Expressive-WICED
C. E. Catalano and A. D. Parakkat (Editors)

PerceptualLift: Using hatches to infer a 3D organic shape from a sketch

Tara Butler [1], Pascal Guehl[1], Amal Dev Parakkat[2] and Marie-Paule Cani[1]

[1] LIX - Ecole Polytechnique/CNRS, [2] LTCI - Telecom Paris
Institut Polytechnique de Paris, France

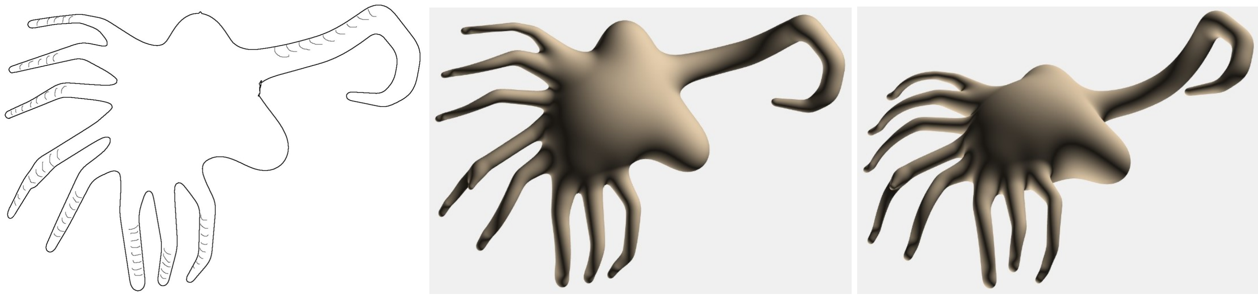

Figure 1: *A sample shape modeled using PerceptualLift. Left to right: 2D sketch input along with user-drawn hatches, front view and a side view of the resulting 3D shape.*

Abstract

In this work, we investigate whether artistic hatching, popular in pen-and-ink sketches, can be consistently perceived as a depth cue. We illustrate our results by presenting PerceptualLift, a modeling system that exploits hatching to create curved 3D shapes from a single sketch. We first describe a perceptual user study conducted across a diverse group of participants, which confirms the relevance of hatches as consistent clues for inferring curvature in the depth direction from a sketch. It enables us to extract geometrical rules that link 2D hatch characteristics, such as their direction, frequency, and magnitude, to the changes of depth in the depicted 3D shape. Built on these rules, we introduce PerceptualLift, a flexible tool to model 3D organic shapes by simply hatching over 2D hand-drawn contour sketches.

CCS Concepts

• ***Computing methodologies*** $\rightarrow$ ***Shape modeling;*** • ***Human-centered computing*** $\rightarrow$ ***Interactive systems and tools;***

1. Introduction

Using 2D sketches to illustrate 3D shapes is a long-standing practice among artists, scientists, designers, and engineers. The perception of the third dimension is usually conveyed by using various shading techniques. While tonal gradients are typically used in artistic sketches, hatches are preferred in academic illustrations due to their ability to accurately depict the volume and surface properties. As shown in Figure 2, these hatches have been an integral part of 3D illustrations for centuries. For viewers, the sketches capture the object silhouette, whereas the hatches encapsulate the shape's internal geometry and curvature.

Thanks to its ability to convey rich visual information, hatching was already explored in Computer Graphics, but only in the context of non-photorealistic artistic rendering [SS10, PHWF01]. However, while the innate ability of humans allows them to infer quite well the underlying 3D surface geometry from hatches, the usage of hatches as cues for 3D modeling was not studied in the literature. The focus of this work is, therefore, on two key aspects: understanding how humans perceive hatches and use them to interpret 3D shapes and then incorporating this perceptual knowledge into a 3D modeling framework. The ultimate goal is to help the general public use these hatches as a simple way to annotate 2D sketches, which the system can use to automatically infer a 3D model - like the example shown in Figure 1.

The main challenge here is the lack of formal rules in interpreting and understanding hatches. On this front, we first conducted a

Proceedings published by Eurographics - The European Association for Computer Graphics.

DOI: 10.2312/exw.20251055

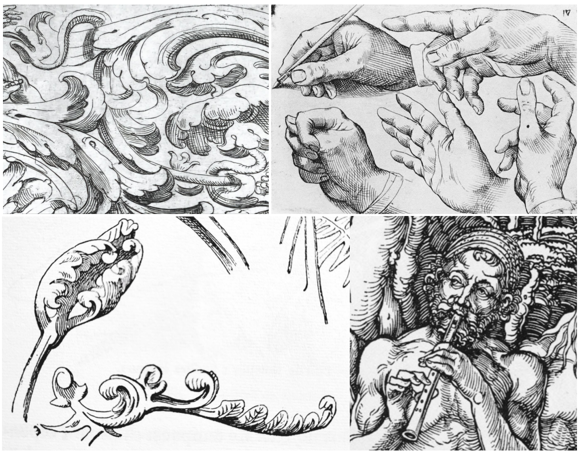

Figure 2: *Examples of drawings using hatching techniques. Top: Odoardo Fialetti (1608), Bottom left: Viollet-le-Duc (1873), Bottom right: Albrecht Dürer (1498).*

detailed perceptual study involving both artists and non-artist participants to explore how 2D sketches with hatching influence depth perception. From this study, we infer a set of rules that relate the characteristics of hatches (e.g., curvature, direction, etc) to the perceived depth of the depicted 3D shape. These rules are then used to develop a computational model, PerceptualLift, used to complement a sketch-based modeling system. In our implementation, we combine it with Matisse [BPCB08], which relies on implicit surfaces to inflate organic shapes from contours. As our results show, PerceptualLift enables users to interactively edit the depth variations of the generated 3D shape through simple 2D annotations.

In summary, we make two scientific contributions:

- **A Perceptual Study**, allowing us to extract general rules to interpret hatching direction, density, and length as information on shape depth variations relative to the local thickness of the depicted shape.
- **An effective Modeling Tool** featuring seamless hatching placement and design coupled with automatic rule interpretation for the interactive creation of free-form, 3D shapes.

2. Related Work

Artists have been using hatching for centuries as a way to represent depth and shape in illustrations. In contrast, the Computer Graphics community has primarily used hatching for non-photorealistic rendering (NPR). In particular, the focus of our work on the usage of hatches for sketch-based 3D modeling was not previously studied. We, therefore, classify the related works in two categories:

Hatching for Non-Photorealistic Rendering (NPR) The earlier works in this direction used hatches as a texture to create a real-time hatching effect with spatial and temporal coherence [PHWF01]. While this work was extended to accommodate dynamic and specular surfaces [KYYL08], alternative techniques tried geometric properties such as curvature directions to directly generate hatches in object space [HZ00, RK00, ZISS04]. Even though such techniques improved the structural consistency, their results often appeared synthetic. To address this limitation and to reproduce stylistic properties and variability as in hand-drawn hatchings, researchers began to explore learning-based techniques. Initial works in this direction focused on learning hatching styles [KNBH12] from single illustrations, which subsequently expanded to explore the possibility of patch-based learning [GI13]. The application of hatching techniques extended beyond these primary focuses, finding its usage in rendering point clouds [WWL*22], animations [LMHB00], and shape-preserving 2D transformations [BPCB08]. It is also worth noting that the placement of strokes [CGL*08] and hatch styles [PK19] and their role in conveying the shapes were also studied in the past. While these works focus on NPR and stylizations, our goal is completely different and is to use hatches to interactively model 3D shapes.

Sketch-Based 3D Modeling As there is a vast literature in this direction, we restrict to the most relevant ones here and refer interested readers to various existing surveys [OSSJ09, BC20]. Starting from the seminal work of Teddy [IMT06], various sketch-based 3D modeling interfaces were introduced to convert free-form 2D sketches to 3D models. Due to its widespread attention, many works followed Teddy, focusing on different challenges - resulting in systems including those designed for cel animation [Joh02], implicit 3D modeling [BPCB08], image-based modeling [CZS*13], animation [DSC*20] and modeling from complex sketches [KH06]. However, works in this direction are usually restricted to simple inflation without the concept of bending out of the plane - the main focus of *PerceptualLift*. The traditional inflation techniques are then extended to consider out-of-plane bending to create complex models by taking user annotations [GIZ09] and reference 3D scene [KYC*17]. Instead of explicitly bending the regions, works like RigMesh [BJD*12] and its companions [ZYC*22] concentrated on creating complex 3D models by following a part-based modeling strategy that mimics the out-of-plane bending. It is also worth noting that a few works explored the possibility of directly inferring 3D models from 2D sketches [SBSS12, XCS*14]. However, these works require heavy user interaction, are not particularly designed for novice users, and cannot capture intricate depth variations.

Silhouette-based techniques such as FiberMesh [NISA07], 3D Modeling with Silhouettes [RDI10], and Silhouette and Stereo Fusion for 3D Object Modeling [ES04] reconstruct 3D shapes primarily from external contours. While these approaches effectively capture overall geometry, they often require multiple views or additional data to infer depth accurately. In contrast, *PerceptualLift* uses internal hatching strokes within a single sketch to deduce depth variations, enabling the creation of complex, non-flat 3D models from a single viewpoint.

Different from traditional sketch-based 3D modeling, recent techniques explored the possibility of using prior knowledge for inferring sketches to create 3D models, working on different types of inputs - single view [ZGG21] and multi-view [DAI*18] in automatic [HWX*21] and interactive [LPL*18] ways. Please refer to the recent survey [XHY*22] for a detailed analysis. Recently, with the power of diffusion models, researchers started using extra cues like texts to help the interpretation of the sketches and to create the desired 3D models [LFLG24]. However, in addition to being limited to generating models seen during the learning, the ability to control the modeling and editing of the resulting shapes is limited. In contrast, *PerceptualLift* generalizes across diverse, organic sketches without structured annotations or predefined assumptions and enhances interactivity by allowing real-time depth adjustments.

3. Perceptual Study: Role of hatches in depth perception

In this section, we present in detail and discuss the results of a preliminary perceptual study conducted to understand how humans perceive hatches and relate them to depth in a drawing.

3.1. Objectives and Hypotheses of the Study

The main goal of this study is to characterize the most salient properties of hatches and understand their role in conveying the changes in depth of a depicted shape. With the prior that the presence of hatches and their location help convey changes of depth, we focused on the following specific characteristics of the hatch strokes, each of them bringing a specific question.

Angle and orientation: What impact do the angle and orientation of the hatch strokes, relative to the outline of the depicted shape, have on the perceived direction of the change of depth?

Curvature: Does the curvature of the hatch strokes play a role in the perceived curvature of the depicted shape?

Thickness and length: Does a change in the thickness or the length of hatches play a role in the perceived magnitude of depth changes in the depicted shape?

Frequency: Can a variation in density or spacing in a hatch pattern impact the perceived shape?

Thinking about these questions, we formulated the following hypotheses:

- **H1:** The use of hatches curved towards the extremities suggests the surface is bending outwards, while hatches curved away from the extremities indicate a surface bending inward (see Figure 3).
- **H2:** The curvature of the hatches depicts the curvature of the shape's cross-section at that point, so straight hatch strokes denote a flat region in the shape (see Figure 4).
- **H3:** The length of the hatches corresponds to the amount of bending in the depth direction (see Figure 5).
- **H4:** Thicker hatches indicate as well a larger amount of bending in the depth direction (see Figure 6).
- **H5:** The frequency of the hatches is a third way to indicate the amount of bending in the depth direction (see Figure 7).

3.2. Experiment

To validate or invalidate these hypotheses, participants were shown a series of visual stimuli - either printed or on a screen - designed to evaluate how effectively they could interpret depth changes based on specific hatching attributes. Each stimulus showed several shapes drawn using contours and hatches, similar to the ones in Figures 3 to 7, and specified a task to perform on top. Note that the same external contour, with a constant thickness, was always used for the shapes in the same sketch to specifically study the role of hatches in their perception.

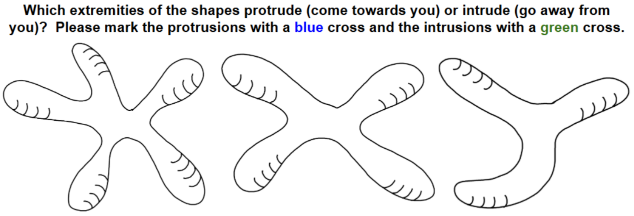

Figure 3: *Influence of the direction of curvature of the hatches. This input was used for a task related to **H1** (top: task description; bottom: sketch to annotate).*

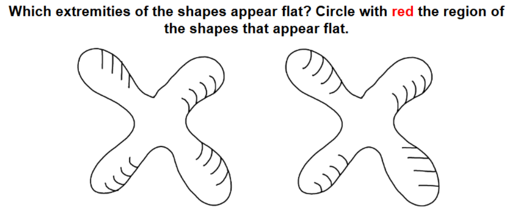

Figure 4: *Influence of the amount of curvature of the hatch strokes. This input was used in a task related to **H2** (top: task description; bottom: sketch to annotate).*

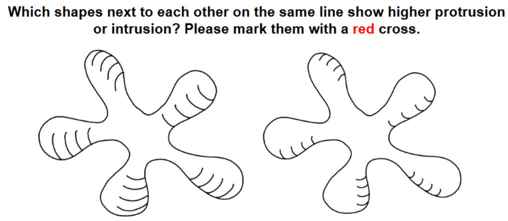

Figure 5: *Influence of the length of the hatch strokes. This input was used in a task related to **H3** (top: task description; bottom: sketch to annotate).*

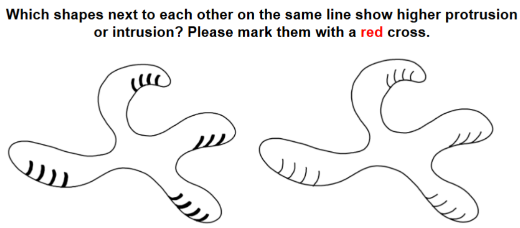

Figure 6: *Influence of the thickness of the hatch strokes. This input was used in a task related to **H4** (top: task description; bottom: sketches to annotate).*

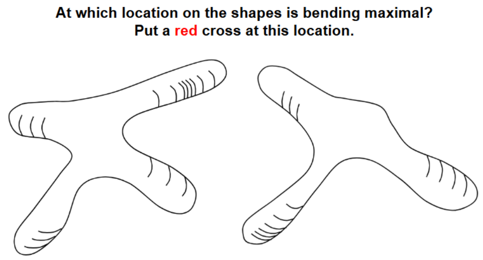

Figure 7: *Role of the frequency of the hatch strokes. This input was used in a task related to **H5** (top: task description; bottom: sketches to annotate).*

3.3. Result of the Perceptual Study

Our study involved 77 participants aged between 15 and 75 years old (41 males and 36 females), containing 42 participants familiar with some form of visual art and 35 unfamiliar with any form of visual art.

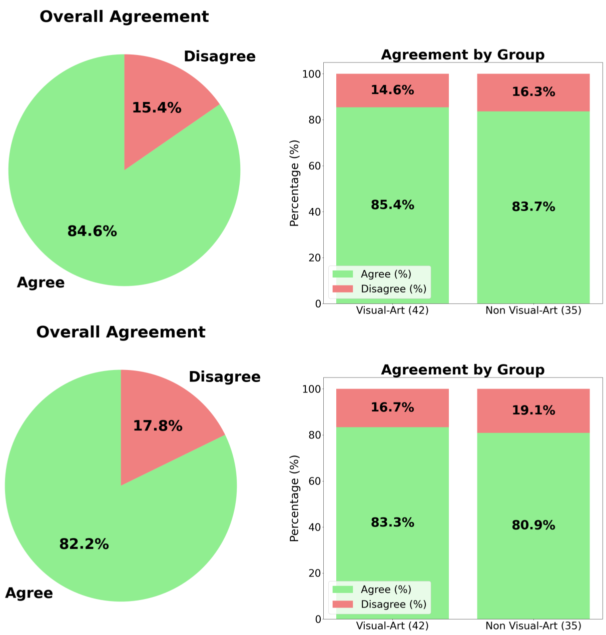

Figure 8: ***Agreement with hypotheses (H1 (top), H5 (bottom)) across all participants and by group:*** *The left chart shows the overall agreement rate among all participants, while the right chart compares agreement percentages between art-related and non-art-related participants. The results indicate a high overall agreement, with minimal differences between the two groups.*

Once a participant was done with the tasks, we evaluated how much his/her answers validated or contradicted the hypotheses listed in Section 3.1. Figure 8 summarizes some of our findings. To better assess the agreement with the hypotheses and compare the two groups of participants, we conducted a **Chi-Square** test for independence. For instance, for the first hypothesis H1, results showed that **84.6%** of participants agreed with the hypothesis, with **85.4%** agreement in the art-related group and **83.7%** in the non-art-related group. The **p-value of 0.13** indicates no significant difference between the two groups. The full results and their detailed analysis are provided in a supplementary document.

In summary, all results indicated that the difference between art-related and non-art-related participants was not statistically significant. This suggests that the correlation between depth and hatches is inherent to human perception independent of their expertise in art. In addition, the experiment not only enabled us to validate all our hypotheses but also to choose between three possible ways to depict the amount of bending in the depth direction: While H3 and H4 related questions demonstrated the role of stroke length and thickness with a reasonable amount of agreement (about 70%), our experiments showed that increasing the frequency of hatches is the best way to convey high local curvature in depth, with an agreement of more than 85% to H5. In particular, we noted that while they may seem related to the amount of bending, hatches of different lengths were also used by trained artists to accurately convey shadowing over a shape, depending on the position of a virtual light in the drawing. In contrast, the frequency of hatches is able to convey the amount of curvature independently from the size of the shadowed region. Therefore, in the remainder of this paper, we build on only three characteristics of hatches stroke to infer a 3D shape from a sketch, namely the fact they are curved, the direction to their center of curvature to indicate the protruding parts, and their pics of frequency to indicate fast changes of depth.

4. Developing a Computational Model

Based on the hypotheses validated in Section 3, we now develop algorithmic rules relying on hatches to depth and use them to develop a computational model for 3D modeling from a sketch. We developed our tool on top of a sketch-based modeling inspired by Matisse [BPCB08] and SCALIS [ZBQC13], based on skeleton-based implicit surface representations. In our modified version of the tool, called *PerceptualLift*, we incorporate hatches to modify the initially inflated implicit surface created by Matisse/SCALIS from a single sketch to overcome the limitation of flat silhouettes. In contrast, the use of hatches allows the users to fully control the 3D geometry.

Our method starts with the user sketching a 2D contour representing the object's external silhouette, enhanced with a set of free-hand-drawn hatch strokes. The contour is first inflated into a 3D model as was done in previous skeleton-based implicit modeling techniques: The medial axis of the contour is computed, simplified to avoid accounting for any local jiggling in the drawing and saved as a graph of strokes (from here on, we refer to it as the skeleton graph), and then used as the skeleton of a SCALIS surface, built as to interpolate the contour (see [ZBQC13]). Instead of maintaining it in the drawing plane facing the camera as in previous work, we rely on the information extracted from hatches for adapting the local depth of the skeleton graph. In practice, we consider the first hatch sketched within a local region as a reference and use its properties in terms of the direction of curvature to identify the direction of protrusion. The other hatches are used to locate regions that bend in the depth direction, and their frequency helps define the amount of bending. This way, our method robustly gives a solution even if the user did not draw consistent curvature directions for the hatches. Note that as explained in Section 3, we do not consider the length of hatches since their frequency was proved a more robust alternative to assess the magnitude of local bending in the depth direction. Similarly, although trained artists may sketch hatches whose radius of curvature conveys the radius of curvature of the shape's local cross-section, we do not enforce this in this work and allow any curved hatch stroke, to make our sketch-based modeling system accessible to beginners. Next, we explain how we use the hatches to update the 3D surface, which is done in three main steps.

4.1. Region Influenced by Bending in the Depth direction

When the user draws hatches, the first step is to understand their region of influence - in other words, the part of the 3D model that should be affected by them. Identifying the region of influence is easy when the user hatches on a segment of the skeleton graph corresponding to a branch leading to a leaf node (degree 1 vertex). However, we need to generalize the concept of "region of influence" to avoid any restriction on where the user can draw hatches. To do this, we first define a weight per edge (v_i, v_j) for our skeletal graph, defined as follows:

$$W(v_i, v_j) = \frac{\pi}{3} ||(v_i, v_j)|| \left(r_i^2 + r_i r_j + r_j^2 \right).$$

where r_i and r_j are the radius of our implicit 3D model at the vertices v_i and v_j. Intuitively, this weight captures the volume contribution of the segment thickness at the edge (v_i, v_j) approximated with a truncated cone. When the user hatches a region of the depicted shape, we discard the skeletal edges in the hatched region, which virtually divides the skeleton graph (assumed of zero topological genus) into two components. We compare the respective sum of edge weights over these two components and select the one with the smallest total weight as the one whose depth should be modified due to bending in the depth direction, in addition to depth changes in the hatched region itself.

4.2. Direction of Bending

Once the region whose depth is influenced by the hatches is identified, we use the first finding of Section 3.1 (validation of H1) to select the right bending direction - i.e. decide whether the shape should bend towards (protrude) or away from the viewer (extrude). To achieve this, we orient the skeletal graph edges corresponding to hatches from the subset with the larger weight towards the subset with the smaller weight and designate the first oriented edge as a vector v. Then, we analyze the first hatch near v to define a vector $d = \frac{m_c - m_s}{||m_c - m_s||}$ (representing the curvature deviation along the hatch) where m_s is the midpoint of the line segment connecting the start and end of the hatch under consideration, and m_c the midpoint of the hatch polyline. Note that, as already stressed, computing the exact center of curvature is not needed since, as already stressed, the curvature value is not required. Moreover, we considered only the first hatch stroke as we assume that all the hatches drawn together are consistent in their intent to represent the bending direction. Based on the vectors v and d, we define the bending direction as:

$$\text{Bending Direction} = \text{sign}(\mathbf{v} \cdot \mathbf{d})$$

As illustrated in Figure 9, a positive value indicates that the shape deforms outward away from the viewer, while a negative means the shape deforms inward towards the viewer.

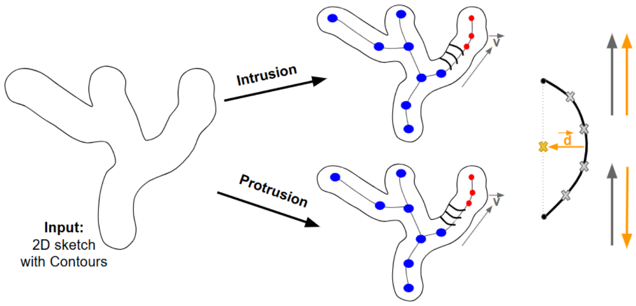

Figure 9: ***Hatch strokes analysis:*** *The direction d towards the center of curvature of hatch strokes is computed and compared to the direction v towards the shape part affected by the bend to determine the orientation of the bend in the depth direction. This will make the influenced shape part either protrude towards the viewer or extrude to the back of the scene.*

4.3. Amount of Bending

After finding the bending region and direction, the bending amount is computed from the density of the hatch strokes, as discussed and validated in Section 3. We then move in the z-direction the vertices of the influenced region of the skeleton graph based on this bending amount while preserving their x and y coordinates. This is done as follows:

Bending strategy and admissible range of curvature: Curvature in the depth direction should be localized in the hatched region of the shape. In this work, we choose to bend this part of the shape according to the arc of a circle (an appropriate choice of organic shapes), while the remainder of the influenced part of the shape then continues straight in the direction of the last skeleton edge. For instance, the shapes in Figure 3 will show uniformly curved branches in the depth direction, while the left extremity of the shape in Figure 5 will continue straight after the last hatch.

The amount of bending in the depth direction should be computed to prevent self-overlapping of the 3D shape. Therefore, the local thickness τ of the shape gives us the minimal radius of curvature (corresponding to the maximal curvature $\frac{1}{\tau}$) that can be used for bending it. See Figure 10, where the shape is seen from the top. In practice, we benefit from our implicit representation, where skeleton-graph edges carry thickness values, to evaluate the range of admissible curvature $c \in \left[0, \frac{1}{\tau}\right]$.

Another remark enables us to reduce even more the range of admissible curvatures, given our specific application of 3D modeling from a sketch and the common hypothesis that the sketch provides a general view of the shape: The protruding (respectively, extruding) parts (those influenced by a hatched region) should make an angle of strictly lower than 90 degrees with the stable parts that remain in the drawing plane, located on the other side of the hatches:

Proceedings published by Eurographics - The European Association for Computer Graphics.

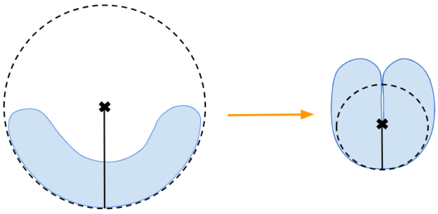

Figure 10: ***Max bending estimation:*** *When the blue shape bends, its radius of curvature (defined as the radius of the osculating circle) decreases but cannot become smaller than the shape's local thickness, else the shape would self-intersect. Moreover, if this is a top view and the shape is sketched from the front, the length of the curved region cannot exceed a quarter of the circumference of the osculating circle; otherwise, the shape would start folding back onto itself from the drawing viewpoint.*

Indeed, if they were fully coming towards us (respectively fully bending backwards), or at the extreme if they were folding back even more above (respectively below) the stable parts, sketching them from the current viewpoint would have been impossible. To enforce this constraint, we reduce the maximal curvature if needed so that the arc of the circle taken by the curved part of the shape (the one covered by the hatches) always remains smaller than 1/4 of the circumference of the osculating circle (see Figure 10).

Lastly, given the infinity of possible admissible depth that still remains, we may choose to complement these constraints using another common hypothesis in sketch-based modeling systems: self-similarity from different viewpoints. In this case, the curvatures used in the depth direction should belong to the same range of curvature as those observed in the drawing plane (x,y). We observed that choice often results in a more plausible first guess.

Given these constraints, which result in a given admissible range of curvatures $[c_{min}, c_{max}]$ at the center of the hatched region, we use the local frequency of the hatch strokes to evaluate a plausible curvature value within the range.

Computing the frequency of hatch strokes: Suppose we have a cluster of n contiguous hatches. To determine its spatial extent, we construct a polyline by connecting the starting points of all hatches, thereby approximating the arc length L of the cluster. Under the assumption of a uniform distribution of hatches along this extent, we define the spatial frequency as:

$$f = \frac{n}{L} \quad (1)$$

To be able to use it to modulate the amount of bending our 3D shape, we normalize it in $[0;1]$ according to the following strategy: Given the width w of the brush used to draw the hatches, a fully hatched area would correspond to N strokes neighboring where $N = L/w$. To assign the frequency 1 to such a fully hatched region, we define:

$$\hat{f} = \frac{nw}{L} \quad (2)$$

$\hat{f}$ can then be directly used to set the desired curvature in the depth direction to:

$$c = c_{min} + \hat{f} \times (c_{max} - c_{min}) \quad (3)$$

Depth values computation: In practice, we compute the depth at each skeleton vertex in the hatched region by projecting it to an arc of a circle viewed from the top, while the depth for the remaining point of the influence region is computed through projection onto a straight line segment.

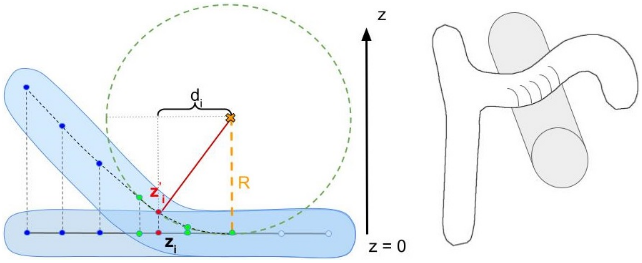

Figure 11: ***Depth Adjustment via Projection (top view):*** *Once the radius and center of curvature are computed from the density of hatches, skeleton vertices in the hatched region is projected to an arc or circle (see left), or a cylinder in 3D (see right), while the remaining points of the influenced part of the shape continue along a straight path, as to ensure smooth changes of depth.*

To be precise, we project the skeleton vertices of the form $(x_i, y_i, 0)$ that lie within the hatched zone onto a circular arc of radius R. As illustrated in Figure 11, this projection is done by updating the z-coordinate of each such vertices as:

$$z_i' = R - \sqrt{R^2 - d_i^2}$$

where R is the radius of curvature, determined based on the frequency of hatches and the shape's local thickness, and d_i is the distance of point i from the start of the curved segment along the original skeleton. To maintain consistency with the input viewpoint, we employ an orthographic camera for visualization.

5. Implementation, Results and Discussion

5.1. User Interface

Our interactive system is implemented in WebGL using JavaScript. *PerceptualLift* interface has two windows - one for 2D sketching, where the user can draw the shape contour and the hatches, and the other where the resulting 3D model is visualized. The users are given options to quickly draw, erase, and modify hatches, as well as options to adjust the hatch density, spatial extent, and distribution. In addition, *PerceptualLift* allows users to interactively fine-tune these parameters using the mouse wheel (see Figure 12).

5.2. Results & Performances

To accommodate beginners who may not be familiar with conventional lighting assumptions—such as light typically coming from the top left (see Figure 2)— our current system processes sketched hatches consistently, as shown in Figure 13. This ensures that the

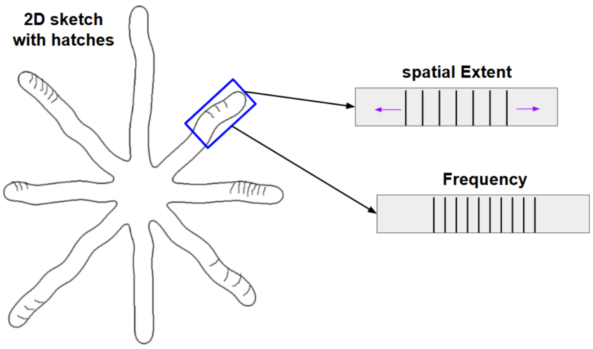

Figure 12: ***User Interface:*** *Left: The user is provided with different brushes for sketching the contour of the shape and the hatches. Right: After sketching, the user can interactively adjust the spatial extent and the frequency of the hatches using the mouse wheel.*

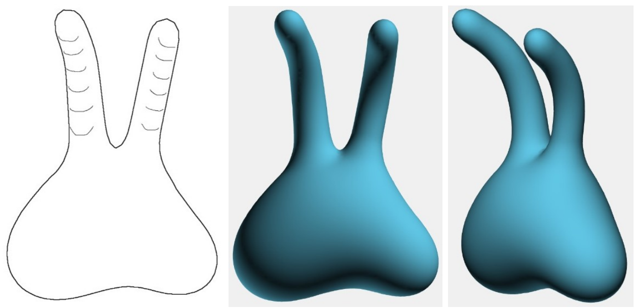

Figure 13: ***Robust Interpretation of Hatching:*** *The system ensures a uniform interpretation of hatching even if the hatched side (which should ideally outline the shaded side of a shape) was not consistent in the input sketch.*

direction from which a user sketches along the shape's border does not affect the interpretation, as confirmed by the user study, which found no perceptual differences in interpretation.

Figures 15 and 16 show a series of results. Our two users were first asked to reproduce some of the pictures from the perceptual user study (three first lines) and then to design a shape of their choice (two last lines). More examples are shown in Figure 1 and in the companion video.

Each of these examples was sketched interactively in just a few minutes using our system—about one minute for the simpler ones and two to three minutes for the more complex ones—with the corresponding 3D shape generated in real time. The bending transformations from user-drawn hatches were computed almost instantaneously.

5.3. Limitations

PerceptualLift validated the fact that an effective sketch-based modeling tool can be designed by interpreting user-defined hatches. However, our current system suffers several limitations.

The first one is that our current implementation does not exploit

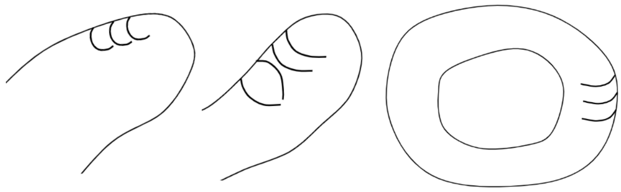

Figure 14: ***Three Limitations Cases:*** *High local curvature of hatch strokes seems to depict a bump in the shape's cross-section (left), which is not taken into account in our method; A meaningless sketch (middle); In shapes of non-zero topological genus (one or several holes), removing the hatched region may not always separate the skeleton into two components, which prevents the user of our algorithm to define the protruding v.s. extruding direction (right).*

all the findings of our perceptual study. In particular, our current implementation does not take the curvature of hatches into consideration and assumes a circular cross-section, making it difficult to always make the desired interpretation. For instance, our system cannot interpret the curved and short hatches at the border of the shape as shown in Figure 14, left, as a bump.

Second, we assume that the hatches are consistent in terms of the direction of curvature and are uniformly distributed. While the former allows us to ignore meaningless hatches (see Figure 14, center), the latter stops us from having possibly more accurate bending variations in the depth direction. Despite these limitations, our current choices provide an easy-to-use interface, accessible to beginners, for generating shapes with non-flat silhouettes.

Additional constraints on our current system include the limitation of the out-of-plane 3D transformations to simple bending, restricting the range of shapes it can represent. It also does not support more complex deformations, such as twisting or tapering along the depth axis. Moreover, while the algorithm does not exactly preserve the original 2D sketch outline, it produces a close and visually coherent approximation when inflating it into 3D. While this does not affect the general shape perception, it may introduce inconsistencies when comparing the final 3D output to the original 2D drawing.

Lastly, the current implementation is also restricted to genus-zero shapes, as extending the current system to accommodate hatches on a non-zero topological genus (as in Figure 14 (right)) would require new perceptual studies to validate the expected behaviour.

6. Conclusion and Future Work

Being able to interpret hatches is a good way to make sketch-based modeling easier and faster since shapes that bend in the depth direction can then be designed from a single viewpoint. This work introduced *PerceptualLift*, a system inspired by artistic and scientific illustrations to generate non-flat 3D models from contour sketches with hatches.

We used a preliminary perceptual study to discover the way per-

Proceedings published by Eurographics - The European Association for Computer Graphics.

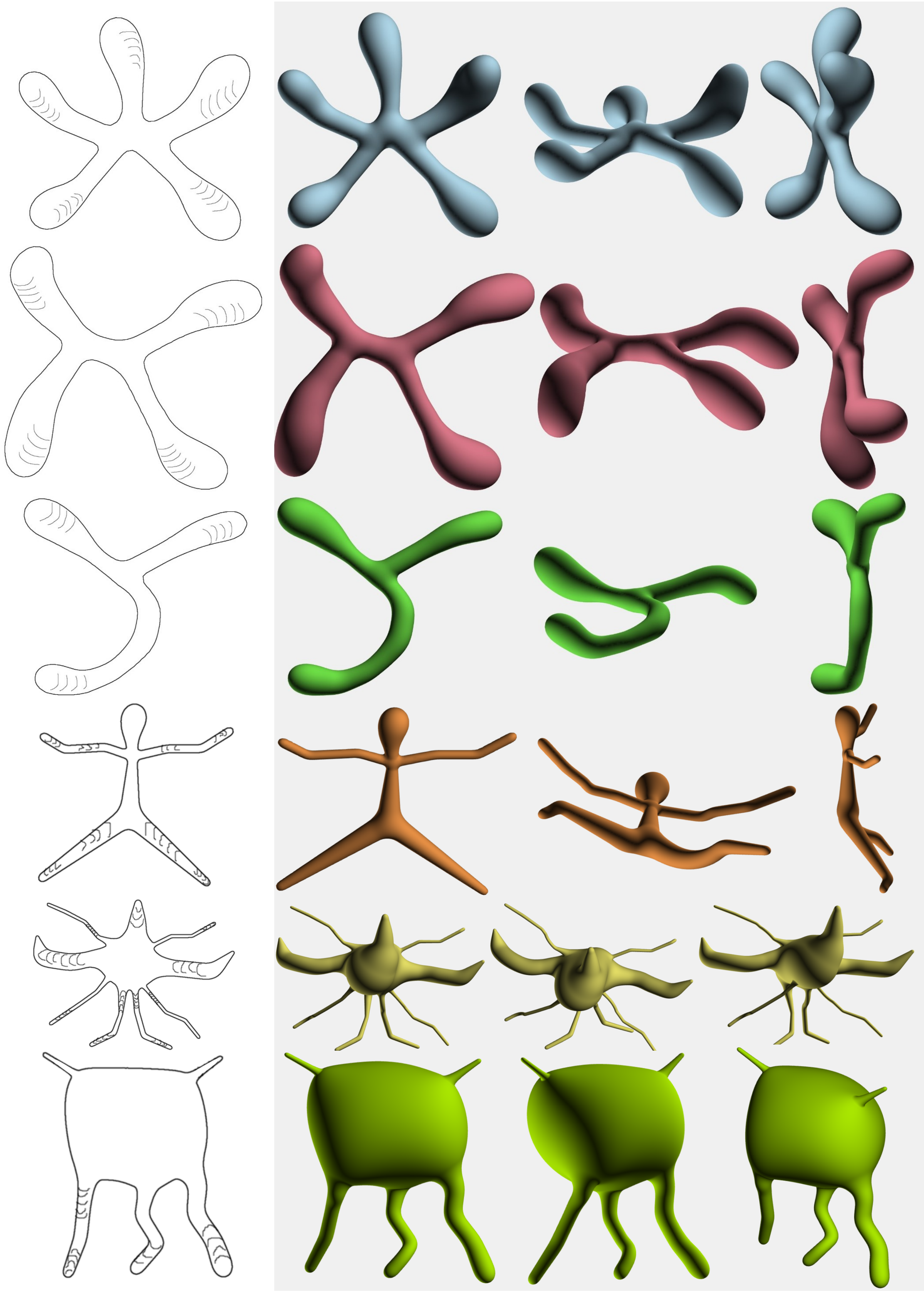

Figure 15: *A few results created using PerceptualLift. Left to right: Input 2D sketch with user-drawn hatches, front view of the output 3D shape, and two different side views.*

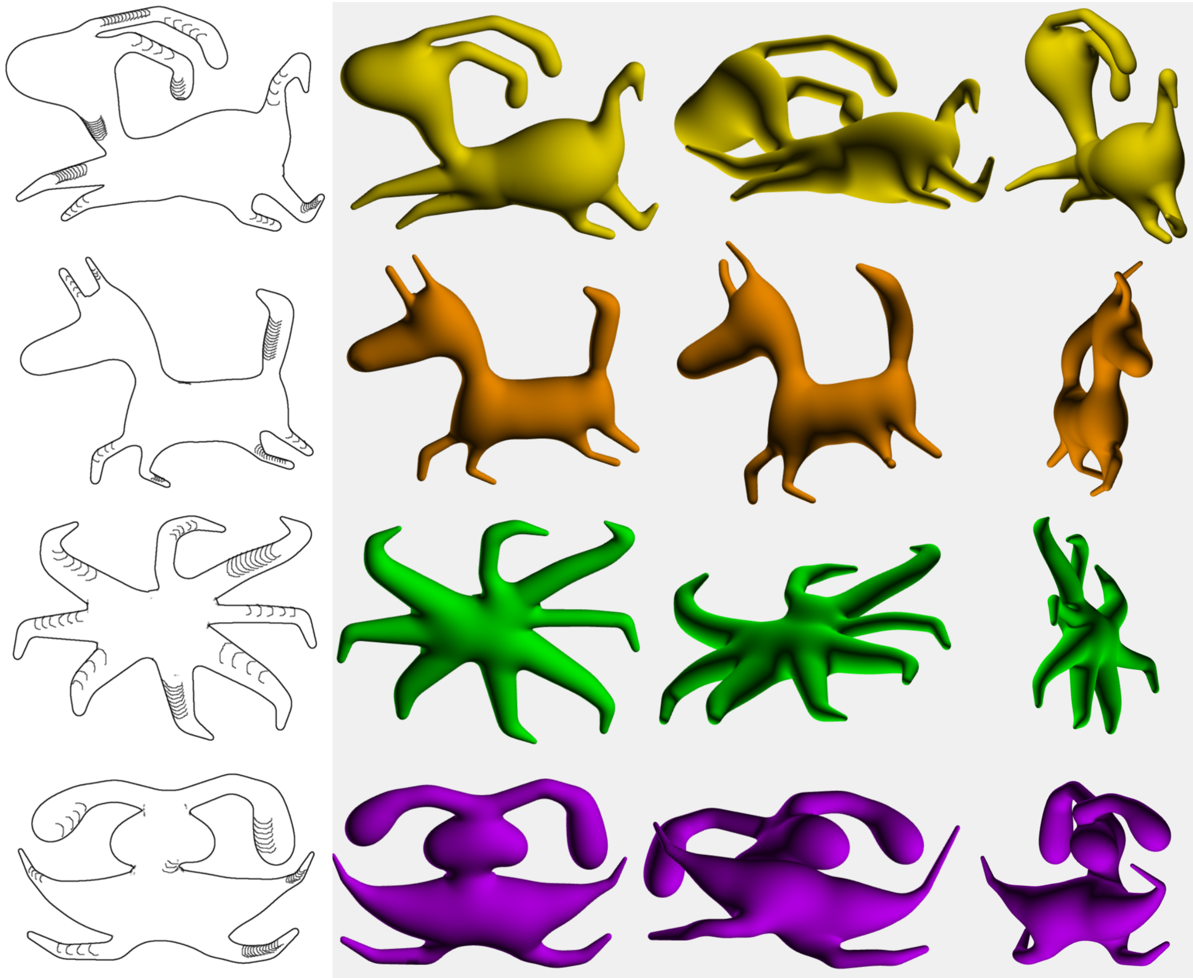

Figure 16: *A few results created using PerceptualLift. Left to right: Input 2D sketch with user-drawn hatches, front view of the output 3D shape, and two different side views.*

ceptual cues can be extracted from artistic hatching. Once formalized, these cues were then used as a computational tool in an intuitive and interactive 3D modeling interface. Our current implementation was designed for both artists and non-artists and, therefore, made robust to coarse, non-regular hatching gestures. More generally, our use of an old-style, rule-based system offers a lightweight and scalable alternative to deep-learning methods which would have required extensive training on a large dataset - the latter being probably difficult to gather for this specific application.

Our solution should not be seen as a stand-alone tool. Indeed, while it enables us to design a curved 3D shape from a single viewpoint, it cannot be used to design shapes with hidden parts, such as those that fold back onto themselves. Therefore, in the future, we plan to incorporate it into a general-purpose sketch-based modeling system such as [IMT06, BPCB08], where users interactively turn around their model to add new parts from the most appropriate viewpoint: thanks to this research, each of the new parts could now be curved out of the drawing plane.

Acknowledgements

The authors thank all anonymous reviewers for their constructive feedback. This work was funded by Prof. Cani's fellowship from Hi!Paris.

References

[BC20] Bhattacharjee S., Chaudhuri P.: A survey on sketch based content creation: from the desktop to virtual and augmented reality. In *Computer Graphics Forum* (2020), vol. 39, Wiley Online Library, pp. 757–780. 2

[BJD*12] Borosán P., Jin M., DeCarlo D., Gingold Y., Nealen A.: Rigmesh: automatic rigging for part-based shape modeling and deformation. *ACM Transactions on Graphics (TOG) 31*, 6 (2012), 1–9. 2

[BPCB08] Bernhardt A., Pihuit A., Cani M.-P., Barthe L.: Matisse: Painting 2d regions for modeling free-form shapes. In *SBM'08-Eurographics Workshop on Sketch-Based Interfaces and Modeling* (2008), Eurographics Association, pp. 57–64. 2, 4, 9

[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*. 2008, pp. 1–11. 2

[CZS*13] CHEN T., ZHU Z., SHAMIR A., HU S.-M., COHEN-OR D.: 3-sweep: Extracting editable objects from a single photo. *ACM Transactions on graphics (TOG) 32*, 6 (2013), 1–10. 2

[DAI*18] DELANOY J., AUBRY M., ISOLA P., EFROS A. A., BOUSSEAU A.: 3d sketching using multi-view deep volumetric prediction. *Proceedings of the ACM on Computer Graphics and Interactive Techniques 1*, 1 (2018), 1–22. 2

[DSC*20] DVOROŽŇÁK M., SÝKORA D., CURTIS C., CURLESS B., SORKINE-HORNUNG O., SALESIN D.: Monster mash: a single-view approach to casual 3d modeling and animation. *ACM Transactions on Graphics (ToG) 39*, 6 (2020), 1–12. 2

[ES04] ESTEBAN C. H., SCHMITT F.: Silhouette and stereo fusion for 3d object modeling. *Computer Vision and Image Understanding 96*, 3 (2004), 367–392. 2

[GI13] GERL M., ISENBERG T.: Interactive example-based hatching. *Computers & Graphics 37*, 1-2 (2013), 65–80. 2

[GIZ09] GINGOLD Y., IGARASHI T., ZORIN D.: Structured annotations for 2d-to-3d modeling. In *ACM SIGGRAPH Asia 2009 papers*. 2009, pp. 1–9. 2

[HWX*21] HU T., WANG L., XU X., LIU S., JIA J.: Self-supervised 3d mesh reconstruction from single images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2021), pp. 6002–6011. 2

[HZ00] HERTZMANN A., ZORIN D.: Illustrating smooth surfaces. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques* (2000), pp. 517–526. 2

[IMT06] IGARASHI T., MATSUOKA S., TANAKA H.: Teddy: a sketching interface for 3d freeform design. In *ACM SIGGRAPH 2006 Courses*. 2006, pp. 11–es. 2, 9

[Joh02] JOHNSTON S. F.: Lumo: Illumination for cel animation. In *Proceedings of the 2nd international symposium on Non-photorealistic animation and rendering* (2002), pp. 45–ff. 2

[KH06] KARPENKO O. A., HUGHES J. F.: Smoothsketch: 3d free-form shapes from complex sketches. In *ACM SIGGRAPH 2006 Papers*. 2006, pp. 589–598. 2

[KNBH12] KALOGERAKIS E., NOWROUZEZAHRAI D., BRESLAV S., HERTZMANN A.: Learning hatching for pen-and-ink illustration of surfaces. *ACM Transactions on Graphics (TOG) 31*, 1 (2012), 1–17. 2

[KYC*17] KRS V., YUMER E., CARR N., BENES B., MĚCH R.: Skippy: Single view 3d curve interactive modeling. *ACM Transactions on Graphics (TOG) 36*, 4 (2017), 1–12. 2

[KYYL08] KIM Y., YU J., YU X., LEE S.: Line-art illustration of dynamic and specular surfaces. *ACM Transactions on Graphics (TOG) 27*, 5 (2008), 1–10. 2

[LFLG24] LIU F.-L., FU H., LAI Y.-K., GAO L.: Sketchdream: Sketch-based text-to-3d generation and editing. *ACM Transactions on Graphics (TOG) 43*, 4 (2024), 1–13. 3

[LMHB00] LAKE A., MARSHALL C., HARRIS M., BLACKSTEIN M.: Stylized rendering techniques for scalable real-time 3d animation. In *Proceedings of the 1st international symposium on Non-photorealistic animation and rendering* (2000), pp. 13–20. 2

[LPL*18] LI C., PAN H., LIU Y., TONG X., SHEFFER A., WANG W.: Robust flow-guided neural prediction for sketch-based freeform surface modeling. *ACM Transactions on Graphics (TOG) 37*, 6 (2018), 1–12. 2

[NISA07] NEALEN A., IGARASHI T., SORKINE O., ALEXA M.: Fibermesh: designing freeform surfaces with 3d curves. In *ACM SIGGRAPH 2007 papers*. 2007, pp. 41–es. 2

[OSSJ09] OLSEN L., SAMAVATI F. F., SOUSA M. C., JORGE J. A.: Sketch-based modeling: A survey. *Computers & Graphics 33*, 1 (2009), 85–103. 2

[PHWF01] PRAUN E., HOPPE H., WEBB M., FINKELSTEIN A.: Real-time hatching. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (2001), p. 581. 1, 2

[PK19] PHILBRICK G., KAPLAN C. S.: Defining hatching in art. In *Proceedings of the 8th ACM/Eurographics Expressive Symposium on Computational Aesthetics and Sketch Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering* (2019), pp. 111–121. 2

[RDI10] RIVERS A., DURAND F., IGARASHI T.: 3d modeling with silhouettes. In *Acm siggraph 2010 papers*. 2010, pp. 1–8. 2

[RK00] ROSSL C., KOBBELT L.: Line-art rendering of 3d-models. In *Proceedings the Eighth Pacific Conference on Computer Graphics and Applications* (2000), IEEE, pp. 87–96. 2

[SBSS12] SHAO C., BOUSSEAU A., SHEFFER A., SINGH K.: Crossshade: shading concept sketches using cross-section curves. *ACM Transactions on Graphics (TOG) 31*, 4 (2012), 1–11. 2

[SS10] SINGH M., SCHAEFER S.: Suggestive hatching. In *CAe* (2010), pp. 25–32. 1

[WWL*22] WAGNER R., WEGEN O., LIMBERGER D., DÖLLNER J., TRAPP M.: A non-photorealistic rendering technique for art-directed hatching of 3d point clouds. In *VISIGRAPP (1: GRAPP)* (2022), pp. 220–227. 2

[XCS*14] XU B., CHANG W., SHEFFER A., BOUSSEAU A., MCCRAE J., SINGH K.: True2form: 3d curve networks from 2d sketches via selective regularization. *ACM Transactions on Graphics 33*, 4 (2014). 2

[XHY*22] XU P., HOSPEDALES T. M., YIN Q., SONG Y.-Z., XIANG T., WANG L.: Deep learning for free-hand sketch: A survey. *IEEE transactions on pattern analysis and machine intelligence 45*, 1 (2022), 285–312. 2

[ZBQC13] ZANNI C., BERNHARDT A., QUIBLIER M., CANI M.-P.: Scale-invariant integral surfaces. In *Computer Graphics Forum* (2013), vol. 32, Wiley Online Library, pp. 219–232. 4

[ZGG21] ZHANG S.-H., GUO Y.-C., GU Q.-W.: Sketch2model: View-aware 3d modeling from single free-hand sketches. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2021), pp. 6012–6021. 2

[ZISS04] ZANDER J., ISENBERG T., SCHLECHTWEG S., STROTHOTTE T.: High quality hatching. In *Computer Graphics Forum* (2004), vol. 23, Wiley Online Library, pp. 421–430. 2

[ZYC*22] ZHANG C., YANG L., CHEN N., VINING N., SHEFFER A., LAU F. C., WANG G., WANG W.: Creatureshop: Interactive 3d character modeling and texturing from a single color drawing. *IEEE Transactions on Visualization and Computer Graphics* (2022). 2