Patterns from Photograph: Reverse-Engineering Developable Products

Amélie Fondevilla, Adrien Bousseau, Damien Rohmer, Stefanie Hahmann, Marie-Paule Cani

Patterns from Photograph: Reverse-Engineering Developable Products

Amélie Fondevilla[a], Adrien Bousseau[b], Damien Rohmer[a,c], Stefanie Hahmann[a], Marie-Paule Cani[a]

[a] *Univ. Grenoble Alpes, CNRS (LJK), Inria*
[b] *Inria, Université Côte d'Azur*
[c] *Univ. Lyon, CPE Lyon*

Abstract

Developable materials are ubiquitous in design and manufacturing. Unfortunately, general-purpose modeling tools are not suited to modeling 3D objects composed of developable parts. We propose an interactive tool to model such objects from a photograph. Users of our system load a single picture of the object they wish to model, which they annotate to indicate silhouettes and part boundaries. Assuming that the object is symmetric, we also ask users to provide a few annotations of symmetric correspondences. The object is then automatically reconstructed in 3D. At the core of our method is an algorithm to infer the 2D projection of rulings of a developable surface from the traced silhouettes and boundaries. We impose that the surface normal is constant along each ruling, which is a necessary property for the surface to be developable. We complement these developability constraints with symmetry constraints to lift the curve network in 3D. In addition to a 3D model, we output 2D patterns enabling to fabricate real prototypes of the object on the photo. This makes our method well suited for reverse engineering products made of leather, bent cardboard or metal sheets.

Keywords: single-view 3D reconstruction, image-based modeling, sketch-based modeling, developable surfaces

1. Introduction

A developable surface is a surface that can be unfolded onto a plane without stretching or tearing. Developable materials such as paper, cardboard, metal sheets, cloth or leather are extensively used in the design of industrial products, from Chinese lanterns, furniture, ship hulls and architecture to many fashion items. Despite multiple fields of application, modeling and editing developable surfaces in 3D is a complex problem, for which standard interactive modeling frameworks do not hold. In this work we investigate a new way to create piecewise developable surfaces, namely reverse engineering them from annotated photographs.

Single-view reconstruction of 3D shapes is an ill-posed problem, as a 2D picture can represent an infinite number of different 3D surfaces. Given annotated contours, prior work managed to address this problem in a few specific cases by complementing the minimization of re-projection error with specific geometric constraints. These constraints include parallelism and orthogonality [17], exact mirror-symmetry [5], or orthogonality of cross-sections in the case of industrial design sketches [33]. In this work, we contribute to this line of research by introducing a new constraint, tailored to developability conditions.

Our work builds on a specific property of developable surfaces that was never used, to our best knowledge, for inferring 3D from a single sketch or annotated photo. Developable surfaces are ruled surfaces with a constant tangent plane (i.e. constant normal) along each ruling. Starting from this property,

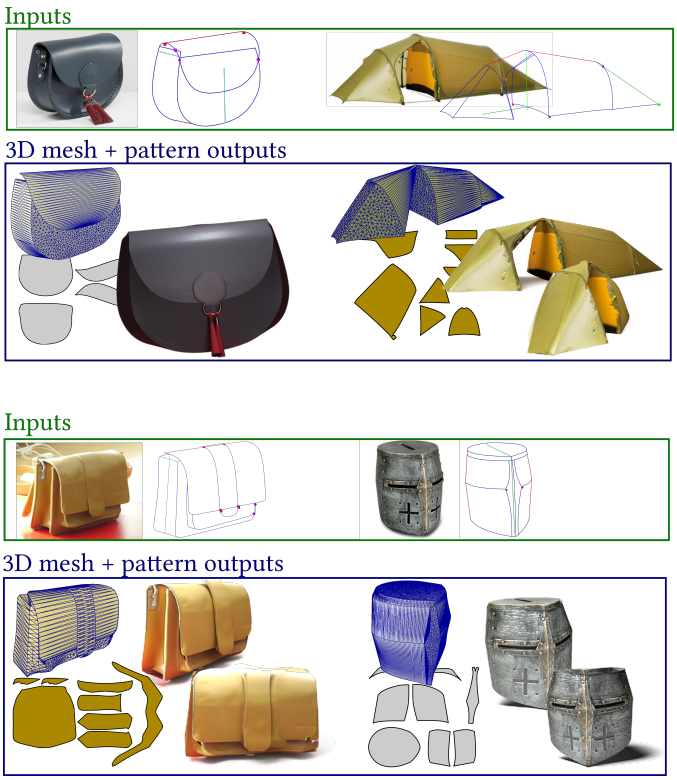

Figure 1: Given an annotated line drawing traced over the photograph of a piecewise-developable object (top green rows) , our method produces a 3D model of the visible parts of the object, and of some of the occluded parts inferred by symmetry. Our method guarantees developability of the model, as shown by the flattened patterns (bottom blue rows).

Email address: amelie.fondevilla@inria.fr (Amélie Fondevilla)

we show that the silhouette of a developable surface is necessarily composed of straight segments aligned with the rulings. Building on these observations, our key idea is to propagate rulings from the object silhouettes towards the interior of surface patches, and to constrain the end-points of surface segments along each ruling to share the same normal in 3D.

Used alone, developability constraints would not be sufficient to infer 3D, since many developable surfaces share the same 2D projection. We therefore complement piece-wise developability with a global symmetry assumption, which is justified in practice by the fact that many objects around us are designed to be symmetric. Leveraging symmetry also allows us to recover some – although not all – occluded parts of the object from a single image. Note that our method is fully formulated using linear constraints, which contrasts with approaches that require non-linear orthogonality constraints between contour curves [17, 33].

We have integrated our algorithm into a sketch-based modeling system where users annotate silhouettes and symmetries over a photograph of the object they wish to reconstruct. Tracing over a photograph allows inexperienced users to quickly model common objects. In contrast, drawing an imaginary developable surface without guidance would require much more expertise, especially to respect the counter-intuitive property that the silhouette of a smooth developable surface is composed of straight segments.

In summary, our contributions are:

1. An end-to-end system to reverse-engineer piecewise developable objects from a single annotated photograph;
2. A method to infer the 2D projection of rulings over a developable surface from its silhouette (Section 5);
3. A new linear energy term for encouraging the developability of a 3D surface reconstructed from the projection of its rulings (Section 6);

We validate our method by reconstructing a variety of piecewise-developable objects from photographs, such as furniture, fashion items, or tents.

2. Related work

A number of methods have been proposed to create 3D objects from drawings and annotated images, as detailed in various surveys [20, 7]. Here we focus on single-image optimization-based methods, to which we contribute by leveraging geometric properties of developable surfaces.

Single-image reconstruction of line drawings. Reconstructing a 3D object from a single drawing is a severely under-constrained problem because each point of the drawing may correspond to an arbitrary 3D point along a viewing ray [1]. A popular approach to single-image reconstruction consists in complementing a re-projection error with various geometric constraints on the lines of the drawing. Such constraints are typically expressed as energy terms in an optimization that balances all concurrent goals. One of the first such algorithms was proposed by Lipson and Shpitalni [17], who focus on polyhedral shapes on which they impose parallelism, orthogonality, planarity and symmetry constraints. This algorithm forms the basis of many subsequent systems, enabling rapid prototyping with finite-element simulation [29], in-context modeling of furniture [16], and 3D reconstruction of complex models composed of multiple parts [34]. However, the constraints used by these methods restrict them to objects dominated by flat, orthogonal faces. Schmidt et al. [24] lift this restriction by reconstructing polyhedral shapes that serve as scaffolds to model curved surfaces, mimicking a traditional drawing technique used by professional designers. Similarly, Shao et al. [25] and Xu et al. [33] derive an orthogonality constraint from cross-section lines that designers draw to convey curvature directions on smooth surfaces. Iarussi et al. [11] also exploit cross-section lines to propagate curvature directions to all pixels of a design sketch, from which they deduce surface normals.

We complement these approaches by proposing a new geometric constraint specific to developable surfaces. Similarly to Iarussi et al. [11], our constraint acts on the surface orientation along lines of minimal curvature. However, we propagate this constraint from the silhouettes of the shape, which alleviates the need for adopting professional drawing techniques such as scaffolds and cross-sections. In addition, our constraint is linear and as much easier to solve than non-linear orthogonality constraints. However, our developability constraint alone is not sufficient to lift a drawing to 3D. We thus complement it with a global symmetry constraint, similar in spirit to the work of Cordier et al. [5] and Öztireli et al. [21].

Our approach is also related to image-based modeling systems that allow users to create 3D shapes from one or a few photographs of a real-world object. Many systems reduce ambiguity by assuming that users model parametric shapes, such as polyhedrons with few parameters [8] and generalized cylinders [4]. An alternative is to build shape priors from large collections of 3D models [10]. In contrast, we focus on leveraging generic geometric constraints rather than low-dimensional parametric models and data-driven priors.

Modeling developable surfaces. Developable surfaces are ubiquitous in design, architecture and fashion, which has motivated the development of dedicated modeling systems. While early method for modeling 3D garment from sketches did not consider developability constraints [30], more recent works take as input an existing 3D model and deform it to achieve developability [32], or approximate it with developable panels [18, 14, 28, 19]. Alternatively, lofting methods find developable surfaces that interpolate 3D boundary curves [9, 23]. In contrast, our input is a network of 2D curves traced over a picture and we exploit properties of developable surfaces to lift the drawing to 3D while ensuring that the output model is developable. Closer to our work is SketchingFolds [12], a sketch-based modeling system that reconstructs fashion items from sketches drawn from two orthogonal viewing directions. The two views serve to compute an initial 3D reconstruction using visual hull, which is then optimized to satisfy surface orientation constraints along contour lines while maintaining devel-

opability. In contrast, our method is capable of inferring 3D from a single input image, and uses developability as a mean to infer 3D information rather than as a post-process optimization. Another notable difference is that we target surfaces composed of rigid materials (paper, cardboard, wood, metal) rather than soft materials with folds.

Similarly to Pérez & Suárez [22], we achieve developability by imposing that the tangent plane is constant along each ruling. However, while they minimize the *warp angle*, i.e. the angle between the normals at each ruling extremity, we obtain a linear constraint by enforcing that adjacent rulings of the surface form planar quads. Our approach is inspired by planar quad meshes as introduced by Liu et al. [18], although the planarity constraint they use is non-linear.

Our approach is also inspired by the work of Ulupinar and Nevatia [31], who reconstruct developable quad patches from 2D contours by assuming that two opposite sides of a patch correspond to rulings, while the two other sides correspond to cross-sections orthogonal to the rulings. They additionally exploit the symmetry of such patches to propagate rulings and cross-sections inside the patch to obtain multiple orthogonality constraints. Our approach differs in the way we identify and propagate rulings over more general patches, and in how we constrain surface orientation along the rulings with a linear energy term. In addition, our optimization acts over a complete curve network rather than on isolated surface patches.

3. Background on developable surfaces

Smooth C^2 developable surfaces are well studied in differential geometry [3]. They are special ruled surfaces, which can be unfolded into the plane without any distortion. Let $\mathbf{X}(u, v) = (1 - v)\, A(u) + v\, B(u)$ be the parameterization of a ruled surface generated by two boundary curves $A(u)$ and $B(u)$, called directrices. The line passing through $A(u)$ and $B(u)$ for any parameter value u is called a *ruling*. Developable surfaces are ruled surfaces with the same tangent plane at all points along a ruling, i.e. the surface normal vectors are constant along a ruling. Developability is thus equivalent to the planarity property

$$\{A,\ B,\ A + \dot{A},\ B + \dot{B}\} \quad \text{are coplanar for all } u, \qquad (1)$$

where $\dot{A}$ denotes the derivative dA/du.

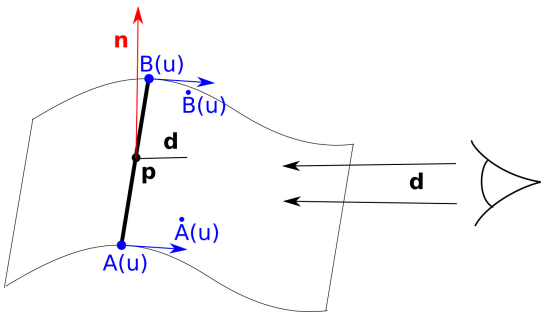

Figure 2: Developable surface and a ruling projecting onto a straight silhouette.

C^2-continuous developable surfaces have the interesting property that their 2D silhouettes are straight segments corresponding to the 2D projection of rulings. Our system exploits this key property to compute the projected rulings of surface patches adjacent to annotated straight silhouettes.

We propose a short derivation of this property. Let us first define a *silhouette* as the set of points of a surface S with normal $\mathbf{n}$ orthogonal to the view direction $\mathbf{d}$ [1]. Note that using this definition, silhouettes do not include borders (ie. boundaries of a trimmed surface). The 2D silhouette is obtained by projection of this set of points to the image plane. We only consider visible silhouettes in the remainder of the paper as we do not expect the user to annotate occluded ones.
Let us consider a point $\mathbf{p}$ on a silhouette of a C^2 developable surface (see Figure 2). By definition, as point of a developable surface belongs to some ruling with a constant normal vector along it. Moreover, as a point of a silhouette, the normal vector at $\mathbf{p}$ is orthogonal to the viewing direction $\mathbf{d}$. Therefore, all points along the ruling have a normal vector orthogonal to $\mathbf{d}$. This makes the ruling being a silhouette and the silhouette being a straight line. As a consequence, all silhouettes of a developable surface are necessarily rulings. Since the rulings are straight lines, their projections form 2D lines in the image plane.

This property is a special case of Koenderink's theorem [15], which states that the sign of curvature of the silhouette is the same as the sign of the Gaussian curvature of the underlying surface. Since developable surfaces have zero Gaussian curvature, their silhouettes have zero curvature. To be fully complete, we further show in Appendix A that the only configuration under which a developable surface can yield a non-straight silhouette is when the silhouette is confounded with the surface boundary, such as when a cylinder is viewed from a viewpoint aligned with its axis of revolution. In our context, we assume that the object is not photographed from such an accidental viewpoint.

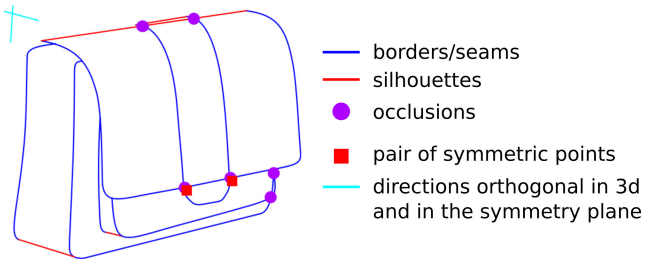

Figure 3: Illustration of the annotations provided by the user.

4. Overview of our system

Our system takes as input a photograph of a developable object, on which the user traces 2D Bézier curves (piecewise cubic C^1 Bézier splines) to delimit surface patches. We also ask users a few annotations, illustrated in Figure 3. We use color to distinguish between *silhouettes (orange), as defined in Section 3, and*

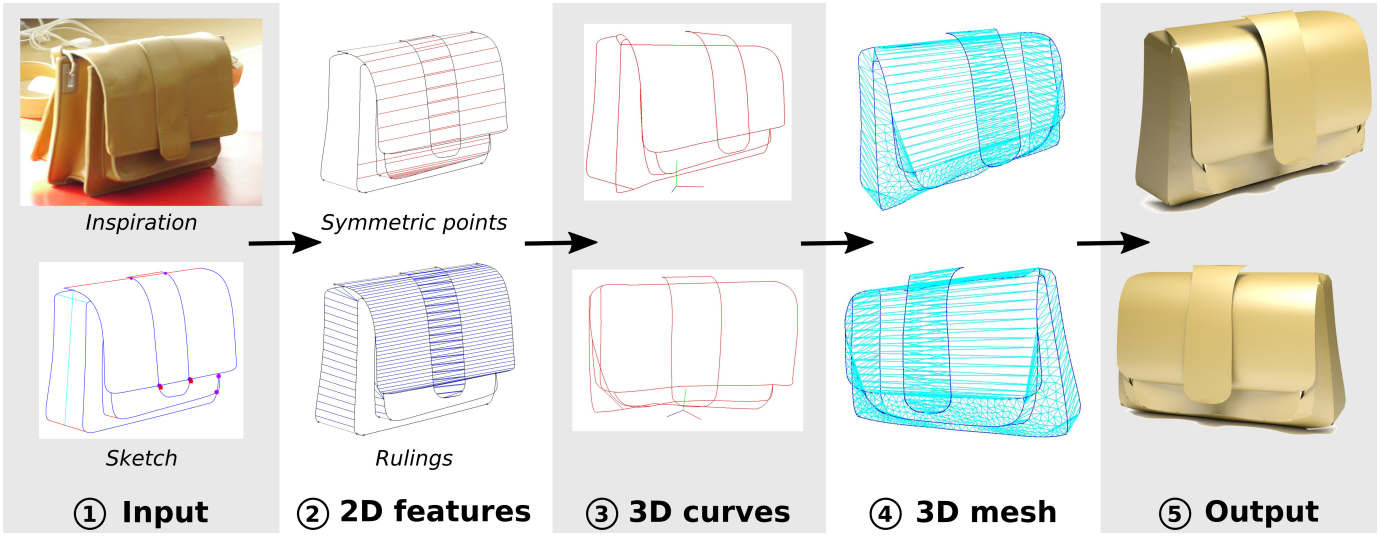

Figure 4: Overview of our approach. (1) Our system takes as input a photograph where the user has traced the object silhouette and the surface patch boundaries. We also ask users to annotate a global symmetry plane and to indicate symmetric curves. (2) We analyze these annotations to register symmetric curves and to propagate rulings from the silhouettes towards the interior of the surface patches. The detected symmetric points and rulings provide us with geometric constraints on the 3D curves, which we express as linear terms in an optimization. (3) Solving this optimization produces a 3D curve network, which we subsequently surface with developable patches. (4,5) These curves are used as boundaries to generate a piecewise developable mesh of the object

other patch boundaries (blue). We also support limited occlusion by letting users indicate 2D intersections that should not be enforced in 3D (purple disks). The other annotations help us exploit symmetry. The user specifies a global symmetry plane by tracing two vectors that represent an orthogonal frame in that plane (cyan lines), and two symmetric points (red squares), which form a third vector orthogonal to the two other ones. Finally, the user indicates each pair of symmetric curves as well as self-symmetric curves. While automatic methods have been proposed to detect global symmetry automatically [5, 21], we found that these annotations are easy to provide while greatly simplifying subsequent analysis. We additionally assume that the photograph is approximately orthographic and taken from an informative viewpoint with little foreshortening.

Figure 4 illustrates the main steps of our method. The first stage of our pipeline analyzes the user annotations (Figure 3) to compute constraints on the 3D interpretation of the curves. We first identify surface patches by extracting the minimal cycles of the curve network (Figure 5). We then generate rulings inside each patch partly delimited by a silhouette curve, by building on our observation that the silhouettes of a developable surface are straight lines aligned with rulings. Finally, we find point to point correspondences between each pair of symmetric curves.

The output of the analysis stage is a 2D set of rulings and a set of 2D pairs of symmetric points, see Figure 4-(2). The second stage of our approach aims at lifting the contour curves in 3D (Figure 4-(3)), which we achieve by constraining their control points to satisfy the detected symmetries while yielding a constant surface normal along rulings. We express these symmetry and developability constraints as linear terms in a function to optimize. This function is complemented by terms expressing the minimization of re-projection errors and foreshortening, enabling us to improve robustness to sketch inaccuracies and perspective distortions.

The last stage of our approach generates a developable triangle mesh for each surface patch, see Figure 4-(4). We simply generate two triangles for each pair of consecutive rulings for the patches partly delimited by silhouette segments. We resort to a more involved surface optimization process [2] for the other patches.

5. 2D curve analysis and rulings extraction

The first stage of our method analyses the input 2D curves and annotations to identify the surface patches, generate their rulings, and build correspondences between symmetric curves.

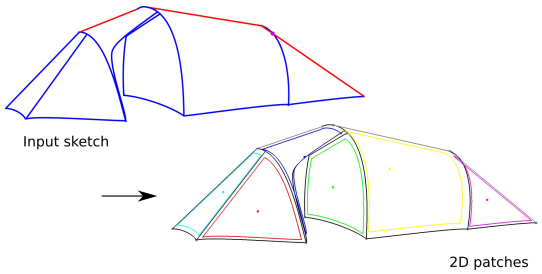

Figure 5: Extraction of 2D patches. Top: input sketch with silhouettes in red and boundaries in blue. Bottom: Different colors are used for the different 2D patch.

5.1. Extracting 2D patches

The input to our method is a network of Bézier curves that represent the object smooth silhouettes, sharp boundaries, or in-

terior seams. We automatically detect intersections between the traced curves and resample them to obtain C^1 piecewise cubic curves with endpoints at intersections. The number of Bézier segments per curve depends on the desired accuracy. We then extract the minimal cycles of this curve network, each cycle representing a surface patch, see Figure 5 for an example. To do so, we represent the curve network as a graph where the nodes correspond to curve intersections and the edges to curve segments between the intersections. We detect all possible cycles in this graph, and reject any cycle that contains another cycle.

5.2. Extracting rulings

As noted in Section 3, the smooth silhouette of a developable surface forms straight line segments aligned with the projected rulings. We now exploit this property to extract rulings over all 2D patches bounded by a silhouette segment.

Our algorithm starts from the straight silhouette segment and propagates it along its adjacent curve segments, as illustrated in Figure 6. Similarly to Ulupinar and Nevatia [31], we guide this propagation by making the additional assumption that the surface patch is a straight generalized cone cut by two parallel planes, although we do not require these planes to be perpendicular to the cone axis. Under this assumption, the tangents of the two curves that intersect each ruling are parallel in 3D. We further assume an orthographic projection, so these tangents are also supposed to be parallel in 2D.

Given the above assumptions, our goal is to form rulings by matching pairs of points along the two curves adjacent to the silhouette segment such that

- The rulings should intersect the curves at points with parallel tangents.
- Adjacent rulings should be as parallel as possible.

While perfect parallelism of rulings is only true for straight generalized cylinders, the second objective acts as a regularization term in the presence of ambiguity of the first term, such as when the two curves we traverse contain straight segments.

We use dynamic time warping [13] to perform this point to point matching: this algorithm uses dynamic programming to minimize a cost function reflecting the quality of the point matches. Denoting C_i and C_j the two curves to be matched, and $\{C_i(k \in [0..N])\}$, $\{C_j(l \in [0..M])\}$ successive point samples on these curves, we express the cost of matching sample $C_i(k)$ to $C_j(l)$ with the recursive equation

$$\begin{aligned}\gamma(k,l) = \alpha(\,\dot{C}_i(k), \dot{C}_j(l)\,) \\ + \min\{\Gamma_{k,l}(k-1,l-1), \Gamma_{k,l}(k-1,l), \Gamma_{k,l}(k,l-1)\}\end{aligned} \tag{2}$$

where

$$\Gamma_{k,l}(r,s) = \gamma(r,s) + \lambda\,\alpha(\,C_i(k) - C_j(l), C_i(r) - C_j(s)\,)$$

and $\dot{C}_i(k)$ denotes the tangent of curve C_i at sample k and α measures the angle between two vectors. The first term of the equation penalizes non-collinear tangents, while the recursive second term penalizes non-collinear successive rulings. See Figure 6.

Once we have computed all rulings using propagation along two contour curves, we reject those lying outside the patch, using the intersection with the last contour.

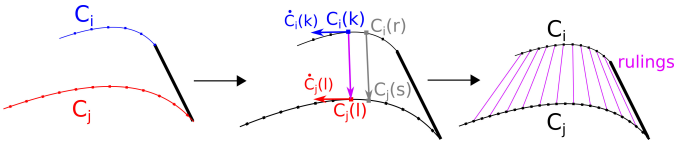

Figure 6: Our method generates 2D rulings over surface patches by propagating silhouette segments using the dynamic time warping algorithm.

5.3. Extracting symmetric correspondences

The last step of the 2D curve analysis stage consists in extracting point-wise correspondences between all symmetric curves in the network. Note that two symmetric curves may belong to different surface patches, as C and C' in Figure 7. Under orthographic projection, the lines that join symmetric correspondences are all parallel to the projected normal of the symmetry plane. In practice, we obtain this direction of symmetry from the two symmetric points annotated by the user (red squares in Figure 7, left). We then build correspondences between each pair of symmetric curves by sampling them curvilinearly and finding for each sample of one curve the closest sample of the other curve in the direction of symmetry.

6. 3D contours optimization

We are now ready to compute the 3D coordinates of the Bézier curves using symmetry and developability constraints. Since the line drawings we target are traced over photographs, they may be distorted by drawing inaccuracy and weak perspective. Such distortions prevent a direct reconstruction using hard 2D positional and symmetry constraints, as performed by Cordier et al. [5]. Instead, we formulate our reconstruction algorithm by defining a set of energy functions as soft constraints on the 3D coordinates of the Bézier control points, and compute a global optimal solution that can deviate from the input curves if necessary.

6.1. Energy formulation

Our energy function is composed of five different quadratic functions. The first three, namely projection accuracy, minimal variation and minimal foreshortening, were introduced by Xu et al. [33]. We will briefly recall their definitions for the sake of completeness. We further introduce two new energy functions: *developability* is the key feature of our method as it enables to restore developable surface patches from the 2D sketches, while *symmetry* is necessary to recover the depth of the object. In contrast to the formulation by Xu et al. [33], all our functions are quadratic, which allows us to find a global minimum using a linear solver.

In the following, we denote B^k the k-th segment of a cubic Bézier curve, and $\{\mathbf{b}_i^k\}_{i=0..3}$ its control points. Note that

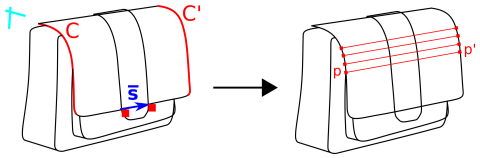

Figure 7: Symmetry correspondences computed for all pairs of symmetric curves according to the global mirror symmetry. The user-annotated pair of symmetric points (red squares) provides the direction of symmetry in 2D.

since the curves are piecewise C^1-continuous, the three consecutive points $\mathbf{b}_2^k, \mathbf{b}_3^k = \mathbf{b}_0^{k+1}, \mathbf{b}_1^{k+1}$ are collinear. We differentiate a 2D control point, whose coordinates are provided by the user sketch, using an upper-bar notation $\bar{\mathbf{q}}$, from the corresponding 3D control point denoted by $\mathbf{q}$. The latter are the unknowns in our system.

Projection accuracy penalizes control points $\mathbf{q}$ for which the projection onto the image plane $\mathbf{q}|_{z=0}$ is far from the existing projection in the sketch $\bar{\mathbf{q}}$:

$$E_{\text{proj}} = \| \mathbf{q}|_{z=0} - \bar{\mathbf{q}} \|^2.$$

Minimal variation penalizes stronger variations in depth than in 2D by favoring an affine relation between 4 successive non-collinear control points. These 4 points constitute either an entire Bézier segment, or cover two segments, such as $\mathbf{b}_1^k, \mathbf{b}_2^k, \mathbf{b}_1^{k+1}, \mathbf{b}_2^{k+1}$. Denoting the 4 points by $\mathbf{q}_{i=0,\dots,3}$, the energy term on non-collinear points is defined as

$$E_{\text{minvar}} = \left\| \varphi_0 \mathbf{q}_0 + \varphi_1 \mathbf{q}_1 + \varphi_2 \mathbf{q}_2 - \mathbf{q}_3 \right\|^2,$$

where $\varphi_0, \varphi_1, \varphi_2$ are the barycentric coordinates of $\bar{\mathbf{q}}_3$ with respect to $\bar{\mathbf{q}}_0$, $\bar{\mathbf{q}}_1$, $\bar{\mathbf{q}}_2$. In addition, the minimal variation term also applies on all triplets of successive collinear points to encourage them to remain collinear in 3D, which is critical to maintain C^1-continuty between Bézier segments. Denoting such triplets by $\mathbf{r}_{i=0,1,2}$, the energy term on collinear points is defined as

$$E_{\text{col}} = \left\| (1-\delta)\mathbf{r}_0 + \delta \mathbf{r}_2 - \mathbf{r}_1 \right\|^2,$$

where $\delta, (1-\delta)$ are the barycentric coordinates of $\bar{\mathbf{r}}_1$ with respect to $\bar{\mathbf{r}}_0$ and $\bar{\mathbf{r}}_2$.

Foreshortening penalizes strong differences in the depth of two successive control points $\mathbf{q}_i$, $\mathbf{q}_{i+1}$ of a Bézier curve

$$E_{\text{foreshort}} = (\mathbf{q}_{z,i} - \mathbf{q}_{z,i+1})^2.$$

Symmetry encourages the symmetry of two points with respect to a global symmetry plane defined by a unit normal $\mathbf{n} = (n_x, n_y, n_z)$.

We first compute the 3D coordinates of the plane normal $\mathbf{n}$ from the annotation of the user. Let $\bar{\mathbf{s}}$ be the 2D vector linking the pair of symmetric points illustrated by the red squares in the sketch, see Figure 3. This vector is the projection of a vector $\mathbf{s}$ that should be collinear to $\mathbf{n}$. Moreover, the cyan segments shown in Figure 3 provide the projections $\bar{\mathbf{u}}$ and $\bar{\mathbf{v}}$ of two vectors that should lie in the plane orthogonal to $\mathbf{n}$. The 3D vectors $\mathbf{u}$ and $\mathbf{v}$ should thus satisfy $\mathbf{u} \cdot \mathbf{v} = 0$, $\mathbf{u} \cdot \mathbf{s} = 0$, and $\mathbf{v} \cdot \mathbf{s} = 0$. Expanding these three equations leads to the expression $\mathbf{s}_z^2 = -(\bar{\mathbf{u}} \cdot \bar{\mathbf{s}})\ (\bar{\mathbf{v}} \cdot \bar{\mathbf{s}})/(\bar{\mathbf{u}} \cdot \bar{\mathbf{v}})$, from which we deduce $\mathbf{n} = \mathbf{s}/\|\mathbf{s}\|$.

Given two 2D symmetrical points $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ with respect to the plane orthogonal to $\mathbf{n}$, the 3D vector $\mathbf{t} = \mathbf{p} - \mathbf{p}'$ and midpoint $\mathbf{m} = \mathbf{p} + \mathbf{p}'$ can be computed using the formulas derived by Cordier et al. [6]

$$\begin{cases} \mathbf{t} = & \left[(\bar{\mathbf{p}}_x - \bar{\mathbf{p}}'_x),\ (\bar{\mathbf{p}}_y - \bar{\mathbf{p}}'_y),\ \dfrac{-n_z}{n_y}(\bar{\mathbf{p}}_y - \bar{\mathbf{p}}'_y) \right]^T \\ \mathbf{m} = & \dfrac{1}{2} \left[(\bar{\mathbf{p}}_x + \bar{\mathbf{p}}'_x),\ (\bar{\mathbf{p}}_y + \bar{\mathbf{p}}'_y), \right. \\ & \left. \dfrac{-1}{n_z} \left(n_x(\bar{\mathbf{p}}_x + \bar{\mathbf{p}}'_x) + n_y(\bar{\mathbf{p}}_y + \bar{\mathbf{p}}'_y) \right) \right]^T. \end{cases}$$

We then propose the following energy formulation for two symmetrical points $\mathbf{p}$ and $\mathbf{p}'$

$$E_{\text{sym}} = \left\| (\mathbf{p} - \mathbf{p}') - \mathbf{t} \right\|^2 + \left\| \frac{1}{2}(\mathbf{p} + \mathbf{p}') - \mathbf{m} \right\|^2. \quad (3)$$

Note that the points $\mathbf{p}$ and $\mathbf{p}'$ are not Bézier control points, but sample points of the arc-length parameterized Bézier curves. However, each sample point can be expressed as a linear combination of the 4 unknown control points $\{\mathbf{q}_i\}_{i=0}^3$ of the Bézier segment they belong to. The energy term E_{sym} is thus quadratic with respect to the variables $\mathbf{q}_i$ of our system.

Developability encourages a constant tangent plane along each ruling. We approximate the developability criteria from Equation (1) using the two points $\mathbf{p}_i$ and $\mathbf{p}'_j$ of curves C and C' joined by a ruling and their immediate neighbors $\mathbf{p}_{i+1}$ and $\mathbf{p}'_{j+1}$. Given these 4 points, consistency of the tangent plane along each ruling $\mathbf{p}_i\mathbf{p}'_j$ is expressed using the energy functional

$$E_{\text{develop}} = \left\| \phi_0 \mathbf{p}_i + \phi_1 \mathbf{p}_{i+1} + \phi_2 \mathbf{p}'_j - \mathbf{p}'_{j+1} \right\|^2. \quad (4)$$

where ϕ_0, ϕ_1, ϕ_2 are the barycentric coordinates of $\bar{\mathbf{p}}'_{j+1}$ with respect to $\bar{\mathbf{p}}_i$, $\bar{\mathbf{p}}_{i+1}$, $\bar{\mathbf{p}}'_j$. Here again, note that each of these 4 points is some affine combination of the Bézier control points of the segments they belong to. E_{develop} is therefore a quadratic function of the unknowns.

6.2. *Optimization*

Finally, the different energy terms are summed over the free variables and assembled together in a global quadratic energy function E

$$E = \omega_1 E_{\text{proj}} + \omega_2 E_{\text{minvar}} + \omega_3 E_{\text{col}} + \omega_4 E_{\text{foreshort}} + \omega_5 E_{\text{sym}} + \omega_6 E_{\text{develop}}$$

In practice, we fix $\omega_1 = \omega_3 = \omega_5 = \omega_6 = 1$ and $\omega_2 = 10^{-1}$, $\omega_4 = 10^{-8}$. The optimal solution can be efficiently computed as the solution of a linear system of the unknown control points. In practice we use an SVD decomposition in order to handle

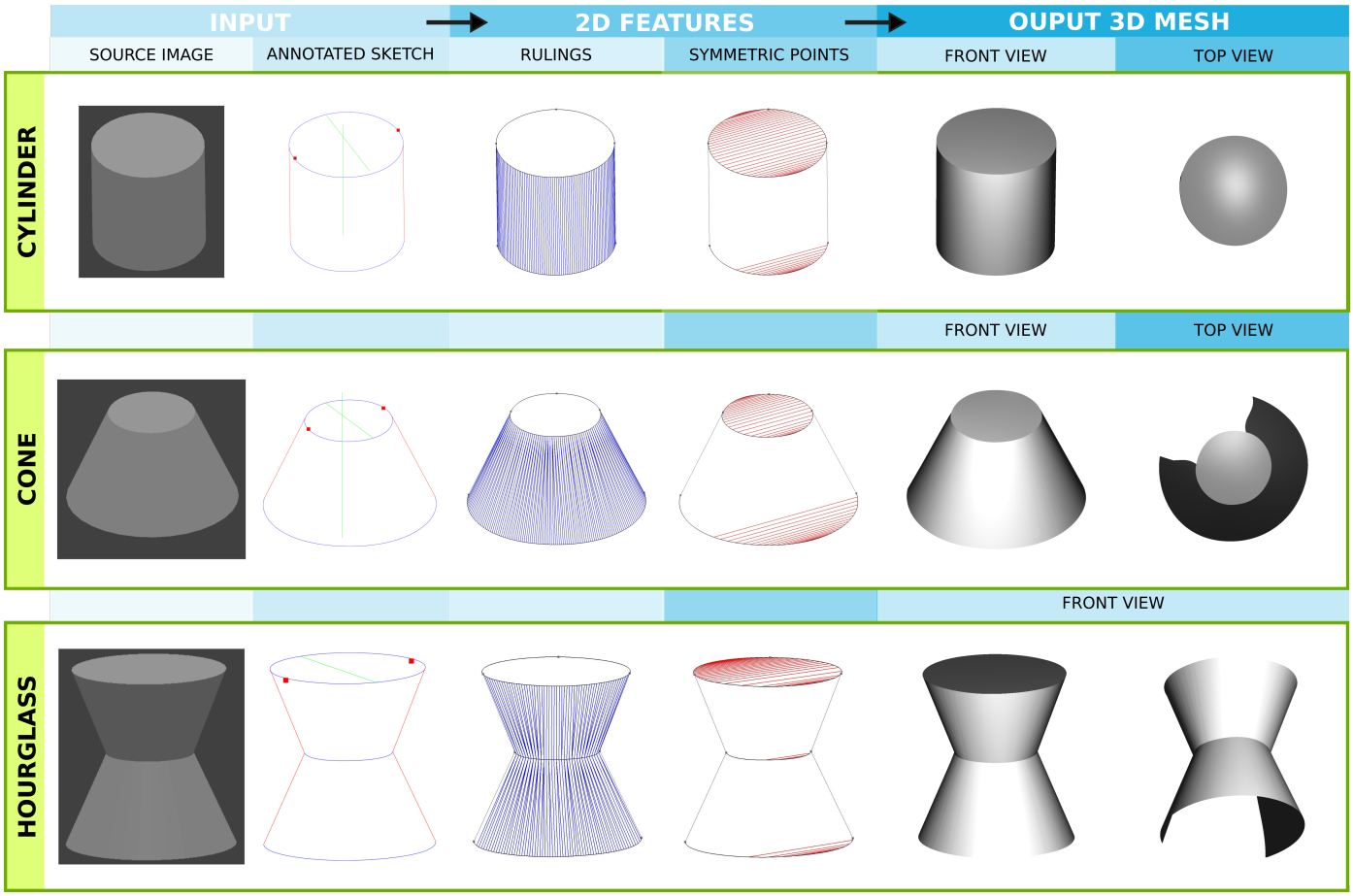

Figure 8: Results on synthetic examples.

potential rank-deficiency of the associated matrix.
For a sketch described by N control points, the size of the matrix is generally of the order of $4N+3R+6S$, where R is the number of rulings, and S the number of pairs of symmetric points.

7. Developable surface generation

Our final step consists in generating a symmetrical surface bounded by the 3D contours and supported by the rulings we computed. Since the resulting surface is developable by construction, we can flatten it to obtain the associated 2D patterns.

Symmetrizing the contours and rulings. While our 3D contours optimization takes advantage of symmetric correspondences, it may result in 3D curves that are not perfectly symmetrical with respect to the global symmetry plane. Inaccuracies of the sketch, perspective weakness, and occluded parts may be the reason for imperfect symmetry. We therefore enforce perfect mirror symmetry in the network of curves and rulings as an extra step. Assuming that the object is seen from an informative 3/4 view, the positive and negative half space separated by the global symmetry plane can be respectively considered as the most reliable, and less reliable sides. Our approach consists in removing all curves and rulings belonging to the negative half space, and generating new ones using mirror symmetry from the one in the positive half space. In the specific case where a patch is self-symmetrical, i.e. being defined in both half spaces, some of the rulings may cross the symmetry plane. We delete such rulings and generate new ones by linking pairs of points which are symmetrical with respect to the symmetry plane.

Generating a triangulated mesh and its 2D pattern. For each patch, we infer a surface interpolating boundary curves. If a patch contains rulings, then the corresponding 3D curves can be trivially associated to a mesh by triangulating consecutive rulings, which does not require the introduction of interior points. If not, we generate a surface with minimal mean curvature interpolating the 3D border using the variational Laplacian approach [27, 2]. In this case, the connectivity of this mesh is generated using a Delaunay triangulation of the 2D patch contours, where triangles are constrained to have a maximal area of 10% of the diagonal of the input image. Note that the resulting mesh could be easily improved using a developability optimization step such as Wang & Tang [32]. As a last step, we generate a 2D pattern for each mesh, using standard parameterization algorithm minimizing stretch [26]. This works well in our case since all patches are close to developable.

8. Results and Validation

8.1. Results

We used our method to model a variety of synthetic and real-world objects.

Synthetic models . For the 3 examples in Figure 8, we created ground-truth 3D objects representing piecewise developable surfaces (cones and cylinders). We rendered each object under an informative viewpoint using orthographic projection, and manually drew the annotations.

Our algorithm generates rulings close to ground-truth, as shown in Figure 9. The 3D reconstructions also correspond well to the expected results, as shown by the top views that reveal near-perfect circular cross-sections. Note that since these models are rendered under perfect orthographic projection, we did not use the foreshortening energy, which acts as a regularization on perspective-distorted drawings.

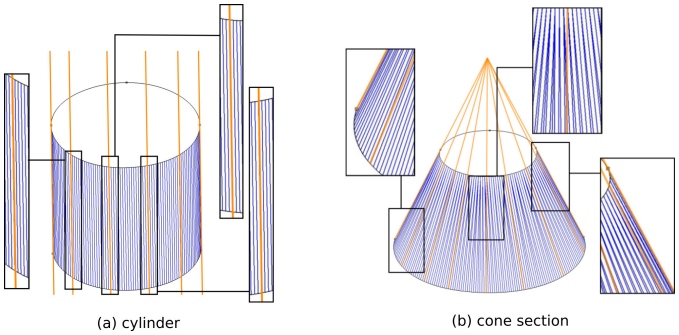

Figure 9: Evaluation of ruling propagation on two synthetic examples. Blue lines represent the rulings found by our algorithm, orange lines represent ground-truth rulings. Our rulings perfectly match ground-truth for the cylinder, and remain close to it for the truncated cone.

Real-world objects . We applied our method to photographs representing real-life objects. The results are displayed in Figure 11. Note that for some of the examples such as the side of the purse on the top, the sketch simplifies the geometry of the object, so that we get stronger developability constraints.
Our method finds plausible rulings for each of the examples. In particular, it succeeds in identifying cylindrical and conical parts, even when the curves linked by rulings do not have the same length. As we only keep rulings that fully lie within the interior of their respective patch, our approach is also able to successfully recover partial cylindrical and conical parts within a patch. This is, for instance, the case on the curved patch of the tent on top of the door hole (see second row of Figure 11). This patch exhibits only 5 rulings despite a very long curved side on the left. Note that this feature also allows to handle partial occlusion associated to concavities (see for instance the top-left patch of the couch in the last row of Figure 11). For the specific case of a perfect cone, we added an extra annotation specifying the location of the apex, as for the right side of the tent. We also illustrate the detected symmetric correspondences, including pairs of symmetric curve (eg left side of the tent) and self-symmetric curves (eg top of the helmet).

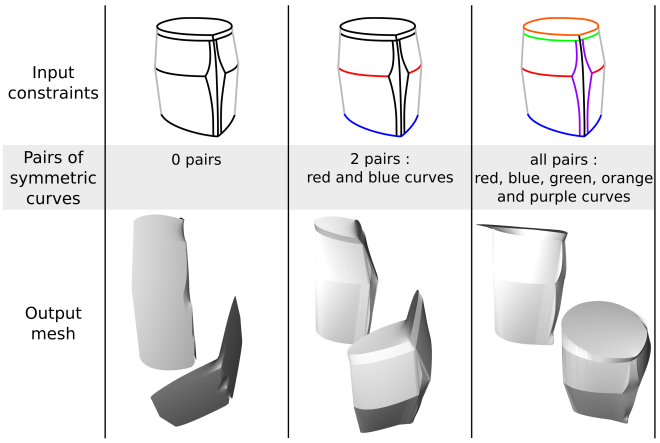

Figure 10: Evolution of the shape obtained while adding incrementally symmetry constraints to the input. The first row shows the symmetry features of the input sketch: each pair of symmetric curve has a given color, black curves correspond to non-symmetric curves, and gray lines to silhouettes. The second row displays two views of the 3D model generated by our method.

Influence of symmetry constraint. Symmetry is an important linear constraint in our method for two reasons: it provides the depth, i.e. the volume to the model, and allows to recover some occluded parts of the object from a single image. In Figure 10 we show the evolution of a result under an increasing number of symmetry annotations, thus allowing the user to iteratively refine the reconstructed 3D model until reaching a satisfying result.

8.2. Evaluation

Metrics. As a mean of evaluation, we measure for each reconstructed model the developability of the patches containing rulings. A ruled surface is perfectly developable if its rulings have a constant tangent plane. We measure the developability error of a ruling by computing the angle between the normals at the ruling's extremities. We average this developability error over all detected rulings of a model. Table 1, last column, shows that this error varies between 6 and 16° for the objects in Figure 11. The computational time varies between 5 and 30 seconds depending on the number of control points in the curve network and the number of rulings (Table 1, second column). Such performances allow an interactive workflow where users can quickly visualize the reconstructed 3D shape and add missing annotations on the photo to improve it if necessary.

Analysis of the developability constraint. The main novelty of our approach resides in the our new developability constraint. We now evaluate its impact on 3D reconstructrion.

We compared our approach with a downgraded version where we removed the term representing developability by setting ω_6 to 0. Without this energy term, the average developability error on the resulting model increases. For example, for the model purse 1, developability error goes from 6.61° with $\omega_6 = 1$ to

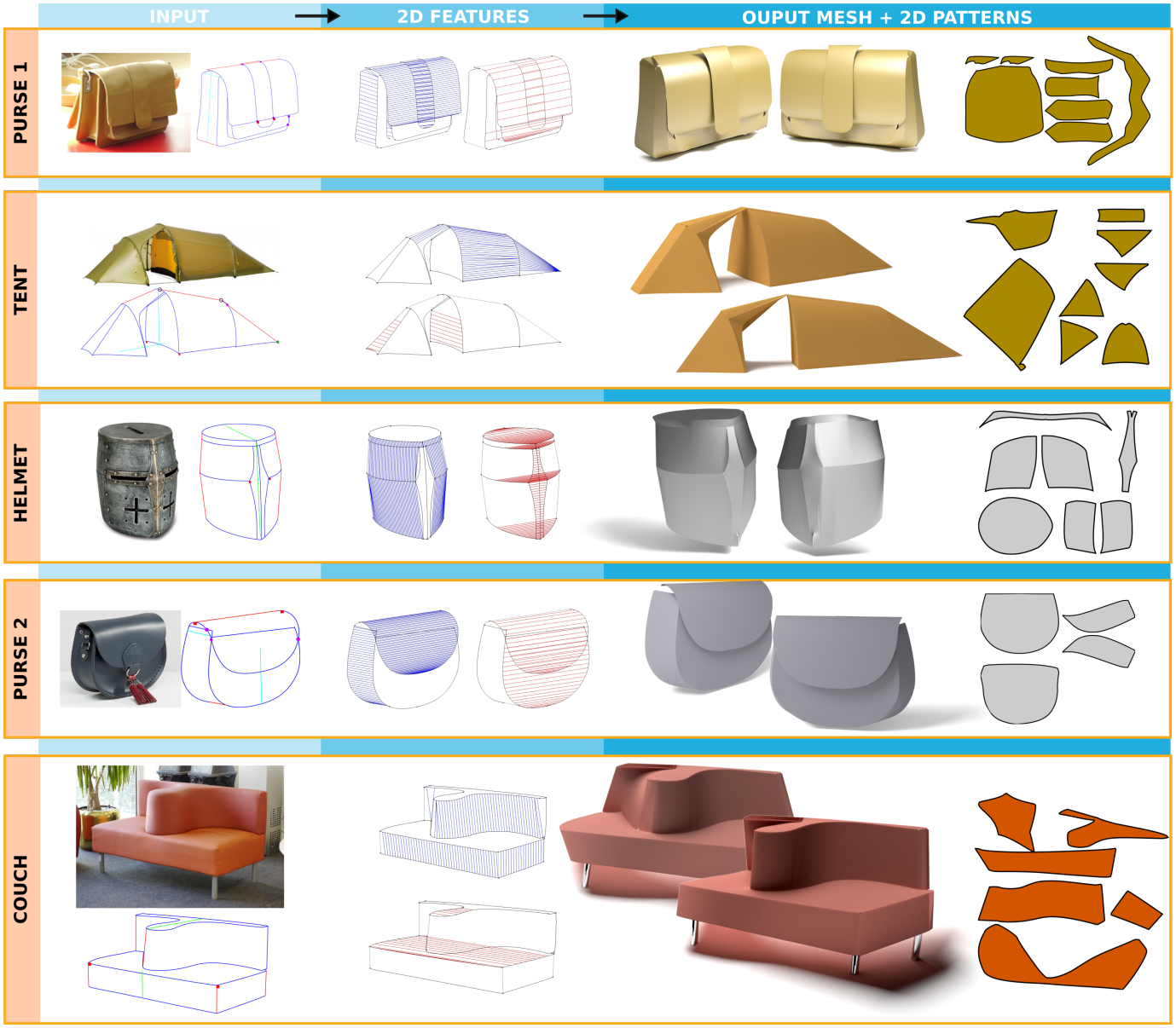

Figure 11: Results on real-world examples. Note that for the complex case of the couch, symmetry constraints were used throughout the process, but our method failed to find pairs of symmetric points for the top part of the couch, due to highly distorted curves. Therefore, we did not use the final symmetrization of the 3D model, and the patterns we provide are not symmetric.

model	time	control points	rulings	pairs of sym. pts	patches (cycles)	avg error develop.
purse 1	28s	119	135	27	7	6.61°
helmet	30s	100	165	93	9	12.49°
purse 2	7.8s	58	112	43	3	9.65°
tent	4.7s	61	26	21	6	16.33°
couch	19s	100	104	18	6	13.84°

Table 1: Dimension, computational time and developability error of the examples in Figure 11

10.22° with $\omega_6 = 0$. We made a similar observation for all other tested models.

We can also note the positive influence of the rulings in the resulting surfaced model. For example, the surface tent model could not be correctly reconstructed using only minimal surfaces, as shown in Figure 12.

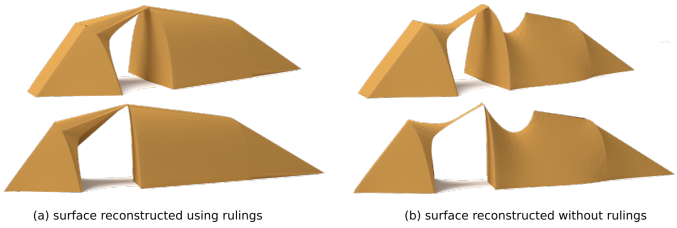

Figure 12: Example of reconstructed surfaces with (a) and without (b) the use of 3D rulings.

Failure cases. In Figure 13 we present two failure cases of our system. Both examples satisfy the assumptions of near orthographic view, global mirror-symmetry of the object and piecewise developability. Our system robustly computes a set of 3D curves and fits the patches.
However, the contours at the top of the cap only consist of patch boundaries and do not exhibit silhouettes. Thus, no rulings are computed by our approach, which results in non-developable surface. Moreover, the visor of the cap contains conical sections, but they do not satisfy our hypothesis of being cut by two parallel planes, thus the criteria of collinear tangent at rulings extremities presented in Section 5.2 does not hold. The rulings extraction is not guaranteed to work in these cases of developable patches, even though the computed rulings in this case remain plausible.
The bag example lacks symmetry input, which is an essential constraint to inflate the volume of the model. Similar to Figure 10-left, middle, the optimization will lead to a flat model. One may overcome these special cases, by allowing the user to e.g increase symmetry annotations with some additionally sketched patch boundaries.

9. Conclusion and Discussion

We have presented the first method enabling the reverse-engineering of symmetric, developable products from a single photo. Our approach assumes a simple orthographic projection. While our least-squares optimization approach tolerates minor perspective distortions as demonstrated by our results, strong perspective effects can yield distorted surfaces. Accounting for perspective would require a more complex formulation of the re-projection error. In addition, perspective should also be taken into account when searching for symmetric correspondences. Figure 14 illustrates how annotating two parallel lines on a symmetric model could help account for vanishing points during 2D analysis.

Since our system only takes a single photograph as input, we cannot reconstruct all occluded parts of the model. Completing the model by simple back-facing symmetry may still lack realism, since some occluded parts, such as the back of the helmet in Figure 11 would not be completed as expected. An interesting direction for future research would be to integrate our developability constraints into a multi-view modeling system where the object would be captured from a few, complementary viewpoints.

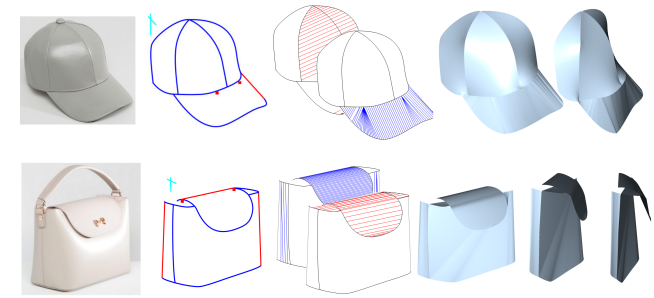

Figure 13: Examples of failure cases that do not exhibit sufficient symmetry and developability features. Note that our system is still able to provide a solution, but the result lacks volume. Note also we present the resulting meshes without the symmetrization step in these examples.

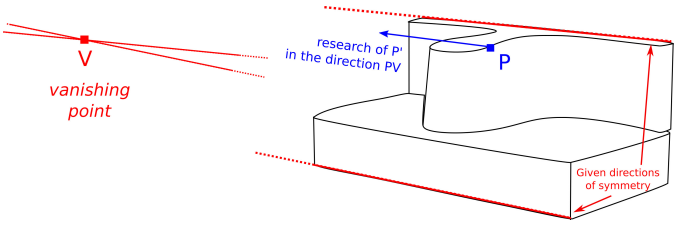

Figure 14: Finding the vanishing point of the image from two annotated red lines would improve the search for the point symmetric to P.

Acknowledgements

This work was supported in part by the ERC starting grant D^3 (ERC-2016-STG 714221) and by research and software donations from Adobe.

This research was also partially funded by the ERC advanced grant no. 291184 EXPRESSIVE.

We thank Estelle Charleroy for the help on the rendering and video montage.

References

[1] Barrow, H., & Tenenbaum, J. (1981). Interpreting line drawings as three-dimensional surfaces. *Artificial Intelligence*, *17*.

[2] Botsch, M., & Kobbelt, L. (2004). An intuitive framework for real-time freeform modeling. *ACM Trans. Graph.*, *23*, 630–4.

[3] Carmo, M. D. (1976). *Differential Geometry of Curves and Surfaces*. Prentice-Hall.

[4] Chen, T., Zhu, Z., Shamir, A., Hu, S.-M., & Cohen-Or, D. (2013). 3-sweep: Extracting editable objects from a single photo. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, *32*, Article 195.

[5] Cordier, F., Seo, H., Melkemi, M., & Sapidis, N. S. (2013). Inferring mirror symmetric 3D shapes from sketches. *Computer-Aided Design*, *45*, 301–11.

[6] Cordier, F., Seo, H., Park, J., & Noh, J. (2011). Sketching of mirror-symmetric shapes. *Visualization and Computer Graphics, IEEE Transactions on*, *17*, 1650–62.

[7] Cordier, F., Singh, K., Gingold, Y., & Cani, M.-P. (2016). Sketch-based modeling. In *SIGGRAPH ASIA Courses* (pp. 18:1–18:222). New York, NY, USA: ACM.

[8] Debevec, P. E., Taylor, C. J., & Malik, J. (1996). Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. *SIGGRAPH'96*, (pp. 11–20).

[9] Frey, W. H. (2002). Boundary triangulations approximating developable surfaces that interpolate a closed space curve. *International Journal of Foundations of Computer Science*, *13*, 285–302.

[10] Huang, Q., Wang, H., & Koltun, V. (2015). Single-view reconstruction via joint analysis of image and shape collections. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, *34*, 87:1–87:10.

[11] Iarussi, E., Bommes, D., & Bousseau, A. (2015). Bendfields: Regularized curvature fields from rough concept sketches. *ACM Transactions on Graphics (SIGGRAPH 2015)*, *34*, 24:1–24:16.

[12] Jung, A., Hahmann, S., Rohmer, D., Begault, A., Boissieux, L., & Cani, M.-P. (2015). Sketching folds: Developable surfaces from non-planar silhouettes. *ACM Trans. Graph.*, *34*, 155:1–155:12.

[13] Keogh, E. J., & Pazzani, M. J. (2001). Derivative dynamic time warping. In *Proceedings of the 2001 SIAM International Conference on Data Mining* (pp. 1–11). SIAM.

[14] Kilian, M., Flöry, S., Chen, Z., Mitra, N. J., Sheffer, A., & Pottmann, H. (2008). Curved folding. In *ACM Transactions on Graphics (TOG)* (p. 75). ACM volume 27.

[15] Koenderink, J. J. (1984). What does the occluding contour tell us about solid shape? *Perception*, *13*, 321–30.

[16] Lau, M., Saul, G., Mitani, J., & Igarashi, T. (2010). Modeling-in-context: user design of complementary objects with a single photo. In *Proc. Sketch-Based Interfaces and Modeling*.

[17] Lipson, H., & Shpitalni, M. (1996). Optimization-based reconstruction of a 3d object from a single freehand line drawing. *Computer-Aided Design*, *28*, 651–63.

[18] Liu, Y., Pottmann, H., Wallner, J., Yang, Y.-L., & Wang, W. (2006). Geometric modeling with conical meshes and developable surfaces. In *ACM Transactions on Graphics (TOG)* (pp. 681–9). ACM volume 25.

[19] Mori, Y., & Igarashi, T. (2007). Plushie: An interactive design system for plush toys. *ACM Transactions on Graphics (proc. SIGGRAPH)*, *26*.

[20] Olsen, L., Samavati, F. F., Sousa, M. C., & Jorge, J. A. (2009). Sketch-based modeling: A survey. *CAG*, *33*.

[21] Öztireli, A. C., Uyumaz, U., Popa, T., Sheffer, A., & Gross, M. (2011). 3d modeling with a symmetric sketch. In *Sketch-Based Interfaces and Modeling* (pp. 23–30).

[22] Pérez, F., & Suárez, J. A. (2007). Quasi-developable b-spline surfaces in ship hull design. *Computer-Aided Design*, *39*, 853–62.

[23] Rose, K., Sheffer, A., Wither, J., Cani, M.-P., & Thibert, B. (2007). Developable surfaces from arbitrary sketched boundaries. In *SGP'07-5th Eurographics Symposium on Geometry Processing* (pp. 163–72). Eurographics Association.

[24] Schmidt, R., Khan, A., Singh, K., & Kurtenbach, G. (2009). Analytic drawing of 3d scaffolds. In *ACM Transactions on Graphics (TOG)* (p. 149). ACM volume 28.

[25] Shao, C., Bousseau, A., Sheffer, A., & Singh, K. (2012). CrossShade: Shading Concept Sketches Using Cross-Section Curves. *ACM ToG (Proc. SIGGRAPH)*, *31*.

[26] Sheffer, A., Lévy, B., Mogilnitsky, M., & Bogomyakov, A. (2005). Abf++: fast and robust angle based flattening. *ACM Transactions on Graphics (TOG)*, *24*, 311–30.

[27] Stanko, T., Hahmann, S., Bonneau, G.-P., & Saguin-Sprynski, N. (2016). Surfacing curve networks with normal control. *Computers & Graphics*, *60*, 1 – 8.

[28] Tang, C., Bo, P., Wallner, J., & Pottmann, H. (2016). Interactive design of developable surfaces. *ACM Transactions on Graphics (TOG)*, *35*, 12.

[29] Tian, C., Masry, M., & Lipson, H. (2009). Physical sketching: Reconstruction and analysis of 3D objects from freehand sketches. *Computer Aided Design*, *41*, 147–58.

[30] Turquin, E., Wither, J., Boissieux, L., Cani, M.-P., & Hughes, J. F. (2007). A sketch-based interface for clothing virtual characters. *IEEE Computer graphics and applications*, *27*.

[31] Ulupinar, F., & Nevatia, R. (1993). Perception of 3-d surfaces from 2-d contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, *15*, 3–18.

[32] Wang, C. C., & Tang, K. (2004). Achieving developability of a polygonal surface by minimum deformation: a study of global and local optimization approaches. *The Visual Computer*, *20*, 521–39.

[33] Xu, B., Chang, W., Sheffer, A., Bousseau, A., McCrae, J., & Singh, K. (2014). True2Form: 3D curve networks from 2D sketches via selective regularization. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, *33*.

[34] Yang, L., Liu, J., & Tang, X. (2013). Complex 3d general object reconstruction from line drawings. In *IEEE International Conference on Computer Vision*.

Appendix A. Silhouette of developable surface

We showed in paragraph 3 that all silhouettes of developable surface necessarily contains the rulings passing by the silhouette points. In this appendix, we derive that rulings are actually the only possible silhouette which are not confounded with the developable surface boundary.

Let us consider a developable surface S parameterized by two parameters $(u, v) \in \mathcal{D} \subset \mathrm{R}^2$ such that [3]

$$\left\{ \begin{array}{ll} & S(u,v) = \varphi(u) + v\,\psi(u) \\ \text{with} & \det(\varphi'(u), \psi(u), \psi'(u)) = 0\ . \end{array} \right. \tag{A.1}$$

We denote $n(u, v)$ the unit normal at parameter (u, v). For the sake of simplicity, we do not explicitly write the parameters (u, v) for the functions when the relation is true for the entire surface. We also denote $S_{,u} = \frac{\partial S}{\partial u}$, and similarly with $S_{,v}$, and $n_{,u}$. We further denote the mixed partial derivative $S_{,uv} = \frac{\partial^2 S}{\partial u \partial v}$.

Note that for a given u_{const}, $S(u_{const}, v)$ is a ruling of the surface, and $S_{,v}(u_{const}, v)$ is a director vector of the ruling. Let us consider a point at parameter (u_0, v_0) on the silhouette satisfying therefore $n(u_0, v_0) \cdot d = 0$, where $n(u_0, v_0)$ is the unit normal at the parameters (u_0, v_0), and d is the constant view direction. The set of parameters yielding to a silhouette in a neighborhood of (u_0, v_0) must satisfy $n(u_0 + \mathrm{d}u, v_0 + \mathrm{d}v) \cdot d = 0$, where $(\mathrm{d}u, \mathrm{d}v)$ are some infinitesimal displacements in the parametric space. As the normal is constant along the v direction, i.e. corresponds to the rulings, the parameter $\mathrm{d}v$ can be dropped. Thus any silhouette of S which is not a ruling should satisfies the relation

$$n_{,u}(u_0, v_0) \cdot d = 0\ . \tag{A.2}$$

We show in the following that Eq. (A.2) only holds if the direction d is aligned with the rulings, i.e. the rare case where the surface is viewed from its side.

First, we can note that satisfying Eq. (A.2) implies that the three vectors

$$(n(u_0, v_0), n_{,u}(u_0, v_0), d) \text{ are forming an orthogonal frame.} \tag{A.3}$$

Indeed, the silhouette condition implies that $n(u_0, v_0)$ is orthogonal to d, Eq. (A.2) implies that $n_{,u}(u_0, v_0)$ is orthogonal to d, and n is necessarily orthogonal to $n_{,u}$ as it is a unit vector. Second, we can note that for a developable surface, the three vectors

$$(n, n_{,u}, S_{,v}) \text{ are an orthogonal frame,} \tag{A.4}$$

for any parameters (u, v). By definition, n is orthogonal to $S_{,v}$. And we can show in the following that $n_{,u}$ is orthogonal to $S_{,v}$. The developability condition from (A.1) can be rewritten in term of surface derivatives as $\det(S_{,u}, S_{,v}, S_{,uv}) = 0$. This determinant can further be expressed in term of scalar and vector product and rewritten as $S_{,uv} \cdot (S_{,u} \times S_{,v}) = 0$ implying that $S_{,uv} \cdot n = 0$. Moreover, one can check that every smooth surface satisfies $S_{,uv} \cdot n = -S_{,v} \cdot n_{,u}$, which leads to the expected conclusion that $S_{,v}$ is orthogonal to $n_{,u}$.

Finally, comparing the two frame in (A.3) and (A.4) leads to the conclusion that d must be parallel to $S_{,v}$, and therefore to the rulings of the surface, which is the expected result to prove. We conclude that silhouette of developable surface which are not only rulings of the surface only arises if the view direction is aligned with the rulings, and then be confounded with the boundary of the surface.

Note that the converse implication is also true. Let us suppose that the surface is viewed from a direction d aligned with the rulings $S_{,v}$. As property (A.4) is satisfied for any parameters (u, v), it implies that d is orthogonal to n, and thus is a silhouette.

As a conclusion we showed that any silhouette of a developable which is not a boundary of the surface is necessarily a straight segments which is a ruling of the surface. Any other possible silhouettes which corresponds possibly to the side view of the surface, arising when the view direction is aligned with the rulings, are confounded with the surface boundary.