EUROGRAPHICS Symposium on Sketch-Based Interfaces and Modeling (2011)
T. Hammond and A. Nealen (Editors)

3D Modeling with a Symmetric Sketch

A. C. Öztireli[1], U. Uyumaz[1], T. Popa[1], A. Sheffer[2] and M. Gross[1]

[1] ETH Zurich, Switzerland - [2] University of British Columbia, Canada

Abstract

We propose a method that allows geometric operations such as view change, deformation, simulation, or symmetrization on a single off-line sketch via a proxy geometry reconstructed directly from the sketch. The reconstruction captures the overall shape of the object depicted by making use of the global relationships of the curves and the assumption that the sketched object is bilaterally symmetric. After cleaning the sketch and extracting the curves, topological and geometric properties of a set of identified points are used to derive robust correspondence and pairwise constraints. These constraints are considered all at once in a spectral algorithm to get the optimum matching of the curves. Depths of points on the matched curves are extracted by utilizing the symmetry assumption. They are then used to reconstruct a smooth geometry. The whole process is automatic except for a few seconds of user interaction.

Categories and Subject Descriptors (according to ACM CCS): Numerical Analysis [G.1.2]: Approximation—Approximation of surfaces and contours; Computer Graphics [I.3.5]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations

1. Introduction

Sketch based 3D modeling has been receiving growing attention due to the intuitive and familiar nature of sketching for humans to communicate 3D information. Although many sketch based interfaces have been proposed, drawing with ordinary paper and pencil still remains to be the most convenient way to draw for artists [OSSJ08]. In addition, there is a vast amount of already drawn sketches in online or private collections. In this work, our goal is to bring 2D sketches to 3D life and manipulate them as if they were in 3D as illustrated in Figure 1.

Applying 3D operations on a sketch requires extracting 3D or 2.5D cues from the sketch. Given a single off-line hand drawn sketch of a 3D object, it is an ill-posed problem to reconstruct the implied geometry due to the ambiguities present [Hof00]. This has led to methods focusing on specific classes [VC07, CKX*08, CSPN10] of objects, which cannot be applied to process general sketches. A more general approach is finding models matching given contours with inflation rules [WH96, KH06, IMT07]. Unfortunately, finding rules that satisfy the needs of the users and consistent geometries is a difficult problem [KH06].

Research on the human visual system revealed that one of the fundamental semantics used to resolve ambiguities in sketches is *symmetry* [TB98, Kon02, MWCP06]. Indeed, it

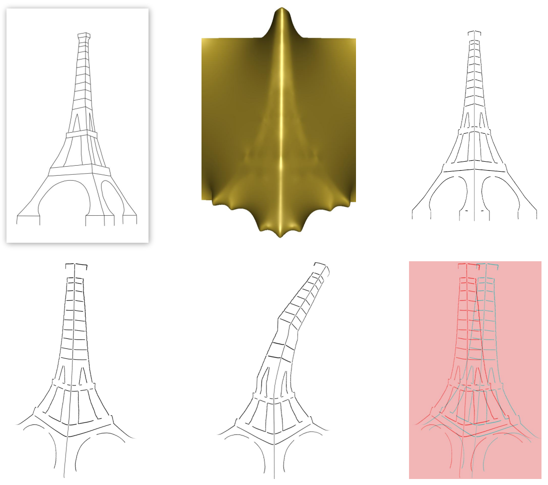

Figure 1: *Top: The input symmetric sketch (left), reconstructed coarse geometry (middle), synthesized symmetric sketch (right). Bottom: a novel view (left), deformation (middle), and an anaglyph image (right).*

has been shown that under orthographic or perspective projection, and certain degrees of symmetry, 3D structure of a shape can be fully recovered [HYHM04] as long as one can determine symmetric pairs of pixels. Many objects that we sketch such as human-made items, buildings, characters,

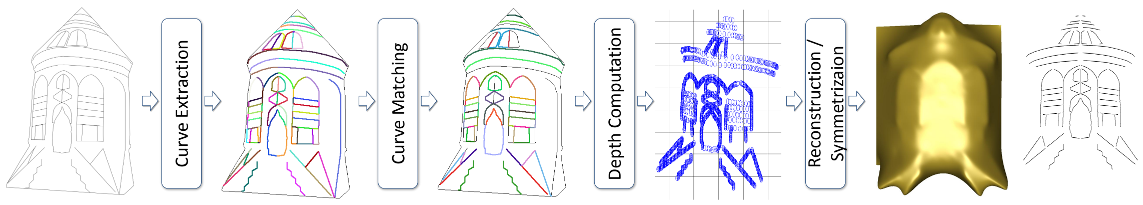

Figure 2: *Overview of our approach.*

faces, and other organic structures exhibit certain degrees of symmetry, hence the symmetry assumption is one of the least restrictive for sketches.

In this paper, we take a single hand-drawn sketch of a bilaterally symmetrical object surface and reconstruct a 3D structure as detailed as the sketch. This is achieved by finding symmetric curve and point pairs, getting the depth values using symmetry, and reconstructing a depth map using these sparse depths. The reconstruction can then be used as a proxy geometry for various sketch processing tasks such as viewing the sketch from a novel view-point, or under a different perspective projection, stereoscopic viewing of the sketch or even applying a 3D deformation to the sketch.

Our main contribution is a curve matching method that can extract symmetric curves in a sketch. We show that it is impossible to find symmetric pairs of points or curves if they are considered in isolation, and thus one should use the *global* structure in the sketch. Hence, we solve the matching problem using global topological and geometric constraints that are considered all at once.

2. Related Work

Shape from symmetry

It has been shown that under various kinds of symmetries and projections, if symmetric pairs of points are known, 3D locations of those points can be recovered uniquely [VP94, FMW02, HYHM04]. Based on this theoretical framework, image-based algorithms that directly reconstruct surfaces [YHR*05], or use the cues given by symmetry to aid in other applications such as photo editing [HHM05] or architectural modeling [JTC09] have been proposed. When images contain rich information about the 3D structure such as texture or shading, one can detect a dense set of symmetric pairs. In contrast, sketches contain much less information about the 3D structure of the depicted object and descriptors cannot be used in identifying symmetric pairs. Even if symmetric pairs are found, they are only on the curves and hence very sparse. Due to these fundamental obstacles, symmetry in sketch-based reconstruction is used only under very restrictive assumptions and for a small set of sketched curves such as planar object contours [CSPN10] or with known symmetric curve pairs [CKX*08, CH08, BBS08]. We overcome these problems by proposing robust curve matching and reconstruction algorithms.

Sketch based 3D reconstruction

Sketch based interfaces for geometric modeling are more intuitive for artists than the 3D tools and thus many methods that try to interpret user strokes to generate 3D reconstructions have been proposed [OSSJ08]. However, most users still find the current interfaces hard to use [GIZ09]. Recent works try to limit the classes of objects to be reconstructed [JTC09, RDI10]. Although this improves both the interaction convenience and the resulting models, reconstructions are limited by the specific classes or databases used. Instead of a full 3D reconstruction, for viewing a sketch or cartoon from different viewpoints, partial 3D information along with specialized stroke interpolation can be used [BCD01,RID10]. However, the results are only suitable for certain styles of depictions. Due to the free form geometry proxy our method reconstructs, we are able to handle a broader set of shapes and modeling tasks.

In addition to interactive sketch based modeling systems, methods to reconstruct 3D content from off-line sketches have been developed. Algorithms have been proposed for reconstructing polyhedral objects with line sketches and under certain assumptions [VC07]. But the set of assumptions limit their applicability for hand drawn sketches of arbitrary objects. For free form sketches, several contour based algorithms to reconstruct 3D surfaces with implied topology have been proposed [WH96, IMT99, KH06]. These algorithms define rules to inflate surfaces out of the contour lines representing depth discontinuities. However, how to choose such rules to get the most intuitive reconstructions is an ill-posed problem without considering further cues from the sketch [KH06].

Although contours at depth discontinuities are fundamental components in a sketch, experimental results show that other curves in a sketch such as the ones at creases, intensity gradients or contours at normal discontinuities are as important for perception [CGL*08, CSD*09]. A classification of contours based on the type of discontinuities that represent a legal shape can be obtained by line labeling algorithms [Mal86]. Labeled lines can then be used in reconstruction methods [KC06]. However, many different legal labellings are possible and they only give where the discontinuities are, not the actual constraints. This causes the reconstruction methods to heavily rely on user input [KC06]. Our method can extract the information present in any class of curves as long as symmetric pairs of curves exist.

Since there can be infinitely many interpretations of a

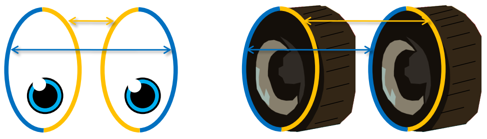
Figure 3: *Inherent ambiguities in symmetric curve matching. Different matchings can lead to different perceptions.*

single sketch [Hof00], many systems incorporate user input such as normals, silhouettes, creases [ZDPSS01], curves with predefined meanings [OS10a], annotations and primitive shapes [GIZ09], or depth inequalities [SSJ*10]. The goal is to keep the interaction intuitive and minimum. Our algorithms are almost fully automatic except requiring the user to click on a few choices presented, which lasts a couple of seconds.

3. Overview

The input to our algorithm is a rasterized sketch. The sketch is first converted into a clean and thinned set of curves using algorithms that operate on the pixels [SSSS01, OS10b] as illustrated in Figure 4 (b). As an optional step, the silhouette is extracted. Since in general, the silhouette of a symmetric object will not be symmetric, it is not used in the matching algorithm unless requested by the user. Curves are extracted from the remaining contours by curve tracing, short curve removal, and branch combination [Pav82].

The next task for our symmetry-based reconstruction algorithm is finding symmetric pairs of curves (Section 4). We accomplish this task by first extracting and matching curve points that have invariant geometric and topological properties under symmetry and orthographic projection. Once symmetric matching curves are found, we compute depths of the points on the curves by utilizing the bilateral symmetry assumption (Section 5). We then fit a 3D proxy geometry using the constructed 3D points and energy terms (Section 6). A schematic view of this pipeline is illustrated in Figure 2.

4. Finding Symmetric Curves and Points

Unlike images, sketches do not contain rich 3D information and thus conventional descriptors fail to give matches [EHBA09]. Finding symmetric curves in a sketch is also an unsolved problem [CSPN10] unless further assumptions such as planarity of curves [HMY04] are made. For orthogonal projections, it is easy to see this is an ill-posed problem, since many curves can match and produce well-defined depth values for the points on them (see Section 5 for an explanation and Figure 3 for an example). Hence, in isolation or locally, no information about matching pairs can be obtained.

To infer matching pairs of curves, we need to define *legal* matchings, by imposing several assumptions on the geometry of the surface and topology of the curves. Research on the human visual system has shown that people tend to interpret close primitives on the image plane as close in the actual scene, and smoothness of curves and surfaces are taken for

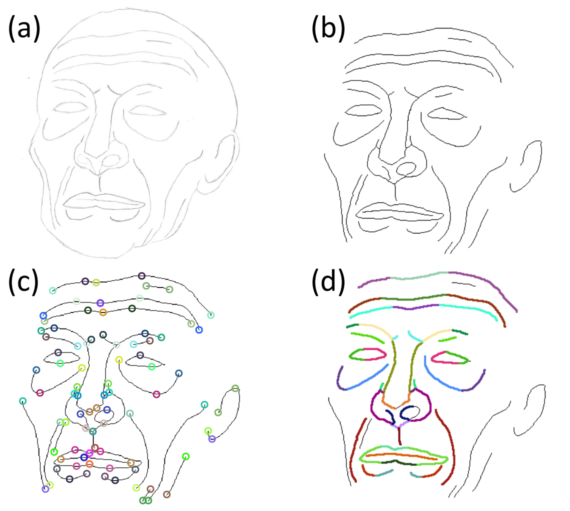

Figure 4: *(a) A hand-drawn and scanned sketch. (b) Extracted curves. (c) Matched check points. (d) Matched curves. Curves with the same colors are matching, while for black curves no match could be found.*

granted if humans see smooth curves in the image [Hof00]. Following these observations, we make the assumption that, given a set of smooth curves close to each other on a smooth surface in a sketch, with high probability they will also be close in the real world (excluding occlusions) and thus for symmetric surfaces, symmetric pairs of curves with the same proximity will be sketched. Using this assumption along with geometric constraints for orthogonal projection allows us to formulate the curve matching problem as a correspondence finding problem, which can be effectively solved by a graph matching method. In general, sketches are not drawn under orthographic projection. However, our matching algorithm is robust to projection, which enables our system to work on typical artist-drawn sketches.

4.1. Check points

In order to infer symmetric pairs of curves, we first identify and match points with invariant properties under symmetry and orthographic projection. One such class of points is the branching points, which are the points where more than one curve meets. In addition to the branching points, some curves will end at isolated points. We use these points in our algorithm as well. Furthermore, if the image plane is rotated such that the projection of the normal of the symmetry plane is parallel to the x axis, then y coordinates of symmetric point pairs will stay the same under orthographic projection (see Section 5). Thus the extrema points on two symmetric curves where the curves change their direction in the y axis should also match for symmetric pairs of curves. We call these three classes of points as *check points* (see Figure 5 and Figure 4 (c)).

Symmetric pairs of check points will have the same y coordinates (after the image plane is rotated according to the

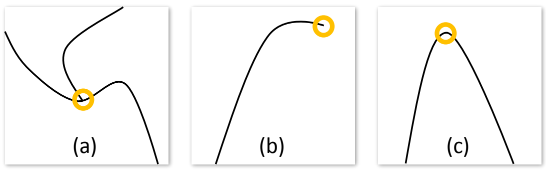

Figure 5: *Definition of check points as (a) branching points where more than one curve meets, (b) points where curves end, or (c) extreme points of curves.*

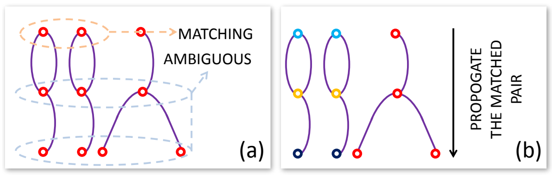

Figure 6: *(a) Single point matching can lead to ambiguities that can be solved by (b) propagating the information from the matched pair.*

symmetry plane normal, see Section 5) and the same number of curves in the proximity, excluding the cases of occlusions or depth discontinuities. They are also easy to extract from the sketch. We utilize these properties in our matching algorithm.

4.2. Features for check points

A feature vector for a check point captures the geometric and topological properties of that point that are invariant under symmetry and projection. It should also be robust to the impreciseness and noise in the sketches. This vector can be used as a signature of that point in a matching algorithm. In the following discussion, we assume that one can detect depth discontinuities and occlusions (for example via line labeling algorithms [Mal86]) and avoid the check points resulting from those. Under this assumption, there are two invariants to be considered for a symmetric pair of check points. The first one is the number of curves connected to or in a proximity of a check point, and the second one is the y coordinate of it.

Due to the imprecise nature of sketches, noise, and artifacts introduced by preprocessing, exact connections will most probably not be available. We define the probability p_i^t of a check point $\mathbf{x}_i$ being connected to a curve c_t as $p_i^t = exp\{-(d_i^t/\sigma_c)^2\}$, where d_i^t is the minimum of the distances of the endpoints for the curve c_t to the check point $\mathbf{x}_i$. We form the vector $\mathbf{p}_i = [p_i^1 \cdots p_i^{n_c}]^T$, and also sort (in descending order) and stack all probabilities into another vector $\bar{\mathbf{p}}_i$, where n_c is the number of curves.

Similar to the number of close-by curves, the y components will also not be exactly the same due to impreciseness of the artist or noise. Thus we again define a Gaussian model for the probability that y_i and $y_{i'}$ are the same as $r_{ii'} = exp\{-((y_i - y_{i'})/\sigma_y)^2\}$.

In our experiments, we found that the matching algorithm is very robust to different values of σ_c and σ_y. For all the results in this paper, we set $\sigma_c = \sigma_y = 10$. We also tested different values in the range of 5 to 30 and in all of the cases, there were no wrong matching pairs.

4.3. Feature comparisons and pairwise constraints

We define the confidence of a pair of check points $(\mathbf{x}_i, \mathbf{x}_{i'})$ as follows:

$$s(i,i') = r_{ii'}\,(1 - \|\bar{\mathbf{p}}_i - \bar{\mathbf{p}}_{i'}\|/n_{mc}) \tag{1}$$

with n_{mc} the maximum number of curves in a neighborhood of radius of $2.5\sigma_c$ around each $\mathbf{x}_i$. Note that in practice we use only the curves in the neighborhood of the check points in the definition of $\bar{\mathbf{p}}_i$'s since the Gaussian is effectively zero outside of the neighborhood. This also means the measure is always between 0 and 1. It increases as the y coordinates or curve proximity probabilities of two check points become more similar.

Although this measure captures similarity between check points, it cannot account for the constraints that force consistent connections between check points. If there are pairs $(\mathbf{x}_i, \mathbf{x}_{i'})$ and $(\mathbf{x}_j, \mathbf{x}_{j'})$ of check points, there should be a curve between $\mathbf{x}_i$ and $\mathbf{x}_j$ and another curve between $\mathbf{x}_{i'}$ and $\mathbf{x}_{j'}$, or a curve between $\mathbf{x}_i$ and $\mathbf{x}_{j'}$ and another one between $\mathbf{x}_{i'}$ and $\mathbf{x}_j$. For this constraint, we define the following measure:

$$s(i,i';j,j') = s_y(i,i';j,j')s_c(i,i';j,j') \tag{2}$$

$$s_y(.) = r_{ii'}r_{jj'}$$

$$s_c(.) = \max\left(\max_t(p_i^t p_j^t)\max_t(p_{i'}^t p_{j'}^t), \max_t(p_i^t p_{j'}^t)\max_t(p_{i'}^t p_j^t)\right)$$

This score measures how compatible two pairs are in terms of the curves between them. The first term s_y ensures that the ends of the two curves have similar y coordinates, and the term s_c measures the odds of having a curve between both $(\mathbf{x}_i, \mathbf{x}_j)$ and $(\mathbf{x}_{i'}, \mathbf{x}_{j'})$, or both $(\mathbf{x}_i, \mathbf{x}_{j'})$ and $(\mathbf{x}_{i'}, \mathbf{x}_j)$. These two cases come from the ambiguous labeling of the pairs.

We illustrate the power of the pairwise constraints in Figures 6 and 7. In Figure 6 (a), it is impossible to find the matching check points on the middle level since all points have the same number of connected curves. The same is true for the check points on the bottom level. In contrast, there can be only one matching pair in the top level. By propagating this information to the lower levels, an algorithm can easily find the correct matches (Figure 6 (b)). In Figure 7, we illustrate this effect on a real example computed by our matching algorithm (see Section 4.4). The wrong pairs in Figure 7 (a) are replaced by the correct ones in Figure 7 (b) by making the algorithm aware of the global structure of the sketch via the pairwise constraints.

4.4. The matching algorithm

We formulate the curve matching problem as a correspondence finding problem among the check points. Once correspondences are found, matching curves can be extracted by considering the check points at their ends.

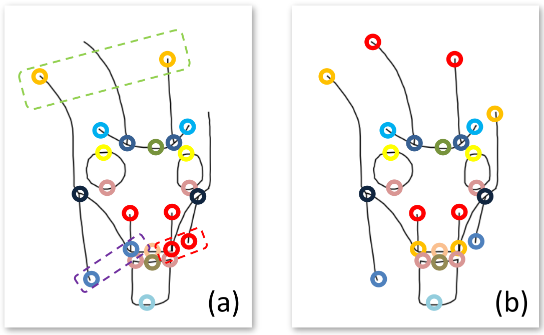

Figure 7: *(a) Without the pairwise constraints, there are several wrong matches, and for some check points no matches are found. (b) By considering the global structure using the pairwise constraints, all matches are found correctly.*

For our problem, we do not have two distinct sets of points to match, thus we should avoid matching each point to itself. More importantly, we need to consider pairwise constraints. To satisfy these requirements, we use a spectral technique to find correspondences [LH05]. The method uses the adjacency matrix of a graph, where the nodes are given by the assignments $a = (i, j)$ of the pairs of check points $(\mathbf{x}_i, \mathbf{x}_j)$ and the weights are determined by how much two pairs agree with each other. Specifically, a matrix $\mathbf{M}$ is built, where $\mathbf{M}_{ab}$ gives the agreement score of the pairs a and b, and $\mathbf{M}_{aa}$ is the score of the pair a. Higher score means higher agreement.

Matching each check point to itself can be avoided by disallowing the pairs $(\mathbf{x}_i, \mathbf{x}_i)$. This means the matrix $\mathbf{M}$ will not contain the corresponding rows and columns. The only exception to this rule is when a check point is on the symmetry plane. We request the user to tag such check points. Pairwise constraints are also naturally integrated into the approach. Letting $a = (i, i')$ and $b = (j, j')$, we set the entries of the matrix as $\mathbf{M}_{aa} = s(i, i')$ or $\mathbf{M}_{aa} = 1$ if $\mathbf{x}_i = \mathbf{x}'_i$ is on the symmetry plane, and $\mathbf{M}_{ab} = s(i, i'; j, j')$.

Once $\mathbf{M}$ is constructed, the principle eigenvector is computed and its components are sorted. Iteratively, the following procedure is applied: remove the current highest component of the principle eigenvector and put the corresponding pair into the set of accepted pairs; remove the components of the vector corresponding to the pairs which are incompatible with the accepted pairs. Since our problem requires a one-to-one matching of the check points, the compatibility condition for our case is: if a pair $a = (\mathbf{x}_i, \mathbf{x}_j)$ is accepted, remove all pairs containing the points $\mathbf{x}_i$ or $\mathbf{x}_j$. The algorithm terminates when the current component is zero or when all components are exhausted.

After the algorithm is terminated, some check points will have unique corresponding check points, and some will be discarded (Figure 8). Two curves are matching, if checkpoints at their ends form pairs. Curves with inconsistent end point pairs are eliminated (Figure 4 (d)). Some inherent ambiguities might still exist after the matching algorithm terminates, such as the case illustrated in Figure 3. In these cases, we ask the user to choose among the set of possible matching curves computed. After all matching curves are found, each point on one curve is matched to the point with the same y coordinate on the matching curve (see Section 5).

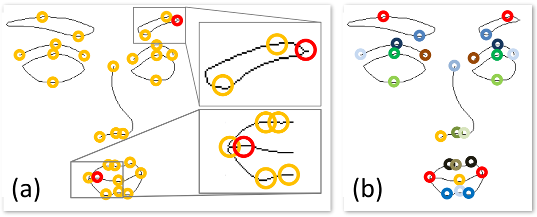

Figure 8: *(a) Due to the thinning algorithm used in the preprocessing of the raw sketch, or the impreciseness of the artist, there appears extra check points (red points). (b) These are eliminated and do not affect the correct matching of the rest of the points.*

4.5. The Matching Pipeline

In summary, we have the following steps for the matching algorithm:

1. The user selects a pair of matching points and the sketch is rotated accordingly.
2. He clicks on the check points on the symmetry plane.
3. All check points are matched by the algorithm.
4. Matching curves are extracted using the matched check points. If there are any ambiguous cases, they are resolved by the user.
5. Points on the matching curves with the same y coordinates are matched to each other.

5. Shape from Symmetry

Once symmetric pairs of points are found, it is known that a symmetric object can be reconstructed from a single view up to certain degrees of freedom depending on the projection type (orthographic or perspective) and the number of symmetries present [VP94, FMW02, HYHM04]. We observe that one can define the condition on 3D locations of points implied by bilateral symmetry just by considering its actual geometric meaning. Let us denote a pixel by $\mathbf{x} = [x\ y\ 0]^T$ such that image plane is aligned with the xy plane and is residing at $z = 0$. Further assume a general projection model is given by the ray field $\mathbf{d}(\mathbf{x})$. Then each point $\mathbf{y}$ on an object is given by $\mathbf{y} = \mathbf{x} + t\mathbf{d}(\mathbf{x})$ for some pixel $\mathbf{x}$. Suppose we have the symmetric pair $\mathbf{y}$ and $\mathbf{y}'$, and that the normal of the symmetry plane is denoted by $\mathbf{n}$. Without loss of generality, we can assume that the plane passes through the origin. With these notations, the following equation holds:

$$\mathbf{y}' = (\mathbf{I} - 2\mathbf{n}\mathbf{n}^T)\mathbf{y} \tag{3}$$

Plugging in the expressions for $\mathbf{y}$ and $\mathbf{y}'$, we can arrive at a linear system for the unknowns t and t'.

In addition to the inherent inaccuracies of hand-drawing, artists often use, intentionally or unintentionally, distorted perspective when they sketch. Thus a parametric model for the camera is very hard to extract from a sketch. However, when they sketch *small* characters and objects, they tend to use almost orthographic projections. Hence, unless the sketch is describing a large object or it is stylized to use exaggerated or general perspective, we can assume that an almost orthographic projection is used.

In an orthographic projection, the rays are described by $\mathbf{d}(\mathbf{x}) = \hat{z}$ and equation 3 can be solved exactly for the z components of $\mathbf{y}$ and $\mathbf{y}'$ (which are t and t') as

$$t = \frac{-\hat{\mathbf{n}}_p^T(\mathbf{x}+\mathbf{x}') + 2n_z^2\hat{\mathbf{n}}^T\mathbf{x}}{2n_z\sqrt{1-n_z^2}} \qquad (4)$$

$$t' = \frac{-\hat{\mathbf{n}}_p^T(\mathbf{x}+\mathbf{x}') + 2n_z^2\hat{\mathbf{n}}^T\mathbf{x}'}{2n_z\sqrt{1-n_z^2}}$$

Here, $\mathbf{n}_p$ is the projection of $\mathbf{n}$ onto the image plane (i.e. $[n_x\ n_y\ 0]^T$). In addition to these equations, equation 3 also implies that the vector $\mathbf{x}' - \mathbf{x}$ is parallel to $\mathbf{n}_p$. In practice, we rotate the sketch such that $\mathbf{n}_p$ is parallel to the x axis so that $\hat{\mathbf{n}}_p = \pm\hat{x}$. This implies that the y coordinates of the symmetric points will be the same since $\mathbf{x}' - \mathbf{x}$ is parallel to $\hat{x}$. This equation also means that every two pairs of symmetric points will produce meaningful depth values as long as $\mathbf{x}' - \mathbf{x}$ is parallel to $\hat{\mathbf{n}}_p$. Hence, it is impossible to know the symmetric pair of a given point, or symmetric curve of a given curve, just by considering the coordinates or shape of it. Therefore one needs to take the global shape information into account to get symmetric pairs.

6. 3D Reconstruction

Given a sparse set of points with depths, we would like to reconstruct a smooth 3D surface that interpolates these points. Due to the sparseness of the depths, imprecise nature of the curves, lack of normals and information about the unsketched parts of the object, we chose to reconstruct the 3D geometry as a depth map over the image plane. We formulate the problem as an energy minimization with constraints and propose to use a continuous formulation [NWT10], with energy terms from surface approximation and deformation literature that are known to produce tight and smooth reconstructions.

We propose to discretize the energy and minimize it along with other terms and the constraints using compactly supported radial basis functions centered at each pixel, to get a continuous function for the depth map. Although this formulation results in a big linear system to solve for the coefficients, it is efficiently solvable thanks to the extreme sparseness. Once the coefficients are computed, each evaluation of the function at a pixel involves only a small set of neighboring pixels around the given pixel.

6.1. Function representation and constraints

We represent the depth map function as a sum of compactly supported radial basis functions $k(\mathbf{x},\mathbf{x}_i) = e^{-||\mathbf{x}-\mathbf{x}_i||^2/\sigma^2}$ such that $f(\mathbf{x}) = \sum f_i k(\mathbf{x},\mathbf{x}_i)$ for pixels $\mathbf{x}_i$. In most applications, σ determines the level of smoothness of f. Although bigger σ gives smoother results, it results in much higher computation times. Since we can determine the smoothness of f through the energy terms, we fixed $\sigma = 1.2$ such that a three-ring neighborhood of each pixel is covered by the effective support of the Gaussian (assuming the size of a pixel is 1). To make $f(\mathbf{x})$ interpolate the known depth values, we should have $f(\mathbf{x}_i^d) = d_i$ where we denote the points (pixels) with depth values by $\mathbf{x}^d$. By writing the expression for $f(\mathbf{x}_i^d) = \sum f_j k(\mathbf{x}_i^d, \mathbf{x}_j) = d_i$, and gathering all equations, we get a linear system $\mathbf{Kf} = \mathbf{d}$, where $\mathbf{K}$ is an n_d by n matrix with n_d denoting the number of depth values and n the number of pixels.

6.2. Energy terms and minimization

We discretize the thin plate spline energy with the following sum: $\sum_k f_{xx}^2(\mathbf{x}_k) + 2f_{xy}^2(\mathbf{x}_k) + f_{yy}^2(\mathbf{x}_k)$. Minimization of this sum leads to a quadratic form $\mathbf{f}^T\mathbf{Tf}$, where $\mathbf{T}_{ij} = \sum_k k_{i,xx}(\mathbf{x}_k)k_{j,xx}(\mathbf{x}_k) + k_{i,xy}(\mathbf{x}_k)k_{j,xy}(\mathbf{x}_k) + k_{i,yy}(\mathbf{x}_k)k_{j,yy}(\mathbf{x}_k)$ and the subscripts denote derivatives.

Similarly, we discretize a specific case of the Sobolev norm, $\int ||\nabla f(\mathbf{x})||^2$, which is also known as an approximation to the membrane or stretching energy [TPBF87, CG91]. This leads to another quadratic form $\mathbf{f}^T\mathbf{Bf}$ with $\mathbf{B}_{ij} = \sum_k \nabla k_i(\mathbf{x}_k)^T \nabla k_j(\mathbf{x}_k)$.

We gather the terms arising from the constraints and the energies as follows:

$$E(\mathbf{f}) = ||\mathbf{Kf} - \mathbf{d}||^2 + \mathbf{f}^T(\lambda_T\mathbf{T} + \lambda_B\mathbf{B})\mathbf{f} \qquad (5)$$

By solving the linear system resulting from this minimization, different continuous functions can be defined depending on the parameters used. For all results in this paper, we set $\lambda_T = 3$ and $\lambda_B = 1$.

7. Results

We ran our method on various sketches with different amount of noise and artifacts. For each sketch, first a depth map and a new sketch with the symmetry plane orthogonal to the image plane is constructed. This sketch is used as a texture to the reconstructed geometry. To get continuous curves out of the sparse set of rotated points with depths, a B-Spline curve is fit to each reconstructed curve. The geometry is in the form of a dense mesh with pixels as vertices. For rendering and other geometric operations this dense mesh can be simplified significantly. The resulting textured models are rendered using flat shading in order to show the sketch.

The matching algorithm is robust to impreciseness, artifacts, perspective distortion, and extra curves. An example matching with very different y coordinates is given in Figure 7. In Figures 4 (c) and 8, although there are many close points with similar neighborhoods and y coordinates, the matching algorithm is able to find the correct matches thanks to the pairwise constraints. It also eliminates some of the check points without matches. Even if some wrong matches of check points persist, they fail to form a consistent labeling for the curve matching step. Thus the algorithm correctly identifies curves without matches, as illustrated in Figure 4.

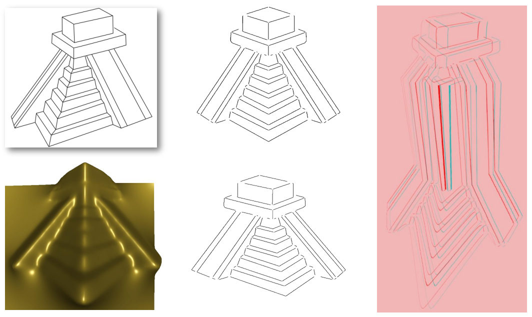

Figure 9: *Left: An input sketch and its reconstruction. Middle: The sketch from different views. Right: An anaglyph image of a deformation of the sketch.*

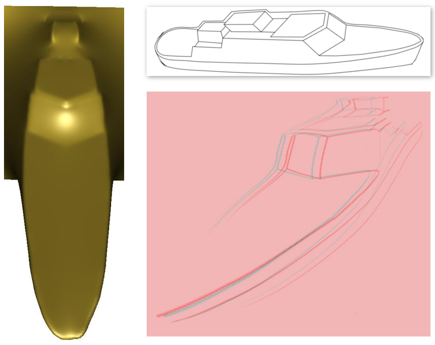

Figure 10: *An input sketch, its reconstruction, and an anaglyph image. Note that for this anaglyph image, the model was reconstructed without rotating the points with depth values, to have the original sketch as the texture onto the reconstruction.*

Several reconstruction, view change and stereoscopic imaging results are shown in Figures 1, 2, 9, and 10. The reconstructed models capture the overall shape of the depicted objects well with correct curvatures. The sketch-textured depictions with the reconstructions match our intuition of the sketch faithfully as illustrated by the rotated views of the sketches. Having a depth map allows us to construct stereoscopic images of the sketches from different angles. It is interesting to see that, even without any 3D cues from shading, a good sense of depth can be obtained from these images.

Using the reconstructed models, one can also deform the geometry or perform physically based simulations as in Figures 1, 9, 11, and 12. The level of deformations is limited by the resolution of the reconstructed mesh and texture. This problem can be solved by using procedural textures, that is, using the B-Spline curves directly as textures.

8. Conclusions

In this paper, we proposed a new method for 3D modeling using a single 2D sketch depicting a symmetric object. Several modeling operations are demonstrated such as novel viewpoint and stereo renderings, and geometric and physically based deformations. 3D manipulation is done via a proxy 3D geometry reconstructed only from a single off-line sketch nearly automatically. Since many man-made and natural objects are bilaterally symmetric, we are able to handle a broad spectrum of object sketches.

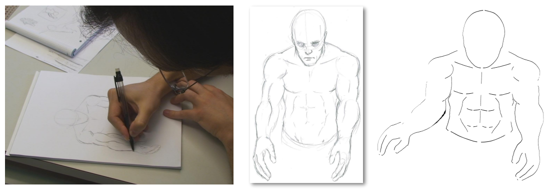

Figure 11: *A sketch drawn by an artist with paper and pencil is scanned, cleaned, and deformed using our method. Note that some details are removed in the cleaning process.*

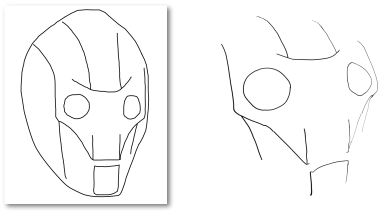

Figure 12: *An input sketch and a frame from its physically based simulation.*

Limitations One limitation of our approach is that it requires symmetric pairs in the sketch. Hidden pairs lead to incomplete reconstructions. This can be eliminated by combining curves from sketches from multiple viewpoints or letting the user draw the non-existent symmetric curves. Another limitation of the algorithm stems from the geometric ambiguity of horizontal segments. For a pair of matching horizontal lines, finding matching points is impossible since all have the same y coordinates. For these cases, we currently assume that the middle points of all pairs on the horizontal segments are the same.

Future Directions Our algorithm can be extended for intrinsic and continuous symmetries, and hidden lines, combined with more sophisticated preprocessing. It can provide partial or full depth information that can aid in many applications requiring extracting 3D or 2.5D content from sketches such as sketch based retrieval, dynamic 3D content creation, and interactive sketch-based modeling interfaces.

Acknowledgments

We would like to thank Maurizio Nitti for providing us with sketches. Cengiz Öztireli is supported by the Swiss National Science Foundation (grant No. 200021-132438 / 1). Tiberiu Popa is partially supported by NSERC.

References

[BBS08] BAE S.-H., BALAKRISHNAN R., SINGH K.: Iloveseketch: as-natural-as-possible sketching system for creating 3d curve models. In *UIST '08* (New York, NY, USA, 2008), UIST '08, ACM, pp. 151–160. 2

[BCD01] Bourguignon D., Cani M. P., Drettakis G.: Drawing for Illustration and Annotation in 3D. In *EG 2001 Proc.*, Chalmers A., Rhyne T. M., (Eds.), vol. 20(3). Blackwell Publishing, 2001, pp. 114–122. 2

[CG91] Celniker G., Gossard D.: Deformable curve and surface finite-elements for free-form shape design. SIGGRAPH '91, ACM, pp. 257–266. 6

[CGL*08] Cole F., Golovinskiy A., Limpaecher A., Barros H. S., Finkelstein A., Funkhouser T., Rusinkiewicz S.: Where do people draw lines? In *ACM SIGGRAPH 2008 papers* (New York, NY, USA, 2008), SIGGRAPH '08, ACM, pp. 88:1–88:11. 2

[CH08] Cheon S.-U., Han S.: A template-based reconstruction of plane-symmetric 3d models from freehand sketches. *Comput. Aided Des. 40* (September 2008), 975–986. 2

[CKX*08] Chen X., Kang S. B., Xu Y.-Q., Dorsey J., Shum H.-Y.: Sketching reality: Realistic interpretation of architectural designs. *ACM Trans. Graph. 27* (May 2008), 11:1–11:15. 1, 2

[CSD*09] Cole F., Sanik K., DeCarlo D., Finkelstein A., Funkhouser T., Rusinkiewicz S., Singh M.: How well do line drawings depict shape? SIGGRAPH '09, ACM, pp. 28:1–28:9. 2

[CSPN10] Cordier F., SEO H., PARK J., NOH J.: Sketching of mirror-symmetric shapes. *IEEE Transactions on Visualization and Computer Graphics* (2010). accepted for publication. 1, 2, 3

[EHBA09] Eitz M., Hildebrand K., Boubekeur T., Alexa M.: A descriptor for large scale image retrieval based on sketched feature lines. In *SBIM '09* (New York, NY, USA, 2009), SBIM '09, ACM, pp. 29–36. 3

[FMW02] Francois A. R. J., Medioni G. G., Waupotitsch R.: Reconstructing mirror symmetric scenes from a single view using 2-view stereo geometry. *Pattern Recognition, International Conference on 4* (2002), 40012. 2, 5

[GIZ09] Gingold Y., Igarashi T., Zorin D.: Structured annotations for 2d-to-3d modeling. *ACM Trans. Graph. 28* (December 2009), 148:1–148:9. 2, 3

[HHM05] Huang K., Hong W., Ma Y.: Symmetry-based photo-editing. *Pattern Recogn. 38* (June 2005), 825–834. 2

[HMY04] Hong W., Ma Y., Yu Y.: Reconstruction of 3-d symmetric curves from perspective images without discrete features. In *in: Proc. Eur. Conf. on Computer Vision* (2004). 3

[Hof00] Hoffman D. D.: *Visual Intelligence: How We Create What We See*. W. W. Norton & Company, February 2000. 1, 3

[HYHM04] Hong W., Yang A. Y., Huang K., Ma Y.: On symmetry and multiple-view geometry: Structure, pose, and calibration from a single image. *Int. J. Comput. Vision 60* (December 2004), 241–265. 1, 2, 5

[IMT99] Igarashi T., Matsuoka S., Tanaka H.: Teddy: a sketching interface for 3d freeform design. In *SIGGRAPH '99* (New York, NY, USA, 1999), SIGGRAPH '99, ACM Press/Addison-Wesley Publishing Co., pp. 409–416. 2

[IMT07] Igarashi T., Matsuoka S., Tanaka H.: Teddy: a sketching interface for 3d freeform design. In *ACM SIGGRAPH 2007 courses* (New York, NY, USA, 2007), SIGGRAPH '07, ACM. 1

[JTC09] Jiang N., Tan P., Cheong L.-F.: Symmetric architecture modeling with a single image. In *ACM SIGGRAPH Asia 2009 papers* (New York, NY, USA, 2009), SIGGRAPH Asia '09, ACM, pp. 113:1–113:8. 2

[KC06] Kaplan M., Cohen E.: Producing models from drawings of curved surfaces. In *SBIM'06* (2006). 2

[KH06] Karpenko O. A., Hughes J. F.: Smoothsketch: 3d free-form shapes from complex sketches. In *ACM SIGGRAPH 2006 Papers* (New York, NY, USA, 2006), SIGGRAPH '06, ACM, pp. 589–598. 1, 2

[Kon02] Kontsevich L. L.: Symmetry as a depth cue. *Human Symmetry Perception and Its Computational Analysis* (2002), 331–347. 1

[LH05] Leordeanu M., Hebert M.: A spectral technique for correspondence problems using pairwise constraints. In *ICCV'05* (2005), pp. 1482–1489. 5

[Mal86] Malik J.: *Interpreting line drawings of curved objects*. PhD thesis, Stanford, CA, USA, 1986. UMI order no. GAX86-12762. 2, 4

[MWCP06] M. W. Chan A. K. Stevenson Y. L., Pizlo Z.: Binocular shape constancy from novel views: the role of a priori constraints. *Perception and Psychophysics 68*, 7 (2006), 1124–1139. 1

[NWT10] Ng H.-S., Wu T.-P., Tang C.-K.: Surface-from-gradients without discrete integrability enforcement: A gaussian kernel approach. *PAMI 32* (2010), 2085–2099. 6

[OS10a] Olsen L., Samavati F. F.: Image-assisted modeling from sketches. In *Proceedings of Graphics Interface 2010* (Toronto, Ont., Canada, Canada, 2010), GI '10, pp. 225–232. 3

[OS10b] Olsen L., Samavati F. F.: Stroke extraction and classification for mesh inflation. In *SBIM'10* (2010), pp. 9–16. 3

[OSSJ08] Olsen L., Samavati F., Sousa M., Jorge J.: A taxonomy of modeling techniques using sketch-based interfaces. In *EG'08 STAR* (2008). 1, 2

[Pav82] Pavlidis T.: *Algorithms for Graphics and Image Processing*. Springer, 1982. 3

[RDI10] Rivers A., Durand F., Igarashi T.: 3d modeling with silhouettes. In *ACM SIGGRAPH 2010 papers* (New York, NY, USA, 2010), SIGGRAPH '10, ACM, pp. 109:1–109:8. 2

[RID10] Rivers A., Igarashi T., Durand F.: 2.5d cartoon models. *ACM Trans. Graph. 29* (July 2010), 59:1–59:7. 2

[SSJ*10] Sýkora D., Sedláček D., Jinchao S., Dingliana J., Collins S.: Adding depth to cartoons using sparse depth (in)equalities. *Computer Graphics Forum 29*, 2 (2010), 615–623. 3

[SSSS01] Shapiro L. G., Stockman G. C., Shapiro L. G., Stockman G.: *Computer Vision*. Prentice Hall, 2001. 3

[TB98] Troje N. F., Bulthoff H. H.: How is bilateral symmetry of human faces used for recognition of novel views? *Vision Research 38*, 1 (1998), 79 – 89. 1

[TPBF87] Terzopoulos D., Platt J., Barr A., Fleischer K.: Elastically deformable models. *SIGGRAPH Comput. Graph. 21* (August 1987), 205–214. 6

[VC07] Varley P., Company P.: Sketch input of 3d models: Current directions. In *VISAPP 2007* (2007), pp. 85ï£¡–91. 1, 2

[VP94] Vetter T., Poggio T.: Symmetric 3d objects are an easy case for 2d object recognition. *Spatial Vision 8*, 4 (1994), 443–453. 2, 5

[WH96] Williams L. R., Hanson A. R.: Perceptual completion of occluded surfaces. *Comput. Vis. Image Underst. 64* (July 1996), 1–20. 1, 2

[YHR*05] Yang A. Y., Huang K., Rao S., Hong W., Ma Y.: Symmetry-based 3-d reconstruction from perspective images. *Comput. Vis. Image Underst. 99* (August 2005), 210–240. 2

[ZDPSS01] Zhang L., Dugas-Phocion G., Samson J.-S., Seitz S. M.: Single view modeling of free-form scenes. *CVPR 1* (2001), 990. 3