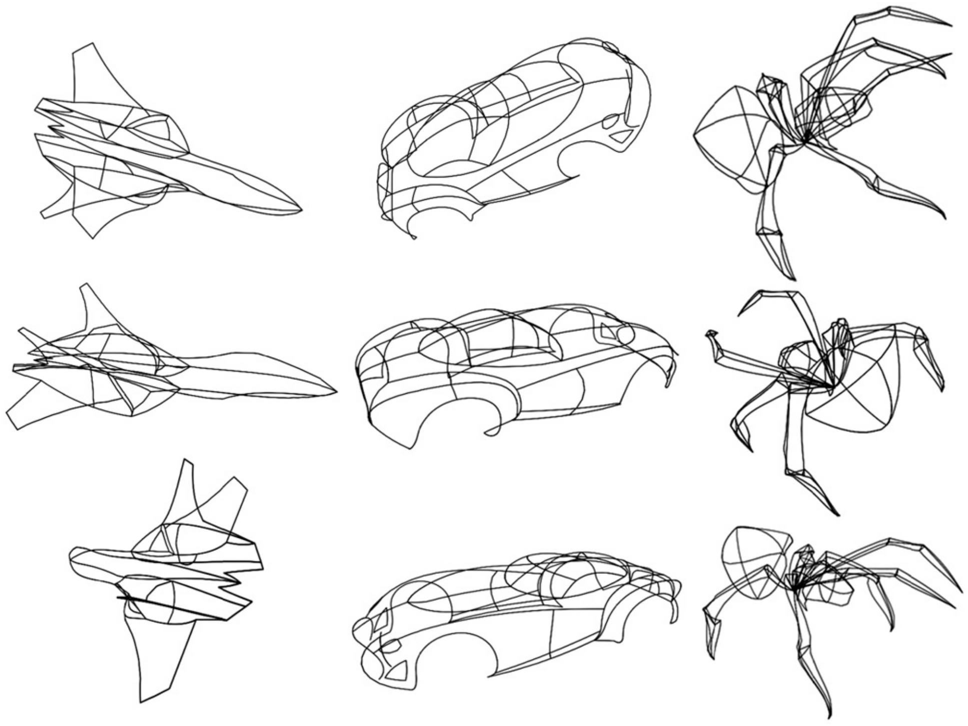


Fig. 20. The first row shows the input sketches. The two others rows show the 3D models from different viewpoints.
Source: Theses sketches are from Bae et al. [12].

symmetry lines are no more parallel; instead, they intersect each other at one point.

References

[1] Schmidt R, Khan A, Kurtenbach G, Singh K. On Expert Performance in 3D Curve-Drawing Tasks. SBM; 2009. 133–140.

[2] Brown E, Wang P. 3D object recovery from 2D images: A new approach. SPIE Proc. Robotics and Computer Vision, vol. 2904. 1996. p. 138–45.

[3] Shoji K, Kato K, Toyama F. 3-D interpretation of single line drawings based on entropy minimization principle. Proc. IEEE Conf. Computer Vision and Pattern Recognition, vol. 2. 2001. p. 90–5.

[4] Liu J, Cao L, Li Z, Tang X. Plane-based optimization for 3D object reconstruction from single line drawings. IEEE Transactions on Pattern Analysis and Machine Intelligence 2008;30(2):315–27.

[5] Igarashi T, Matsuoka S, Tanaka H. Teddy: A sketching interface for 3D freeform design. SIGGRAPH 1999;409–16.

[6] Karpenko O, Hughes J. SmoothSketch: 3D free-form shapes from complex sketches. ACM Transaction on Graphics 2006;25(3):589–98.

[7] Cordier F, Seo H. Free-form sketching of self-occluding objects. IEEE Computer Graphics and Applications 2007;27(1):50–9. Special issue on Sketching.

[8] Cordier F, Seo H, Park J, Noh JY. Sketching of mirror-symmetric shapes. IEEE Transactions on Visualization and Computer Graphics 2011;17(11):1650–62.

[9] Cohen J, Markosian L, Zeleznik R, Hughes J, Barzel R. An interface for sketching 3D curves. ACM I3DG 1999 Symposium on Interactive 3D Graphics 1999; 17–21.

[10] Shen D, Ip H-S, Teoh EK. Robust detection of skewed symmetries by combining local and semi-local affine invariants. Pattern Recognition 2001; 34(7):1417–28.

[11] Schmidt R, Khan A, Singh K, Kurtenbach G. Analytic drawing of 3D scaffolds. ACM Transactions on Graphics 2009;28(5).

[12] Bae S-H, Balakrishnan R, Singh K. ILoveSketch: As-natural-as-possible sketching system for creating 3D curve models. In: The Proceedings of ACM Symposium on User Interface Software and Technology 2008. CA, USA: Monterey; 2008. 151–160.

[13] Li Y, Pizlo Z, Steinman RM. A computational model that recovers the 3D shape of an object from a single 2D retinal representation. Vision Research. 2009; 49(9):979–91.

[14] Melkemi M, Cordier F, Sapidis NS. An algorithm to detect the weak-symmetry of a simple polygon. ICIAR 2011;1:365–74.

[15] Melkemi M, Cordier F, Sapidis NS. A provable algorithm to detect the weak-symmetry of a polygon, International Journal of Image and Graphics, World Scientific (in press).

[16] Öztireli AC, Uyumaz U, Popa T, Sheffer A, Gross MH. 3D Modeling with a Symmetric Sketch. SBM; 2011. 23–30.

[17] Freeman WT. The generic viewpoint assumption in a framework for visual perception. Nature 1994;368:542–5.

[18] Nealen A, Igarashi T, Sorkine O, Alexa M. FiberMesh: Designing Freeform Surfaces with 3D Curves. In: ACM Transactions on Computer Graphics. San Diego, USA: ACM SIGGRAPH 2007; 2007.