Multi-view Wire Art

KAI-WEN HSIAO, National Tsing Hua University
JIA-BIN HUANG, Virginia Tech
HUNG-KUO CHU, National Tsing Hua University

Wire art is the creation of three-dimensional sculptural art using wire strands. As the 2D projection of a 3D wire sculpture forms line drawing patterns, it is possible to craft multi-view wire sculpture art—a static sculpture with multiple (potentially very different) interpretations when perceived at different viewpoints. Artists can effectively leverage this characteristic and produce compelling artistic effects. However, the creation of such multi-view wire sculpture is extremely time-consuming even by highly skilled artists. In this paper, we present a computational framework for automatic creation of multi-view 3D wire sculpture. Our system takes two or three user-specified line drawings and the associated viewpoints as inputs. We start with producing a sparse set of voxels via greedy selection approach such that their projections on the virtual cameras cover all the contour pixels of the input line drawings. The sparse set of voxels, however, do not necessary form one single connected component. We introduce a constrained 3D pathfinding algorithm to link isolated groups of voxels into a connected component while maintaining the similarity between the projected voxels and the line drawings. Using the reconstructed visual hull, we extract a curve skeleton and produce a collection of smooth 3D curves by fitting cubic splines and optimizing the curve deformation to best approximate the provided line drawings. We demonstrate the effectiveness of our system for creating compelling multi-view wire sculptures in both simulation and 3D physical printouts.

CCS Concepts: • Computing methodologies → 3D imaging; Parametric curve and surface models;

Additional Key Words and Phrases: image-based modeling, wire art, skeleton extraction, shape deformation

ACM Reference Format:

1 INTRODUCTION

Wire sculpture is a unique art form that creates complex objects out of wire. The use of wire as a medium for sculpturing has been widely applied to furniture design [Postell 2012], crafting wire wrapped jewelry [Jarussi et al. 2015; WigJig 2015], and wire sculptural art [2007]. Very recently, French artist Matthieu Robert-Ortis has created compelling wire sculptures that exhibit two distinct image interpretations when viewed at different perspectives. Figure 2 presents two examples of multi-view wire sculpture art.¹

Similar to shadow art, multi-view wire art allows us to create objects with multiple interpretable line drawings at different viewpoints. Furthermore, wire art offers additional flexibility in that the

¹More examples can be found at http://cargocollective.com/matthieu-robert-ortis

En-video
wire sculpture does not require a light source for casting shadows to see the hidden images. It is thus of great interests for enabling designers and hobbyists to create multi-view wire art. However, designing and creating multi-view wire art is prohibitively difficult for novice users due to the required artistic skills and significant efforts in resolving conflicting contour constraints from two or more line drawings.

Our work. We present a system for enabling users to create the desired multi-view wire art sculpture. As shown in Figure 1, our system takes two or three line drawing images and the associated viewpoints provided by the user as inputs and produce the 3D wire sculpture that forms the user-specified line drawings at different viewpoints. This problem is challenging because the wire structure satisfying all constraints by input line drawings often does not exist (see Figure 3). To tackle the challenge, we develop tools for constructing the sculpture by balancing the smoothness of the curves and the similarity between the specified line drawing and the projection at multiple viewpoints.

Specifically, our approach consists of two main stages: (1) visual hull reconstruction, (2) 3D curve skeleton extraction and deformation. In the first stage, we use a greedy selection approach to generate a collection of voxels covering all the contour pixels of the input line drawings and introduce a constrained 3D pathfinding algorithm to link the resulting isolated components into one single connected component that represents the final visual hull. Second, based on the reconstructed visual hull, we extract a 3D curve skeleton by exploiting a novel quality measurement that captures projection errors and structural complexity of the curve skeleton. We then fit individual skeleton lines using cubic splines and perform image-guided curve deformation to improve the projection error with respect to the input line drawings. Although our algorithm runs automatically, we provide interactive tools that enable interactive editing for users to repair or simplify the generated wire sculpture using 2D strokes. We demonstrate the effectiveness of our system on a wide variety of different line drawings. Numerous results show that the proposed method can produce multi-view wire sculptures with small distortions in the projected views.

Our contributions. (1) A system for the automatic creation of multi-view wire sculpture art using only user-specified 2D line drawings and the associated viewpoints. Our system is capable of generating complex wire sculptures that are extremely difficult to design and implement manually (e.g., three views); (2) A suite of computational techniques tailored to resolving conflicting constraints from multiple views while maintaining similarity between the projections and the input line drawings; (3) A characterization of quality over voxel resolution and the interval of viewing directions.

2 RELATED WORK

3D reconstruction. Our problem builds upon the concepts and techniques in 3D reconstruction from multiple 2D image observations. Structure-from-motion (SfM) [Schonberger and Frahm 2016; Snavely et al. 2006] and multi-view stereo (MVS) [Collins 1996; Furukawa and Ponce 2010; Goesele et al. 2007; Huang et al. 2018; Kuhn et al. 2017; Schönberger et al. 2016] algorithms are capable of reconstructing detailed 3D models in unconstrained settings (e.g., photos from the Internet). We refer the readers to [Furukawa and Hernández 2015] for a comprehensive overview of MVS algorithms. These methods, however, rely on establishing point correspondence across images and therefore have difficulty in handling smooth regions or thin wire structures. In light of this, several recent approaches exploit higher order features such as lines or curves as primitives for 3D reconstruction [Fabbri and Kimia 2010; Hofer et al. 2017; Liu et al. 2017; Usmézbas et al. 2016]. Assuming clean foreground-background separation, visual hull and silhouette intersection based algorithms can be also applied to construct 3D models from image observations [Laurentini 1994; Lazebnik et al. 2007; Matusik et al. 2000; Szeliski 1993] or from user-specified sketches [Olsen et al. 2009; Rivers et al. 2010]. Similar to visual hull based algorithms, our method also uses multiple silhouettes (from user-specified line drawings) as inputs to construct a 3D model of a wire sculpture where its projections match with the input silhouettes as much as possible. The difference lies in that our input silhouettes may conflict with each other (see an example in Figure 3) and produce many isolated components in the visual hull, causing difficulty in manufacturing.
a physical wire sculpture. Our work focuses on resolving the inconsistency and constructing a collection of smooth, continuous curve skeletons while minimizing the distortion between the projected views and the input line drawings.

Multiple interpretations from an object. Our work on creating multi-view wire art is related to several approaches for producing multiple visual interpretations from a single object. The perception of a static object (e.g., image, relief, sculpture) can vary dramatically depending on viewing distances [Oliva et al. 2006], figure-ground organization [Kuo et al. 2016], illumination from a certain direction [Alexa and Matusik 2010; Bermano et al. 2012], viewing directions [Keiren et al. 2009; Sela and Elber 2007], or casting shadows onto external planar surfaces [Min et al. 2017; Mitra and Pauly 2009]. Won and Lee [2016] further extended the idea to create shadow theatre from dynamic objects (i.e., animated characters). Our work differs in that we use 3D wire as a medium, allowing us to exhibit a clear prescribed line drawing either from certain viewing directions or cast shadows on planar surfaces using a point light source.

Wire sculpture design and modeling. More recently, there is a line of works devoted to modeling and fabricating the wire sculptures of a specific form. Larussi et al. [2015] presented a computational framework to assist the creation of wire wrapped jewelry. Liu et al. [2017] extended the dimension from 2D to 3D modeling of wire sculptures using only a few input images of physical objects. Miguel et al. [2016] proposed a computer-aided tool to facilitate converting a 3D model into a stable, self-supporting wire sculpture that is fabricatable with a 2D wire bending machine. Similarly, Zehnder et al. [2016] developed a computational design tool, targeting for the fabrication of ornamental curve networks defined on the 3D surfaces. Our work also falls into the category of 3D wire design and modeling but targets fundamentally different context.

3 METHOD OVERVIEW

Our system takes input as a set of 2D line drawing images (or simply images), \( I = \{I_1, \ldots, I_n\} \), along with the associated camera parameters specified by an user (i.e., pose, and perspective or orthogonal projection), \( \mathcal{P} = \{P_1, \ldots, P_n\} \), where \( n \in \{2, 3\} \) views in our experiments. Our goal is to reconstruct a set of 3D curves \( \mathcal{C} = \{C_1, \ldots, C_k\} \) represented by cubic splines. These curves together form a smooth, continuous, and compact 3D wire sculpture, which while seemingly irregular in both its geometry and structure, can precisely interpret the input images when viewed or cast shadows from the specified viewpoints. Figure 4 presents an overview of our system.

Pre-processing. For each input image, \( I_k \), we first apply a simple thresholding and thinning algorithm to extract a set of one-pixel wide 2D curve lines. We represent these foreground pixels of \( I_k \) as a graph \( G_k = (V_k, E_k) \), where \( V_k \) are pixels along the curve lines and \( E_k \) encode their connectivities. If the graph \( G_k \) contains multiple connected components, we further add edges to connect them according to their spatial proximity and use a conventional minimum spanning tree algorithm to get a connected graph.

Given the preprocessed input images, our system starts by back-projecting the foreground pixels to the 3D domain using the associated camera parameters to form a set of generalized cones [Laurentini 1994]. We then discretize the space into volumetric grids (or voxels) and find a minimum set of voxels such that the foreground pixels in each view can be completely covered by the projections of voxels. Applying a flooding algorithm produces a discrete visual hull, which often contains an excessive amount of isolated components in our context (Figure 4(b), Section 4.1). To integrate these isolated
components into one connected component, we first connect two proximate components via a 3D pathfinding method that jointly analyzes the spatial relations between two components in 3D and their 2D counterparts. Then we solve a binary labeling problem that retains necessary links to form a connected visual hull while minimizing the projection error due to the additional inconsistent voxels from the links (Figure 4(c), Section 4.2).

Here, the reconstructed visual hull represents a well-defined space in 3D where the curves of final wire sculpture would lie in. We first adopt a state-of-the-art volumetric thinning algorithm [Liu et al. 2010] to effectively compute a shape- and topology-preserving curve skeleton from the reconstructed visual hull. However, the extracted curve skeleton is typically complex in geometry, contains numerous redundant parts, and thus requires further refinement to be physically realizable (e.g., 3D printing). To this end, we devise a novel quality measurement tailored for wire modeling (Section 5.1) by iteratively removing the constituent skeleton lines to strike a balance between projection error and structure compactness (Figure 4(d), Section 5.2). Finally, we obtain a smooth, continuous, and compact 3D wire sculpture by fitting individual skeleton lines with parametric cubic splines, followed by an image-guided curve deformation to improve the projection accuracy with respect to the input images (Figure 4(e), Section 5.3).

4 VISUAL HULL RECONSTRUCTION

Reconstructing a 3D wire sculpture from multiple reference 2D images is related to shadow art [Mitra and Pauly 2009] as well as shape-from-silhouette applications [Laurentini 1994]. In this section, we describe a common step shared by existing works to reconstruct a 3D visual hull—a set of voxels whose projections best approximate the reference images using the associated camera parameters. With the reconstructed visual hull, we can then extract a compact wire sculpture with smooth, continuous 3D curves. Specifically, we aim to reconstruct the visual hull meeting the following two requirements: (1) Completeness: all the foreground pixels in the reference line drawings should be covered by the projections of the visual hull (Section 4.1); (2) Connectivity: the constituent voxels of the visual hull must form a connected component (Section 4.2).

4.1 Discrete Visual Hull Generation

Given a set of reference images, denoted as \(\{G_k, P_k\}_{k=1}^n\), our system starts by back-projecting the 2D image, \(G_k = \{V_k, E_k\}\), to 3D using the camera projection parameters, \(P_k\), and generating a set of generalized cones. To find a 3D volume that is compatible with \(G_k\) and \(P_k\), we compute a bounding cube based on the intersection of camera’s viewing frustums. We then discretize the cubic volume into \(N \times N \times N\) uniform voxels \(x_i \in \mathbb{R}^3\) with \(N\) indicates the resolution of voxelization. We define the relations between voxels and the reference images as follows. Let \(\hat{x}_i^k \in \mathbb{R}^3\) be the 2D projection of \(i\)th voxel \(x_i\) with respect to projection \(P_k\) for the \(k\)th view. We label a image pixel \(p^k \in V_k\) as complete if it falls within one of the circles (with a radius of 2 pixels) centered at \(\hat{x}_i^k\). We call a voxel consistent if it can establish the mapping \((\hat{x}_i^k, V_k)\) for all \(k\) views.

The reconstruction process starts by initializing the visual hull and a set of consistent voxels. Not surprisingly, the projection of these voxels leads to highly incomplete images due to the inconsistency between the reference images (Figure 5(left)). Mitra and Pauly [2009] tackle the inconsistency problem by a global optimization that deforms the reference images. While such a deformation framework performs well for solid shapes, it becomes ill-posed for line drawing images that contain many hollow regions formed by thin and complex line structures. We propose a simple yet effective approach to further expand visual hull with voxels that are aware of both i) projection error in 2D and ii) spatial compactness in 3D. For each of the incomplete pixel \(p^k \in V_k\), we can find a ray of voxels that map to \(p^k\). To favor voxels that have small projection errors and are spatially close to the target visual hull, we define the cost of the \(i\)th voxel \(x_i\) as follows:

\[
\sum_{k=1}^{n} D_k(\hat{x}_i^k) + (1 - \sum_{x_j \in N_{x_i}} D_{\text{prox}}(x_i, x_j)),
\]

where \(D_k\) is a distance transform map that measures the projection error with respect to image \(G_k\). The term \(N_{x_i}\) contains the voxels in the visual hull as well as \(12 \times 12 \times 12\) neighboring voxels to \(x_i\). The function \(D_{\text{prox}}(x_i, x_j)\) computes the proximity between voxels using a normalized Gaussian kernel with \(\mu = x_i\) and \(\sigma = 2\). Then, we employ a greedy algorithm that expands the visual hull by selecting the least-cost voxel for each incomplete pixel \(p^k\) and repeats the process until all the pixels in \(V_k\) are labeled as complete. As shown in Figure 5(right), this process results in a discrete visual hull that fulfills the completeness requirement. We include the pseudocode that details the above procedure in the supplementary document.

4.2 Connectivity Optimization

The discrete visual hull typically contains an excessive amount of isolated components. These isolated components significantly increase the difficulty in manufacturing a physical wire sculpture. Therefore, our goal in this step is to find an optimal set of 3D paths such that the isolated components are linked to form a single connected component while minimizing the projection error caused by inconsistent voxels that compose the 3D paths. We formulate the problem as a conventional minimum spanning tree problem on a graph representing the isolated components and their spatial relationships.

---

ACM Transactions on Graphics, Vol. 37, No. 6, Article 242. Publication date: November 2018.
Graph construction. First, we apply a flooding algorithm to constituent voxels of discrete visual hull and generate a collection of isolated components, denoted as \( X = \{X_1, \ldots, X_m\} \). Next, we construct a component graph \( G = (X, E) \) by adding a graph edge \( e_{ij} \) to \( E \) if the shortest distance between two components \( X_i \) and \( X_j \) is below a pre-defined threshold (15% of the edge length of the bounding cube in our experiments). For each graph edge \( e_{ij} \), we employ the \( A^* \) pathfinding algorithm with best-first-search strategy to trace a 3D path from \( X_i \) to \( X_j \). Two voxels, \( x_{src} \in X_i \) and \( x_{dat} \in X_j \), which define the shortest distance between \( X_i \) and \( X_j \), represent respectively the source and destination for 3D pathfinding. We define a heuristic function for guiding pathfinding as follows:

\[
h(x_i) = \sum_{k=1}^{n} D_k(x^k_{i}) + 0.6\|x_i - x_{dat}\|^2.
\]

The first term encourages selecting a voxel with smallest projection error, while the second term regularizes the tracing direction toward the destination voxel \( x_{dat} \). The resulting 3D path, composed of voxels that connect the two components \( X_i \) and \( X_j \), is denoted as \( P_{ij} \).

Constrained 3D pathfinding. Note that a greedy pathfinding algorithm that freely explores the volume space is computationally expensive. Here, we exploit the mapping between the target visual hull and the input images to construct a volumetric manifold to which the searching space is restrained while maintaining the projection quality. Specifically, given a pair of components \( \{X_i, X_j\} \) and corresponding projections \( \{\hat{X}_i^k, \hat{X}_j^k\} \) by \( P_k \), we compute a 2D shortest path from \( \hat{X}_i^k \) to \( \hat{X}_j^k \) on the \( k \)-th input image \( G_k \), and back-project pixels in the shortest path to obtain a constraint volume, denoted as \( H_{ij}^k \). We define the constraint volume as \( H_{ij} = \bigcup_{k=1}^{n} H_{ij}^k \), where \( n \) is the number of input images. Intuitively, as the constraint volume \( H_{ij}^k \) has projections along the contour on \( k \)-th image, tracing path within the constraint volume thus does not incur additional projection errors while pruning out a large portion of the volume space. We modify the pathfinding algorithm to consider only voxels in \( H_{ij} \) when connecting components \( X_i \) and \( X_j \) and hence obtain a significant boost in timing (Section 7.2). Figure 6 illustrates the proposed constrained 3D pathfinding approach.

Shape-aware edge weight. We define the edge weight \( w_{ij} \) in the component graph as follows:

\[
w_{ij} = \frac{1}{|P_{ij}|} \sum_{\forall x_i \in P_{ij}} \sum_{k=1}^{n} D_k(x^k_{i}) + D_{cont}(X_i, X_j).
\]

Intuitively, the first term measures the projection error induced from path \( P_{ij} \) with respect to the input images. Here, we introduce the continuity penalty term, \( D_{cont}(X_i, X_j) \), to avoid endpoints shrinkage effects due to volumetric thinning as follows:

\[
D_{cont}(X_i, X_j) = \begin{cases} 
0, & \exists k, s.t. \hat{X}^k_i \cap \hat{X}^k_j \neq \emptyset \\
10, & \text{otherwise}
\end{cases}
\]

\( D_{cont}(X_i, X_j) \) captures whether the 2D projections of two disjoint components \( X_i \) and \( X_j \) form a continuous line segment in the projective views. If so, we want to ensure a path connecting two such components. This is achieved by adding a large constant value (10 in Equation 4) to those components that do not present such a relationship. Figure 7 shows the effect of this penalty term.

Optimization. We apply Kruskal’s algorithm to the component graph \( G \) and obtain a minimum weight spanning tree. The final visual hull corresponds to the voxels that compose the resulting spanning tree.

5 3D CURVE EXTRACTION

In this section, we describe our method for extracting smooth and continuous 3D curves from the reconstructed visual hull. To this end, we first employ a volumetric thinning method [Liu et al. 2010] to extract a shape- and topology-preserving curve skeleton from the visual hull. This curve skeleton is then divided into a set of skeleton lines according to its topology. However, as the process of visual hull reconstruction is performed locally and does not take into account the compactness of global structure, the extracted curve skeletons may contain many redundant parts with complex geometry. This makes the physical realization process (e.g., 3D printing) extremely challenging, or not even possible. To tackle this problem, we perform a structure simplification on the curve skeleton that grounds on a novel quality measurement tailor-made for our context to strike a balance between projection error and structure compactness.
where $E$ with $w$ Two functions, ACM Transactions on Graphics, Vol. 37, No. 6, Article 242. Publication date: November 2018.

(5) middle support flexible user controls (Section 6).

$E$ computed by input error $E$ the skeleton lines on the projected view using a circle shape with $E$ recovered by the projection of curve skeleton. Note that we render deviation estimate (i) the and input image in the projected view. Specifically, we propose to to control the relative weights between the two energy terms.

$L$ represents the skeleton line that is composed of a sequence of skeleton lines $L$. Given a curve skeleton denoted as $L$, we can easily obtain a smooth and continuous wire sculpture by fitting the individual skeleton lines with a parametric curve using cubic splines. However, a naïve curve fitting method often leads to irregular curve shapes inherited from the zigzag features of reference skeleton lines (see Figure 8 (left, middle)). To avoid these artifacts, we devise a curve deformation model that iteratively...
We evaluate our method on a wide variety of input line drawings. While our method can produce a 3D wire sculpture that best approximates the given line drawings at the specified view in fully automatic fashion, the users may want to modify and refine the structure to enhance artistic effects of the structure or fix errors produced by our system. Here, we introduce two types of 2D strokes: (i) repair strokes and (ii) simplify strokes to support interactive user editing on the wire sculpture. The user simply needs to specify 2D strokes on the image of the wire sculpture projected onto the k-th input image, while our system will automatically update the local structure by (i) collecting skeleton lines that intersect with user strokes in the projected view; and (ii) re-running the structure simplification process on the selected skeleton lines with updated parameters. For the repair strokes, we increase the parameter $\omega_k$ in Equation 6 from 2 to 8 to encourage the completeness in the k-th input image. For the simplify strokes, we double the parameter $\alpha$ in Equation 5 to amplify the structure compactness during the simplification. We demonstrate the effect of user editing in Figure 9.

### 7 EXPERIMENTAL RESULTS

#### 7.1 Visual results

We evaluate our method on a wide variety of input line drawing images with varying complexity. We implement two types of exhibition modes: (i) orthogonal mode and (ii) in-plane mode. The former one arranges the virtual cameras in a mutually orthogonal setting, while the latter places all the cameras on the same plane with varying viewing angles. We automatically generated 20 multi-view wire sculptures in total. Figure 12 shows a gallery of the generated wire sculptures exhibited in orthogonal mode. Examples of in-plane exhibition mode can be found in Figure 1 (middle, right). In addition to static exhibition, the best way to demonstrate such a unique sculpture art is through a dynamic setting, where the 3D model, camera poses, and lighting are animated and working together to depict different input images over time. Please refer to supplementary material for more static and dynamic examples.

#### 7.2 Evaluation

In this section, we evaluate and characterize the performance of our method over several important design choices. The source code, input line drawings, and pre-computed wire sculptures will be made publicly available to stimulate further research.

**Dataset and evaluation metrics.** In addition to the triple-image examples shown in the paper, where the input images are from the same category and have similar complexity and style, we further prepared an extra set of 21 triple-image cases by randomly picking three images out of a large set of 45 input line drawing images with different styles and varying complexity.

We use two quantitative metrics to evaluate the quality of generated 3D wire sculptures. Our first metric measures the projection errors: $E_{dev} + E_{incomp}$ (Section 5.1). The project errors, however, may not reflect the quality well due to its sensitivity to outliers. Our second metric measures the projection accuracy by mean average precision (mAP). Specifically, we vary the distance threshold ranging from 0 (precise matching) to 10 pixels (coarse matching), compute the precision and recall at each step, and compute the mAP value across all images.
Fig. 12. **Visual results of multi-view wire art.** We present several examples of input line drawings with varying complexity. Given a set of three input images (middle), our system automatically generates 3D wires that exhibits three distinct line drawings when perceived from three orthogonal view points (left, right). Our method handles inputs with varying complexity. More results can be found in the supplementary material.
Multi-view Wire Art

Resolution of voxelization. As the initial geometry and structure of 3D curve skeleton are determined by the reconstructed visual hull, voxel resolution plays a critical role in recovering the details of the visual hull, particularly due to the intricate details presented in the line drawings. We characterize the performance over voxel resolutions $N^3 = [512^3, 256^3, 128^3, 64^3]$ in orthogonal mode. Figure 13 shows a visual example as well as the quantitative results. Our results validate the intuition that increasing voxel resolution allows us to capture finer details in the inputs. However, this comes at the cost of memory and computational complexity. We believe that using an octree-based method can better address the trade-off than the uniform voxelization in this work.

Effect of angle intervals. When placing all the three viewpoint on the same plane, the reprojection accuracy depends heavily on the selected angle intervals. The similarity between the respective projections and the input images degrades significantly when angle intervals are small (because it is difficult to optimize the 3D wire with dramatically different projections under a small view point shift) or close to 90 degrees (because the first and the third views will be located at opposite viewing directions and create conflicting contours).

Effect of parameter $\alpha$. In Section 5.1, we use a parameter $\alpha$ to control the relative importance between the projection error and structure compactness during the structure simplification process. Figure 14 shows the effect of $\alpha$ by generating curve skeleton with varying $\alpha$. Setting $\alpha$ to either small value (0.1) or large value (0.9) may produce over-complex or over-simplified results. Our default setting with $\alpha = 0.4$ (fixed throughout all of our experiments) strikes a balance between projection error and structure compactness.

Viewing angles. In the in-plane mode with three images, we investigate the effect of angle interval and report the evaluation results in Figure 15. In the case of three input drawings, placing the viewpoints at an angle interval of 60 degrees achieves the best visual and quantitative results.

Quality indicator. (Left) Scatter plot for the 21 cases. (Right) Two examples of lowest and highest value of $Q_{\text{metric}}$. The voxels of initial visual hull and expanded visual hull are colored in blue and red, respectively. Our quality indicator correlates well with the final projection errors of the wire sculptures.
Quality indicator. To evaluate how well the quality of the wire sculpture relates to the initial discrete visual hull, we propose a quality indicator metric, \( Q_{\text{metric}} = \frac{|VH_{\text{initial}}|}{|VH_{\text{expanded}}|} \), which computes the ratio between the number of voxels in the initial visual hull \( VH_{\text{initial}} \) and that in the expanded visual hull \( VH_{\text{expanded}} \) (see Figure 5). Higher values of \( Q_{\text{metric}} \) indicate more triple-consistent voxels in the initial visual hull. For the 21 test cases used for evaluation, we found that the ratio correlates well with the projection errors with a Spearman’s rank correlation coefficient = -0.818. Figure 16 shows the scatter plot of all 21 test cases. As computing the ratio \( Q_{\text{metric}} \) is efficient (< 1 minute), we can use it as quick test for quality, which allows us to filter out challenging cases that may lead to failures.

Timing analysis. Here we report the timing statistics of running our non-optimized method on the dataset using a moderate PC with an Intel Core i7 6700K (3.4GHz) and 32GB memory. The overall time complexity depends on the voxel resolution and the complexity of input images. For example, the average running time for simple images (e.g., 3rd row in Figure 12) and complex images (e.g., 1st row in Figure 12) with resolution 512\(^3\) are 83 and 510 minutes, respectively. Decreasing the resolution to 256\(^3\) will yield a 7X speedup. The major computational bottleneck lies in the connectivity optimization step in Section 4.2 (73% of the total running time). Using the proposed constrained 3D pathfinding algorithm provides more than 2X speedup. The structure simplification (Section 5.2) that occupies 27% of running time mainly depends on the complexity of input images. The average running time for generating discrete visual hull (Section 4.1) and curve fitting and deformation (Section 5.3) took about 1 – 3 seconds and is thus negligible. Employing a hierarchical multi-scale implementation and parallelizing the sequential algorithm of 3D pathfinding can further speed up the system.

Comparison with Shadow Art. We ran the code of [Mitra and Pauly 2009] on 11 three-view examples. As shown in Figure 17, the resultant voxel structures generated by [Mitra and Pauly 2009] may contain many disjoint components due to the highly inconsistent nature of input line drawings. Moreover, a severely distorted projection is often inevitable due to an image deformation approach for eliminating inconsistent voxels. Our system, in contrast, can generate voxel structures with one single connected component without introducing significant projection errors. Please refer to the supplementary material for a complete set of comparison.

7.3 Limitations

Our method has limitations in the following aspects.

Results. Surprisingly, we find that the our method does not perform well on simple input images. Figure 18 shows an example in orthogonal mode. In this case, our method produces clearly visible artifacts due to the difficulty in resolving inconsistency from simple contours. Our approach also has difficulty in resolving dense line strokes (e.g., hairs) in the projected image due to limited voxel resolution.

Speed. Our current implementation may take up to several hours to construct a multi-view wire sculpture (depending on the chosen voxel resolution and input line drawing complexity). Once the initial
wire sculpture is constructed, the user editing (Section 6) can run at interactive rate.

Optimization. While our results validate the capability of our algorithm for generating multi-view wire sculptures, the proposed method involves on several heuristic (and therefore brittle) optimization steps. Formulating the problem as a more principled global optimization framework is an important future direction.

Fabrication. Our system does not incorporate any physical simulations nor impose the single wire assumption as in [Liu et al. 2017]. Consequently, we cannot guarantee the generated wire sculptures are 3D printable. Possible ways to achieve fast and inexpensive 3D printing include considering the constraints of the wire-bending machine in the modeling process [Miguel et al. 2016], or decomposing the wire sculpture into a set of single wires and fabricating each wire separately.

8 CONCLUSIONS

We have presented a method for enabling novice users to construct multi-view wire sculptures. The core idea of our approach lies in reconstructing a visual hull and extracting a curve skeleton to balance projection error and spatial compactness. Through extensive evaluations in both simulation and 3D physical printout, we show that our method can produce smooth and compact wire sculptures exhibiting multiple prescribed images with minimum distortions. We believe that our framework can help democratize this unique art form and enable artists, designers, and hobbyists to create their own multi-view wire sculptures. We see several interesting future directions. This work focuses on creating wire sculptures with one connected component. Creating sculptures with multiple parts may provide more complex interactions among parts and offer richer viewing experiences. Our current approach resolves the inconsistency among input line drawings through minimizing the projection errors based on the original input images. An semantic-aware method may help significantly improve the visual quality by favoring projected lines that are plausible in the input line drawings.

ACKNOWLEDGMENTS

We are grateful to the anonymous reviewers for their comments and suggestions. We also like to thank National Applied Research Laboratories and Xian-Guang Zhong for helping on the 3D printing. The work is supported in part by the Ministry of Science and Technology of Taiwan (107-2218-E-007-047- and 107-2221-E-007-088-MY3).

REFERENCES


Ricardo Fabbi and Benjamin Kimia. 2010. 3d curve sketch: Flexible curve-based stereo reconstruction and calibration. In CVPR.


Andi Ussumzras, Ricardo Fabbi, and Benjamin Kimia. 2016. From multiview image curves to 3D drawings. In ECCV.

