

# Reconstructing Thin Structures of Manifold Surfaces by Integrating Spatial Curves

Shiwei Li<sup>1</sup>

Yao Yao<sup>1</sup>

Tian Fang<sup>2</sup>

Long Quan<sup>1</sup>

{slibc|yyaoag|quan}@cse.ust.hk fangtian@altizure.com

<sup>1</sup>The Hong Kong University of Science and Technology <sup>2</sup>Altizure.com

## Abstract

The manifold surface reconstruction in multi-view stereo often fails in retaining thin structures due to incomplete and noisy reconstructed point clouds. In this paper, we address this problem by leveraging spatial curves. The curve representation in nature is advantageous in modeling thin and elongated structures, implying topology and connectivity information of the underlying geometry, which exactly compensates the weakness of scattered point clouds. We present a novel surface reconstruction method using both curves and point clouds. First, we propose a 3D curve reconstruction algorithm based on the initialize-optimize-extend strategy. Then, tetrahedra are constructed from points and curves, where the volumes of thin structures are robustly preserved by the Curve-conformed Delaunay Refinement. Finally, the mesh surface is extracted from tetrahedra by a graph optimization. The method has been intensively evaluated on both synthetic and real-world datasets, showing significant improvements over state-of-the-art methods.

## 1. Introduction

One of the general weaknesses of existing dense multi-view stereo (MVS) methods is the thin structure reconstruction. Fig. 1(a) shows a typical example generated by the state-of-the-art method in [48]: although the ground and the windmill body are well reconstructed by point clouds and meshes, thin windmill blades remain severely incomplete.

The main reason is that most stereo algorithms perform dense matching on local image patches. The local appearance of thin structures becomes highly varied due to background changes, making the multi-view matching inconsistent (illustrated in Fig. 1(c)). Also, point clouds are scattered and imply no connectivity information of the underlying geometry. Even if some dense points are recovered for thin structures, they are usually sparse and incomplete, and thus prone to be filtered out as outliers when extracting the manifold surface (see the effect from dense points to meshes in Fig. 1(a)). Moreover, thin structures are sensitive

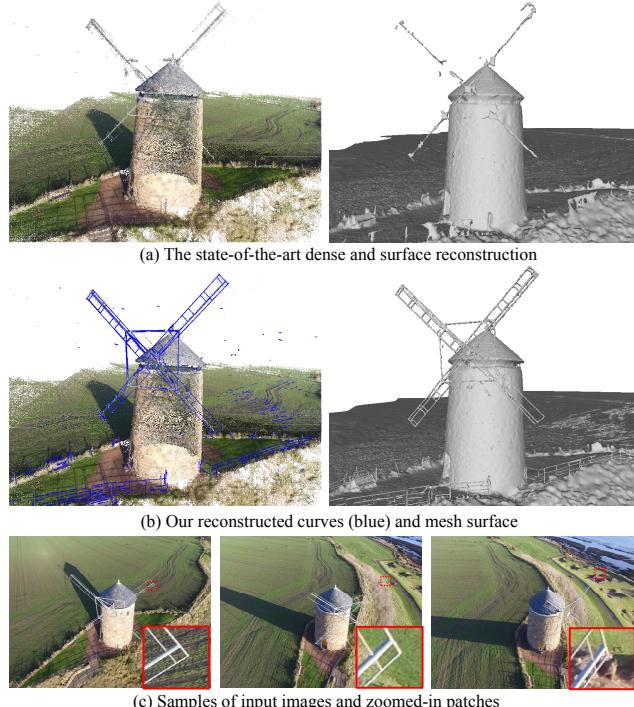


Figure 1: (a) The point cloud and mesh reconstruction by [48]. (b) Our method reconstructs 3D curves in blue on the left and a complete mesh on the right. (c) Samples of input images. The zoomed-in patches of the windmill blade vary significantly due to background changes, resulting in the incomplete surface in (a).

to the accuracy of camera geometry. If the reprojection error of matching points is larger than the thickness of thin structures themselves, they can hardly be reconstructed [12].

To compensate the essential weaknesses of point-based methods, we resort to another representation – 3D curves. The curves can be computed from 2D edges of images. It's known that edges are more robust to image noises (in detection [44]), more tolerant to SfM errors (in matching [32]) as they are sparse and distinctive in nature, better in characterizing the topology of thin and elongated structures, and providing connectivity information which is useful as a constraint in surface reconstruction. Unlike dense point

Representation	3D points	3D curves
Reconstructed from	Local image patches (5-7 px)	Image edges (1 px)
Sample of surface	Local pointwise	1D continuous
Properties	Dense, scattered	Sparse, connected
Suitable for	Smoothly-varying or planar surfaces	Thin structures, object silhouettes

Table 1: Comparisons between two representations.

matching that relies on the appearance of local image patch, the edge detection is much less affected by background changes, yielding stable reconstructions of spatial curves. Also, the edge can depict thin geometries down to single pixel width, while the local image patch usually resides in 5 to 7 pixels. All these advantages are exactly complementary to weaknesses of scattered point clouds. Table 1 summarizes the comparison between them. With a proper combination, two representations will benefit from each other.

In this paper, we demonstrate that leveraging spatial curves can dramatically improve the thin structure reconstruction. We propose a complete solution using spatial curves for surface reconstruction. First, 3D curves are generated from image edges based on the initialize-optimize-extend strategy. Then, the confidence region of curves is estimated. Next, these curves are combined with point clouds (generated by MVS) to form the tetrahedra, which is improved by the *Curve-conformed Delaunay Refinement*. Finally, the mesh surface is extracted from the tetrahedra via a graph optimization considering energy terms derived from both points and curves. Our experiments show that the additional spatial curve is crucial to successful thin structure reconstructions, significantly outperforming point-based state-of-the-art methods.

The main contributions of this paper include:

- the novel approach to compute 3D curves based on the initialize-optimize-extend strategy (Section 2);
- the Curve-conformed Delaunay Refinement for tetrahedra, which is crucial to preserve the reconstruction of thin structures (Section 3.2);
- the unified Delaunay tetrahedra and graph optimization framework to combine point clouds and 3D curves for manifold surface extraction (Section 3.3).

At last, we note that our work only aims at improving the reconstruction of 1D thin structures (*e.g.*, pylon, wires), instead of tackling with 2D thin structures (*e.g.*, paper, road signs) as in previous works [45, 1]. Also, the definition of “thin structures” is relative to image resolution: if the resolution is sufficiently high, or equivalently, images are captured from close distance, the thin structure becomes not that “thin” anymore and can be recovered by point-based methods. Nevertheless, our proposed method has shown strong ability to reconstruct very thin structures that are unable to be recovered by previous point-based methods.

## 1.1. Related Works

**Fine structure reconstruction.** Most dense reconstruction methods approximate surfaces with local planes (*i.e.*, point plus normal), or referred to as 3D patches [15], which work well for planar and smooth-varying areas, but they struggle with thin, elongated or wiry structures. Some previous works [45, 1] have improved 2D thin surface reconstruction, either using point-based non-manifold representation [45] or multi-scale kernels [1], showing promising results. But neither of them excels at recovering 1D thin structures, which is a more challenging problem since the point cloud is often incomplete in MVS. Li *et al.* [27] and Huang *et al.* [20] proposed to fit the point cloud of 1D thin structures into cylindrical skeletons, also requiring laser-level density of point cloud. Yucer *et al.* [50] exploited segmentation method on dense video frames to recover the thin structures, which is not applicable to sparse image input. Some volumetric methods have also shown impressive results in thin structure reconstruction, using direct ray reasoning [29, 35] or silhouette information [23, 8, 33, 37]. However, the volumetric model is initialized by the fusion of multiple depth maps computed from stereo matching, which still suffers from background changes problem as mentioned before. Besides, volumetric methods are in nature not suitable in recovering thin structures as the thinness is eventually limited by the resolution of volumes (the depth of Octree [30]). Our method effectively averts these two problems by directly using 2D edges to generate 3D curves, and a flexible Delaunay tetrahedra framework [25] for mesh surface reconstruction.

**Curve-based reconstruction.** The usages of image edges, curves and lines are ubiquitous in stereo vision, including camera motion problems [42, 9, 32], multi-view reconstruction [7, 24, 37] or sparse abstractions [10, 43, 19].

To mention most-related techniques in curve-based reconstruction, Teney *et al.* [43] proposed a spatial probability distribution for 3D curve generation. Fabbri *et al.* [10, 11] developed a curve sketch framework based on coarse-to-fine optimization, which was later extended [46] with a graph model for improved curve qualities. Other than that, Hofer *et al.* [19] focused on 3D line reconstruction, and proposed an efficient algorithm using hypothesis-then-verify scheme. Recently, Nurutdinova *et al.* [32] combined the 3D curves with feature points for improved accuracy of SfM. More recently, Liu *et al.* [28] presented the candidate selection strategy for 3D curve reconstruction, showing impressive results from sparse image input. These works share a similar goal to Section 2: computing the 3D curves. But the only work had considered surface reconstruction from curves is [47]. They aim at sparse surface abstraction by considering the curves as scaffolds of objects, while ours focuses on dense surface reconstruction by combining points and curves for improved thin structure reconstruction.

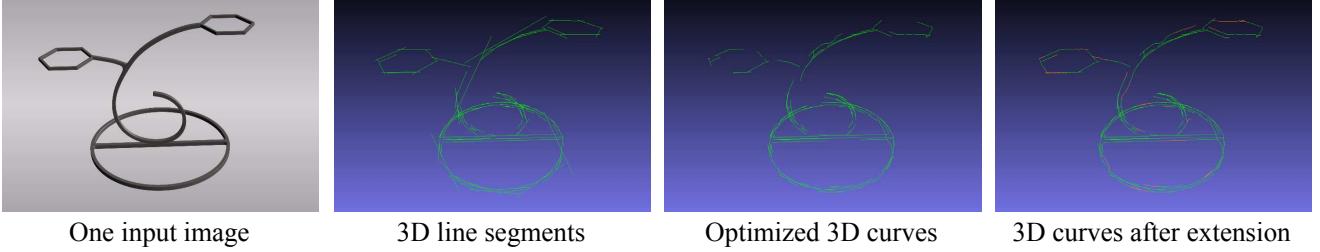


Figure 2: The 3D line segments are first computed from multi-view images, and then optimized to accurate 3D curves, followed by an extension step to increase the completeness (orange highlights the extended parts of curves).

## 2. Reconstruction of Spatial Curves

In this section, we present our 3D curve reconstruction based on the initialize-optimize-extent strategy. To avoid confusion, we refer the term “curve” to 3D curves, and “edge” to 2D image edges in the whole context of the paper. The effects of each step are illustrated in Fig. 2.

**Initialization from lines.** As edges can be too abundant in natural images, direct reconstruction from all edges leads to extremely large search space and massive mismatches due to ambiguities [43, 10]. In contrast, we initialize from straight edges (line segments) first. Because line segments are special cases of edges, more confident and prominent, this particular choice dramatically reduces ambiguities in matching. Let the input images be  $\{\mathbf{I}\} = \{\mathbf{I}_0, \mathbf{I}_1, \dots\}$  (the set expansion is omitted below) and camera parameters  $\{\theta\}$ . We apply Edge Drawing [44] on images to compute 2D edge maps  $\{\mathbf{E}\}$  and line segments. These 2D line segments are then fed into the Line3D [19] algorithm to compute 3D line segments  $\{\mathbf{L}\}$ , regarded as the initialization of curves. Note that  $\{\mathbf{L}\}$  are strictly straight and thus fail to represent the accurate geometry. The following optimization and extension would improve the accuracy and completeness.

**Optimization to curves.** To compute 3D curves  $\{\mathbf{C}\}$  that best describe 2D edge maps  $\{\mathbf{E}\}$ , we formulate this problem as a model-to-data alignment. Let an arbitrary 3D point lie along the curve represented by a parametric equation  $\mathbf{C}(t) \subset \mathbb{R}^3, t \in [0, 1]$ , the alignment cost for one curve  $\mathbf{C}$  is the integral error of all points on the curve:

$$\begin{aligned} E(\mathbf{C}) &= \sum_{i=1}^N E_i(\mathbf{C}) \\ &= \sum_{i=1}^N \int_0^1 \min_{\mathbf{x}' \in \mathbf{E}_i} \|\Pi(\theta_i, \mathbf{C}(t)) - \mathbf{x}'\| dt, \end{aligned} \quad (1)$$

where  $i$  iterates  $N$  visible views of the curve,  $\Pi(*, *)$  means camera projection and  $\mathbf{x}'$  is the closest edge point to the projected point. The closest distance can be efficiently queried from a pre-computed map  $\mathbf{I}_{dist}$ , which is generated by flood-filling the edge map with L2-distance from every pixel to its closest edge pixel, *i.e.*,  $\mathbf{I}_{dist}(\mathbf{x}) = \min \|\mathbf{x} - \mathbf{x}'\|, \mathbf{x} \in$

$\mathbf{I}, \mathbf{x}' \in \mathbf{E}$ . The Eq. 1 converts to:

$$E(\mathbf{C}) = \sum_{i=1}^N \int_0^1 \mathbf{I}_{dist_i} \circ \Pi_{\theta_i} \circ \mathbf{C}(t) dt. \quad (2)$$

To minimize the energy in terms of a general curve representation  $\mathbf{C}$ , we compute the variation with respect to an infinitesimal displacement  $\delta\mathbf{C}$  of the curve:

$$\lim_{\epsilon \rightarrow 0} \frac{\partial E(\mathbf{C} + \epsilon\delta\mathbf{C})}{\partial \epsilon} = \sum_{i=1}^N \int_0^1 D_{\mathbf{I}_{dist_i}} J_{\Pi_{\theta_i}} \frac{\partial(\mathbf{C} + \epsilon\delta\mathbf{C})(t)}{\partial \epsilon} dt, \quad (3)$$

where  $D_{\mathbf{I}_{dist_i}}$  is the gradient map of  $\mathbf{I}_{dist_i}$ , and  $J_{\Pi_{\theta_i}}$  the Jacobian of projection matrix. The optimization gradient can be found by converting the equation to  $\nabla E \cdot \delta\mathbf{C}$  format, depending on the concrete parameterization of curve.

In practice, we parameterize the curve as a chain of anchor points  $\mathbf{C} = \{\mathbf{A}_0, \dots, \mathbf{A}_{M-1}\}, \mathbf{A}_i \in \mathbb{R}^3$ , discretized evenly from the initial 3D line segment  $\mathbf{L}$ . The number of anchor points  $M$  is determined by a constant ratio  $\lambda$  multiplying the line segment’s longest projection length in pixel unit. We have  $\lambda = 0.1$  in our experiments. Then, we optimize the positions of all anchor points by minimizing the sum of data-term and smoothness-term. Based on Eq. 3, the gradient of data-term of each anchor point  $\mathbf{A}$  is:

$$\mathbf{v}_{data}(\mathbf{A}) = \frac{\partial E(\mathbf{A})}{\partial \mathbf{A}} = \sum_{i=1}^N w_i \cdot D_{\mathbf{I}_{dist_i}(\mathbf{x}_A)} \cdot J_{\Pi(\theta_i, \mathbf{A})}, \quad (4)$$

where  $\mathbf{x}_A = \Pi(\theta_i, \mathbf{A})$  is the projected 2D point of  $\mathbf{A}$ ,  $w_i = depth_i(\mathbf{A})/f_i$  the contribution weight of view  $i$ , which is the ratio between the depth and focal length  $f_i$  in pixels.

For the gradient of smoothness, we mimic the elastic effect using a 1D Laplacian operator over adjacent anchor points:

$$\mathbf{v}_{smooth}(\mathbf{A}_i) = \frac{1}{2}(\mathbf{A}_{i+1} + \mathbf{A}_{i-1}) - \mathbf{A}_i. \quad (5)$$

The gradient is therefore  $\nabla E(\mathbf{A}) = \mathbf{v}_{data} + \beta\mathbf{v}_{smooth}$ , where  $\beta = 0.5$  is the weighting of smoothness. The step

size of optimization is set to be one pixel on average to keep the convergence stable. In most cases the optimization converges in 20 iterations (at most 20 pixels from initialization to the local optima). After optimization, we filter anchor points of curve  $\mathbf{A}$  if there exists a projected point with distance to image edges larger than 20 pixels, i.e.,  $\exists 1 \leq i \leq N : \mathbf{I}_{dist}(\Pi(\theta_i, \mathbf{A})) > 20$ .

**Completion by extension.** Since curve optimization is based on initialized straight edges, which are less complete than actually existing edges, an extension step is necessary to increase the completeness. We repeatedly create new points  $\mathbf{A}_{ext.0/1}$  by extending the optimized curve  $\mathbf{C} = \{\mathbf{A}\}$  from its outward directions of two endpoints:

$$\begin{aligned} \mathbf{A}_{ext.0} &= \mathbf{A}_0 + \overrightarrow{\mathbf{A}_1 \mathbf{A}_0} \\ \mathbf{A}_{ext.1} &= \mathbf{A}_{last} + \overrightarrow{\mathbf{A}_{last-1} \mathbf{A}_{last}}. \end{aligned} \quad (6)$$

Then the position of extended points  $\mathbf{A}_{ext.0/1}$  are refined using Eq. 4 and 5. The extension terminates when reaching to a dead end of any edge map  $\mathbf{E}$  or the residual of optimization  $E(\mathbf{A})$  is larger than a certain threshold  $t_{opt} = 2.5$ .

### 3. Manifold Surface Reconstruction

In this section we propose a manifold surface reconstruction using 3D curves and point clouds. This task is non-trivial as points and curves are two different representations. While a 3D point is a local sample of surface, a 3D curve is a 1D continuous sample of surface. The reconstructed curves can be categorized into three types (Fig. 3):

- a) the occluding contours of thin structures;
- b) the occluding contours of significant volumes with sharp geometries;
- c) the sharp texture (or shadows) on a planar or smooth surface, containing no geometric information.

Note that the occluding contour in general is dynamic in different views. However, for high-curvature geometries such as (a) and (b), their occluding contours are stationary and can be reconstructed by the algorithm in Section 2.

The main intention of integrating curves is to strengthen the reconstruction of type (a). While type (b) and (c) can be well reconstructed by traditional point-based methods, these curves are expected to fuse with MVS points and do not affect the original surface quality. A preliminary result is shown in Fig. 3.

One straightforward idea is to crudely sample the points from curves and regard them as regular dense points. However, the valuable connectivity information would be deprecated and the normal directions remain unknown, which are useful in many surface reconstruction methods [2].

Instead, we propose a curve-conformed Delaunay-based surface extraction approach. First, we estimate the confidence region of curves (Section 3.1). Then, we construct the

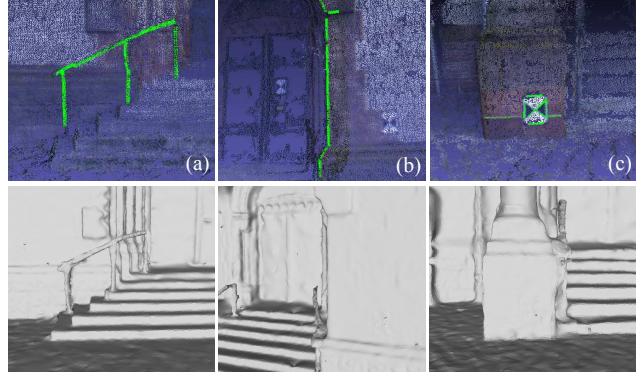


Figure 3: Three types of geometries a 3D curve can represent. Upper: curves and dense points. Lower: our surface meshes. The curves (a) contribute to the solid reconstruction of thin handrails, while curves (b) (c) are fused into surface. Data: *Herjesu-25* [40]

tetrahedra using points and curves, and optimize it based on three criteria in order to recover the thin structures inside tetrahedra (Section 3.2). Finally, the mesh surface is extracted from tetrahedra using s-t cut by minimizing energy derived from both points and curves (Section 3.3).

The advantages of the proposed method are threefold. 1) For curve type (a), the thin structure can be flexibly modeled by the adaptive tetrahedron. 2) For type (b) and (c), the fusion with MVS points is straightforward in the Delaunay tetrahedra framework [25]. 3) Compared with implicit function methods (e.g., Poisson [22]), it does not require the input normal orientation, which is hard to define for curves as they might represent three different types of geometries.

#### 3.1. Confidence region of curves

Due to pixelation errors, a 3D point triangulated from 2D points does not represent only one point, but a certain region in space, referred to as *covariance* [18] or *uncertainty* [12] in previous literatures. Here, we call it *confidence region* (*cr*). Likewise, the curve has such region surrounding it that can be estimated from images. As pixelation effect of one curve differs on each anchor point, we approximate the *cr* of a small curve segment  $\overline{\mathbf{A}_i \mathbf{A}_{i+1}}$  to be *circular truncated cone*. For each anchor point  $\mathbf{A}$ , its radius  $r(\mathbf{A})$  of *cr* conforms to average half image pixel size, computed as:

$$r(\mathbf{A}) = \frac{1}{2N} \sum_{i=1}^N \left( \frac{\text{depth}_i(\mathbf{A})}{f_i} \right), \quad (7)$$

where  $i$  iterates  $N$  visible views,  $f$  is the focal length in pixels. Then, the curve segment  $\overline{\mathbf{A}_i \mathbf{A}_{i+1}}$  has a *cr* of *circular truncated cone* with two bottoms of radius  $r(\mathbf{A}_i)$  and  $r(\mathbf{A}_{i+1})$ . We test if a point  $\mathbf{P}$  is inside this segment's *cr* by:

$$\|\mathbf{P} - \mathbf{P}'\| \leq r(\mathbf{A}_i) \cdot \alpha_{\mathbf{A}_i} + r(\mathbf{A}_{i+1}) \cdot (1 - \alpha_{\mathbf{A}_i}), \quad (8)$$

where  $\mathbf{P}'$  is a point on  $\overline{\mathbf{A}_i \mathbf{A}_{i+1}}$  closest to  $\mathbf{P}$ , and  $\alpha_{\mathbf{A}_i} = \frac{\|\mathbf{P}' - \mathbf{A}_{i+1}\|}{\|\mathbf{P}' - \mathbf{A}_i\| + \|\mathbf{P}' - \mathbf{A}_{i+1}\|}$ . Thus the *cr* for the entire curve is a

combination of circular truncated cones. In our surface reconstruction, we enforce the constraints within the space of  $cr$  in order to preserve the fragile topology of thin structures (Section 3.2 and 3.3).

### 3.2. Curve-conformed Delaunay refinement

To give brief backgrounds, *Delaunay Tetrahedralization* (DT) constructs the tetrahedra  $\mathbf{T}$  from 3D points. The task of surface reconstruction is to determine the binary label of each tetrahedron  $\tau \in \mathbf{T}$  indicating if it is inside or outside the surface, where the in-between triangles are the mesh surface  $\mathbf{M}$ . As mentioned, the MVS points (denoted as  $\{\mathbf{P}_{mvs}\}$ ) at thin structures are often sparse and incomplete, which tend to produce large or badly-shaped tetrahedron cells and make it difficult to recover the thin structure with correct topology. Therefore, our goal here is to refine the tetrahedra  $\mathbf{T}$  by adding extra vertices (called *Steiner points*). The rules of adding *Steiner points* follow three criteria.

**Curve conformity.** It is important that the mesh topology inside the tetrahedra conforms the curves, *i.e.*, every curve segment  $\overline{\mathbf{A}_i \mathbf{A}_{i+1}}$  exists in tetrahedra  $\mathbf{T}$ . Inspired by Ruppert’s *encroachment removal* [34], we recursively split the curve segment in half if it does not appear in  $\mathbf{T}$  until all segments are composed by one or multiple edges in  $\mathbf{T}$ .

**Tetrahedron shape.** To measure the shape quality of the tetrahedron  $\tau$ , we use the *radius-edge ratio*  $\rho(\tau) = r/l_{min}$ , where  $r$  is the radius of circumsphere and  $l_{min}$  is the shortest side of  $\tau$ . The range of  $\rho(\tau)$  is between  $\frac{\sqrt{6}}{4}$  (when  $\tau$  is regular tetrahedron) and  $+\infty$ . As suggested in [38],  $\rho(\tau) \leq \rho_0 = 2$  is assumed to be good shape.

**Tetrahedron volume.** As the confidence region of curve segments is very local, the tetrahedra near them are preferably small enough so that thin structures can be extracted from these tetrahedra. We control the volume by constraining the side length of tetrahedron: we recursively split the input curve segment  $\overline{\mathbf{A}_i \mathbf{A}_{i+1}}$  in half if it is longer than a parameter  $k$  multiplying their average radius of confidence regions  $(r(\mathbf{A}_i) + r(\mathbf{A}_{i+1}))/2$ . The value of  $k$  controls the volume size of tetrahedron near curve segments. Smaller value of  $k$  gives denser tetrahedralization meanwhile introduces more *Steiner points*.  $k = 0.5$  is a good balance.

We combine above criteria into the algorithm illustrated in Algorithm 1. First, we split the curve segment by half if it is relatively long. Then, the tetrahedra  $\mathbf{T}$  is constructed from anchor points of curve  $\{\mathbf{A}\}$ , followed by a while loop. Inside the while loop, the *Steiner point*  $v$  is added according to the *curve conformity* and *tetrahedron shape* criteria, where the former criterion has higher priority. This is to ensure that the output tetrahedra conform the curve so that we can query the tetrahedron and assign graph energies (in Section 3.3). The while loop terminates when no more *Steiner*

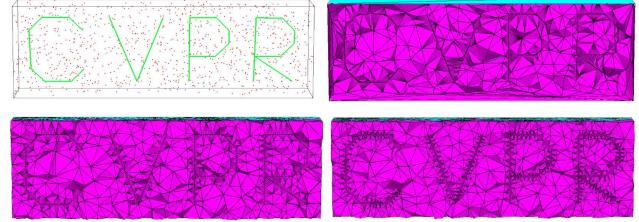


Figure 4: The Delaunay refinement process. Upper: the input curves (we add random points in red for better visualization of the initial tetrahedra), the initial tetrahedra after splitting long segments. Lower: the intermediate and final refined tetrahedra. Here, the *confidence region* of the “CVPR” curve is manually given.

---

#### Algorithm 1 Curve-conformed Delaunay refinement

---

```

// input: MVS points { $\mathbf{P}_{mvs}$ }
//       Curve segments { $\overline{\mathbf{A}_i \mathbf{A}_{i+1}}$ }
// params:  $k = 0.5$ ,  $\rho_0 = 2$ 
while ( $\exists \|\overline{\mathbf{A}_i \mathbf{A}_{i+1}}\| > k \cdot \frac{r(\mathbf{A}_i) + r(\mathbf{A}_{i+1})}{2}$ ):
    split  $\overline{\mathbf{A}_i \mathbf{A}_{i+1}}$  in half           // tetrahedron volume
endwhile
initialize the tetrahedra  $\mathbf{T}$  from anchor points { $\mathbf{A}$ }
while ( $\exists \overline{\mathbf{A}_i \mathbf{A}_{i+1}}$  not in  $\mathbf{T}$ ) or ( $\exists \rho(\tau) \leq \rho_0$ ):
    if ( $\overline{\mathbf{A}_i \mathbf{A}_{i+1}}$  not in  $\mathbf{T}$ ):          // curve conformity
         $v$  = midpoint of  $\overline{\mathbf{A}_i \mathbf{A}_{i+1}}$ 
    elif ( $\rho(\tau) \leq \rho_0$ ):           // tetrahedron shape
         $v$  = circumsphere center of  $\tau$ 
    endif
    add  $v$  into  $\mathbf{T}$  using Lawson’s flip [26].
endwhile
insert { $\mathbf{P}_{mvs}$ } into  $\mathbf{T}$ 
// output: final tetrahedra  $\mathbf{T}$ 

```

---

point can be added. Finally we insert MVS points  $\{\mathbf{P}_{mvs}\}$  back to  $\mathbf{T}$ . The process is visualized in Fig. 4.

### 3.3. Graph-based extraction of manifold surface

The goal of extracting mesh surface  $\mathbf{M}$  from tetrahedra  $\mathbf{T}$  is equivalent to deciding the binary labels of each tetrahedron belonging to inner or outer space of surface, where the in-between triangular facets are the mesh surface.

Previous methods [31, 21, 25, 49] using graph-cuts [5] to deal with tetrahedra binary labeling problem have shown excellent results. Here, we use the s-t cut framework similar to [25] to set up the graph of tetrahedra. Denote the directed graph as  $\mathbf{g} = (\mathbf{t}, \mathbf{f})$ , where each node  $\tau_i \in \mathbf{t}$  is a tetrahedron, and graph edge  $f_j \in \mathbf{f}$  is a triangular facet shared by two tetrahedra. The graph status is determined by the weight from each tetrahedron node  $\tau_i$  to interior and exterior of surface ( $In(\tau_i)$  and  $Out(\tau_i)$  respectively), as well as the directed smoothness  $f(\tau_1, \tau_2)$ . The final in-out labeling is carried out by s-t cut on three energy terms:

$$E(\mathbf{T}) = E_{point} + E_{qual} + E_{curve}. \quad (9)$$

$E_{point} + E_{qual}$  are derived from [25] to penalize visibility conflicts of MVS dense points as well as improve the surface quality. For conciseness, we rewrite the graph energy term as weighting accumulations:

$$\left\{ \begin{array}{ll} f_j(\tau_1, \tau_2) += w_{f_j} \cdot \alpha_{vis}, & \text{if } \tau_1, \tau_2 \text{ are incident to } f_j \text{ and} \\ & \text{a line of sight emits through } \tau_1, f_j, \tau_2. \\ In(\tau_i) += \alpha_{vis}, & \text{if } \tau_i \text{ is behind a line of sight.} \\ Out(\tau_i) += \alpha_{vis}, & \text{if } \tau_i \text{ contains a camera.} \end{array} \right.$$

where the *line of sight* refers to the vector from camera position to the reconstructed point,  $\alpha_{vis}$  is a constant parameter. The facet weighting  $w_{f_j} = \lambda_{qual}(1 - \min(\cos(\psi), \cos(\phi)))$ , where  $\psi$  and  $\phi$  are angles of  $f_j$  with the circumspheres of  $\tau_1$  and  $\tau_2$ ,  $\lambda_{qual}$  controls the global surface quality. In another word, it gives stronger smoothness connection between tetrahedra of better shape. By default  $\alpha_{vis} = \lambda_{qual} = 1.0$ .

$E_{curve}$  This is a novel term to enforce *connectivity constraints* and *space occupancies* for curves. First, we construct a sparse adjacency matrix whose rows and columns are curves and tetrahedron cells respectively. We denote the value of each element  $adj(\mathbf{C}, \tau) = 1$  if the center of tetrahedron  $\tau$  is inside the confidence region using the Eq. 8, and 0 otherwise.

To enforce *connectivity constraints*, we add smoothness weights by  $\beta$  to the facet  $f_j$  (on both directions) if two tetrahedra are inside the same curve and share  $f_j$ , i.e.,  $adj(\mathbf{C}, \tau_1) = adj(\mathbf{C}, \tau_2) = 1$ , and  $f_j = \tau_1 \cap \tau_2$ . We encourage tetrahedra belonging to the same curve to have the same label, i.e., a curve is either inlier or outlier. The value of  $\beta$  should be proportional to  $\alpha_{vis}$ . We found  $\beta = 3 \cdot \alpha_{vis} = 3.0$  is an effective value for removing apparent outliers.

To enforce *space occupancies*, we add weights to tetrahedron node  $In(\tau)$  by  $\alpha_{vis}$  if  $adj(\mathbf{C}, \tau) = 1$ . This simple operation naturally favors longer curves to be inlier because it cost more energy to label a long curve as outlier. Whereas the short curves are more likely to be outliers, it would be filtered if it has high conflicts to point visibilities. Lastly, we penalize visibility conflicts on the anchor points of curves. In sum, the weighting accumulation is:

$$\left\{ \begin{array}{ll} f_j(\tau_{1(2)}, \tau_{2(1)}) += \beta, & \text{if } \tau_1, \tau_2 \text{ are adjacent and} \\ & adj(\mathbf{C}, \tau_1) = adj(\mathbf{C}, \tau_2) = 1. \\ In(\tau_i) += \alpha_{vis}, & \text{if } adj(\mathbf{C}, \tau_i) = 1. \\ f_j(\tau_1, \tau_2) += w_{f_j} \cdot \alpha_{vis}, & \text{if } \tau_1, \tau_2 \text{ are incident to } f_j \text{ and} \\ & \text{a line of sight emits through } \tau_1, f_j, \tau_2. \end{array} \right.$$

Finally, we apply s-t cut on the graph to determine the binary labels and extract the mesh surface  $\mathbf{M}$ .

### 3.4. Discussions

**Delaunay quality.** We stress that the extracted mesh quality is closely related to the quality of Delaunay tetrahedra.

We can imagine tetrahedralization is the procedure to divide the space into tetrahedron cells. Better quality of tetrahedra means dividing the space in a more structured manner, which is equivalent to decreasing the discretion errors.

One may wonder if an alternative to our curve-conformed Delaunay refinement (Section 3.2) would be simply sampling the points in confidence regions of curves, and mix them with MVS points to construct the tetrahedra and surface mesh. We have compared with this approach in Section 4.1 and turned out it is quantitatively less superior than the proposed approach. Fig. 7 shows that good quality tetrahedra are more resistant to topological artifacts.

**Adaptive fusion.** The Delaunay tetrahedra framework makes fusion adaptive, including the fusion with MVS points and the self-intersections of curves (when two curves cross each other or one curve is reconstructed multiple times). In our algorithm, we treat all three types of curves uniformly. Fig. 3 shows the surface reconstruction from three types of curves. For type (a), the curve contributes to the solid reconstruction of handrails. For type (b) and (c), the curves are adaptively fused into the surface with MVS points. See experiments for more qualitative results.

**Thickness of thin structures.** One should not consider the curve as the body of thin structure. Instead, it is a 1D sample of the surface. We view it as the *topology proxy* to guide and enforce the correct reconstruction. Its *confidence region*, inferred from image edges, specifies the reasonable region in space that the *connectivity constraints* and *space occupancies* should take effect inside. This region alone does not determine the actual thickness. In practice, most thin structures are reconstructed by the fusion of dense points and multiple curves, and all of them may affect to the final thickness. The intention of adding the curve representation is mainly to enforce the solid and correct topology of initial mesh  $\mathbf{M}$ , instead of recovering the accurate thickness. As the initial mesh extracted from tetrahedra is usually noisy, a following mesh smoothing and refinement (e.g., Vu's method [48]) are applied to increase the accuracy. Note that the mesh of thin structure is highly prone to shrinkage in Laplacian smoothing, and thus a non-shrinking smoothing operator (e.g., Taubin's filter [41]) is suggested.

## 4. Experiment

We implement the algorithm in C++. We have used Edge Drawing [44] and Line3D [19] to initialize 3D curves. The tetrahedra is manipulated by CGAL [6] and TetGen [39].

### 4.1. Quantitative evaluations on synthetic dataset

Our method is first evaluated on six rendered datasets (Fig. 5) collected from *Blendswap* [4], which are representative objects of thin structures. We add random textures to

Method	chair	chair2	fence	holder	lamp	pylon
PMVS [15, 22]	76.60 / 82.47 / 79.47	50.70 / 80.5 / 62.22	40.28 / 59.28 / 47.97	29.95 / 51.56 / 37.89	29.22 / 69.74 / 41.18	15.71 / 81.89 / 26.36
MVE [14, 13]	64.65 / 80.72 / 71.80	45.48 / 49.82 / 47.55	37.04 / 44.25 / 40.33	20.74 / 39.86 / 27.28	26.89 / 55.41 / 36.21	20.12 / 70.2 / 31.28
Vu [16, 48]	<b>98.78</b> / 91.89 / 95.21	<b>70.48</b> / 79.78 / 74.84	<b>82.56</b> / 71.76 / 76.78	<b>92.98</b> / 94.51 / 93.74	<b>84.96</b> / 71.60 / 77.71	77.70 / 77.52 / 77.61
Ours-Line	91.28 / 92.12 / 91.70	65.28 / 82.52 / 72.89	74.68 / 83.90 / 79.02	82.28 / 94.76 / 89.21	73.86 / 74.68 / 74.27	78.80 / 83.47 / 81.07
Ours-Sample	93.60 / 90.20 / 91.87	67.82 / 84.50 / 75.25	76.82 / 87.20 / 81.68	87.85 / 93.58 / 90.63	77.69 / 77.15 / 77.42	78.74 / 82.37 / 80.51
Ours-Curve	97.75 / <b>93.64</b> / <b>95.65</b>	69.54 / <b>84.89</b> / <b>76.45</b>	80.85 / <b>87.61</b> / <b>84.09</b>	92.91 / <b>95.10</b> / <b>93.99</b>	80.72 / <b>78.15</b> / <b>79.41</b>	<b>89.13</b> / <b>85.10</b> / <b>82.01</b>

Table 2: Results of accuracy / completeness /  $F_1$  score (in %) at a certain evaluation threshold ( $0.001 \cdot \text{camera center to scene center}$ ). Higher is better. Overall, *Vu* method produces better accuracy, while *Ours-Curve* gives better completeness and  $F_1$  score among all six datasets. Also, *Ours-Curve* is better than *Ours-Line* and *Ours-Sample* in all measurements. See Section 4.1 for detailed explanations.

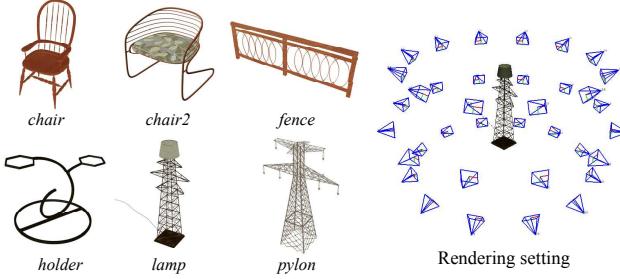


Figure 5: Six synthetic dataset and the rendering setting.

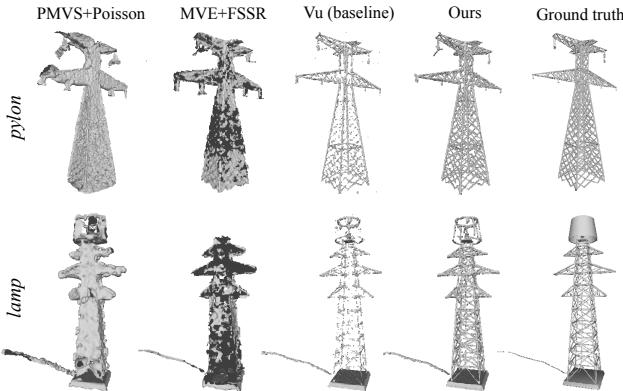


Figure 6: Qualitative comparisons of *pylon*, *lamp* datasets.

them and use the diffuse lighting in rendering. A total number of 36 images are rendered by the Blender software [3] at 4K resolution from spherical views. To quantitatively measure the difference between ground truth and reconstructed models, we apply the evaluation method in [36] to calculate the *accuracy* ( $p$ ), *completeness* ( $r$ ) and  $F_1$  score for all reconstructed meshes. We first sample a sufficiently dense point cloud from the mesh. The *accuracy* is defined as the percentage of sampled points of reconstructed mesh which are within a distance threshold to nearest sampled points of ground truth, and *completeness* the other way around. The distance threshold is set to  $0.001 \cdot r_{cam}$  where  $r_{cam}$  is the distance of camera center to the spherical scene center. The  $F_1$  score is the harmonic mean of *accuracy* and *completeness*, *i.e.*,  $F_1 = 2(r \cdot p) / (r + p)$ , which reflects the overall reconstruction quality. Three state-of-the-art combinations of dense and surface reconstruction methods are compared with our method (*Ours-Curve*), namely

- *PMVS*: dense reconstruction of PMVS [15] and Poisson surface reconstruction [22];
- *MVE*: dense reconstruction of [17] and surface reconstruction of FSSR [13], implemented by MVE [14];
- *Vu*: dense reconstruction of plane sweeping [16] and visibility-consistent surface reconstruction [48].

Because our method uses the same *plane sweeping* algorithm as *Vu* to generate MVS points, method *Vu* can be seen as our baseline. During the experiment, except for manual adjustment of the proper *octree depth* and *trim* parameter in Poisson surface reconstruction, all parameters are set to default, provided in their literatures or source codes.

The quantitative results are compared in Table 2. Among six datasets, *PMVS* and *MVE* are overall less competitive than other methods. We believe this is because they use implicit function methods to compute mesh surface (*i.e.*, Poisson [22] and FSSR [13]). These methods are prone to over-inflated meshes, and require an extra trimming algorithm to clean up the unreliable part, which is fragile and sensitive to its parameters. Other than that, *Vu* and *Ours-Curve* show competitive results. While *Vu* shows higher accuracy, *Ours-Curve* overall gives better completeness and  $F_1$  score. The qualitative comparison in Fig. 6 shows that *Vu* fails to reconstruct complete thin structures, resulting in fragmentary parts, whereas our method produces very complete reconstruction of thin structures.

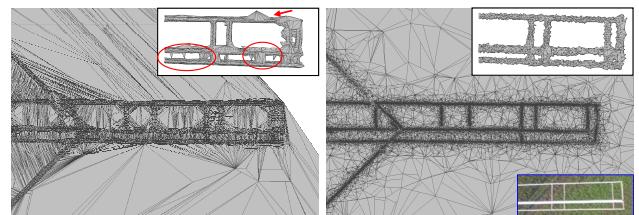


Figure 7: The comparison of tetrahedra, extracted surface meshes between (left: *Ours-Sample*) naive point sampling in curve confidence regions and (right: *Ours-Curve*) using our curve-conformed Delaunay refinement. Data: *windmill*.

**Component analysis.** To validate the effectiveness of our 3D curve optimization (Section 2) and the curve-conformed Delaunay refinement (Section 3.2), two other different settings of our method are compared. *Ours-Line* is reconstruction without enabling the curve optimization and extension

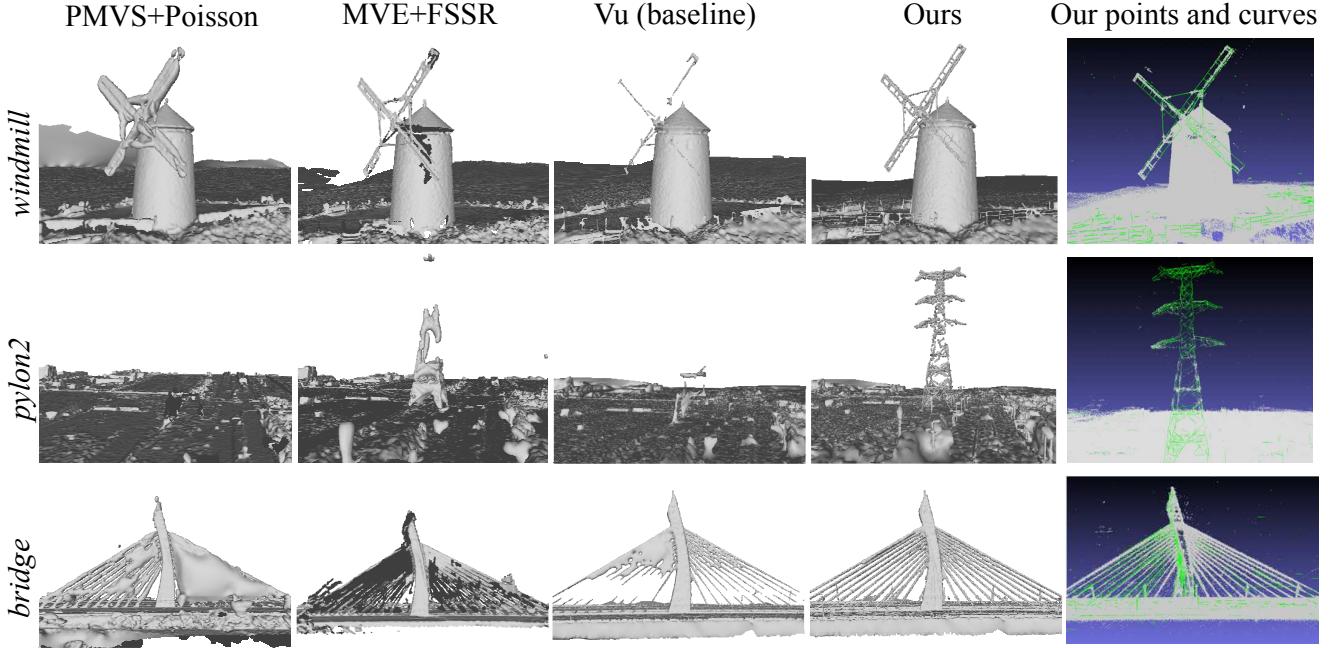


Figure 8: Qualitative comparisons of *windmill*, *pylon2* and *bridge*. Our intermediate points and curves (in green) are shown in the rightmost column. Please refer to supplementary materials for more results. **Best viewed on screen.**

steps (*i.e.*, directly using 3D line segments as curves and pass them to our surface reconstruction). *Ours-Sample* is to replace the curve-conformed Delaunay refinement step with uniformly sampled points inside the confidence region of 3D curves. These points are then mixed with MVS points to construct the tetrahedra for surface reconstruction. Table 2 shows that for all six datasets, *Ours-Curve* consistently outperforms *Ours-Line* and *Ours-Sample* in terms of accuracy and completeness, which validates the importance of both proposed components.

Fig. 7 shows the comparison of tetrahedra and meshes between *Ours-Curve* and *Ours-Sample*. The tetrahedra of *Ours-Sample* are badly-shaped. The elongated tetrahedra would increase the possibility of wrong connections between two separate parts that close to each other, and thus produce wrong topology of extracted mesh. In contrast, *Ours-Curve* can robustly yield the correct topology thanks to more structured and even distribution of tetrahedra.

#### 4.2. Qualitative evaluations on real-world dataset

Our method is also evaluated on three real-world datasets, namely *windmill*, *pylon2*, *bridge*. Some qualitative results are shown in Fig. 8. We first comment on two implicit surface function methods: *PMVS+Poisson* generates incorrect over-dilated surfaces, because the Poisson function [22] incorrectly interpolates the surface geometry inferred from noisy or incomplete dense points of thin structures. Likewise, *MVE+FSSR* also suffers from this problem, but it uses an extra clean-up step to remove the unreliable parts of

mesh. This however suffers from the sensitive clean-up parameter and would produce non-water-tight meshes. The Delaunay methods generate more neat meshes, but *Vu* suffers from incompleteness problem at thin objects. Overall, our method significantly outperforms other methods and successfully reconstructs the super thin structures such as windmill blades, wires and cables of the bridge. The meshes of them are all water-tight and manifold. For more results, please refer to supplementary materials.

#### 5. Conclusion

We have proposed the manifold surface reconstruction using spatial curves in this paper. The curves show strong abilities to enforce thin structures in the scene, which is difficult to be reconstructed by traditional point-based methods. The Delaunay tetrahedra framework adaptively and robustly fuses the point cloud with curves. Evaluations on synthetic and real-world datasets have shown the introduced curve can effectively improve the result both quantitatively and qualitatively.

The examples of thin structures shown in the paper (*e.g.*, pylon, wires) are used to be reconstructed by high-accuracy laser scans. Our method shows the potential of photogrammetry methods to evolve previous applications.

**Acknowledgement.** This work is supported by Hong Kong RGC 16208614, T22-603/15N, Hong Kong ITC P-SKL12EG02, and China 973 program, 2012CB316300.

## References

- [1] S. Aroudj, P. Seemann, F. Langguth, S. Guthe, and M. Goesele. Visibility-consistent thin surface reconstruction using multi-scale kernels. *ACM Transactions on Graphics (TOG)*, 36(6):187, 2017.
- [2] M. Berger, A. Tagliasacchi, L. Seversky, P. Alliez, J. Levine, A. Sharf, and C. Silva. State of the art in surface reconstruction from point clouds. In *EUROGRAPHICS star reports*, volume 1, pages 161–185, 2014.
- [3] Blender Online Community. Blender - a 3d modelling and rendering package.
- [4] Blend swap. <https://www.blendswap.com>.
- [5] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.
- [6] CGAL, Computational Geometry Algorithms Library. <http://www.cgal.org>.
- [7] R. Cipolla and A. Blake. Surface shape from the deformation of apparent contours. *International journal of computer vision*, 9(2):83–112, 1992.
- [8] D. Cremers and K. Kolev. Multiview stereo and silhouette consistency via convex functionals over convex domains. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(6):1161–1174, 2011.
- [9] A. Elqursh and A. Elgammal. Line-based relative pose estimation. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3049–3056. IEEE, 2011.
- [10] R. Fabbri and B. Kimia. 3d curve sketch: Flexible curve-based stereo reconstruction and calibration. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1538–1545. IEEE, 2010.
- [11] R. Fabbri and B. B. Kimia. Multiview differential geometry of curves. *International Journal of Computer Vision*, 120(3):324–346, 2016.
- [12] W. Förstner. Uncertainty and projective geometry. In *Handbook of Geometric Computing*, pages 493–534. Springer, 2005.
- [13] S. Fuhrmann and M. Goesele. Floating scale surface reconstruction. *ACM Transactions on Graphics (TOG)*, 33(4):46, 2014.
- [14] S. Fuhrmann, F. Langguth, and M. Goesele. Mve-a multi-view reconstruction environment. In *GCH*, pages 11–18, 2014.
- [15] Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, 2010.
- [16] D. Gallup, J.-M. Frahm, P. Mordohai, Q. Yang, and M. Pollefeys. Real-time plane-sweeping stereo with multiple sweeping directions. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2007.
- [17] M. Goesele, N. Snavely, B. Curless, H. Hoppe, and S. M. Seitz. Multi-view stereo for community photo collections. In *International Conference on Computer Vision (ICCV)*, pages 1–8. IEEE, 2007.
- [18] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.
- [19] M. Hofer, M. Maurer, and H. Bischof. Efficient 3d scene abstraction using line segments. *Computer Vision and Image Understanding*, 157:167–178, 2017.
- [20] H. Huang, S. Wu, D. Cohen-Or, M. Gong, H. Zhang, G. Li, and B. Chen. L1-medial skeleton of point cloud. *ACM Transactions on Graphics (TOG)*, 32(4):65–1, 2013.
- [21] M. Jancosek and T. Pajdla. Multi-view reconstruction preserving weakly-supported surfaces. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3121–3128. IEEE, 2011.
- [22] M. Kazhdan and H. Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (TOG)*, 32(3):29, 2013.
- [23] K. Kolev and D. Cremers. Integration of multiview stereo and silhouettes via convex functionals on convex domains. pages 752–765, 2008.
- [24] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. *International journal of computer vision*, 38(3):199–218, 2000.
- [25] P. Labatut, J.-P. Pons, and R. Keriven. Robust and efficient surface reconstruction from range data. In *Computer graphics forum*, volume 28, pages 2275–2290. Wiley Online Library, 2009.
- [26] C. L. Lawson. Properties of n-dimensional triangulations. *Computer Aided Geometric Design*, 3(4):231–246, 1986.
- [27] G. Li, L. Liu, H. Zheng, and N. J. Mitra. Analysis, reconstruction and manipulation using arterial snakes. *ACM Transactions on Graphics (TOG)*, 2010.
- [28] L. Liu, D. Ceylan, C. Lin, W. Wang, and N. J. Mitra. Image-based reconstruction of wire art. *ACM Transactions on Graphics (TOG)*, 36(4):63, 2017.
- [29] S. Liu and D. B. Cooper. A complete statistical inverse ray tracing approach to multi-view stereo. In *Computer Vision and Pattern Recognition (CVPR)*, pages 913–920. IEEE, 2011.
- [30] D. Meagher. Geometric modeling using octree encoding. *Computer graphics and image processing*, 19(2):129–147, 1982.
- [31] C. Mostegel, R. Prettenthaler, F. Fraundorfer, and H. Bischof. Scalable surface reconstruction from point clouds with extreme scale and density diversity. *arXiv preprint arXiv:1705.00949*, 2017.
- [32] I. Nurudinova and A. Fitzgibbon. Towards pointless structure from motion: 3d reconstruction and camera parameters from general 3d curves. In *International Conference on Computer Vision (ICCV)*, pages 2363–2371. IEEE, 2015.
- [33] M. R. Oswald, J. Stühmer, and D. Cremers. Generalized connectivity constraints for spatio-temporal 3d reconstruction. In *European Conference on Computer Vision (ECCV)*, pages 32–46. Springer, 2014.
- [34] J. Ruppert. A delaunay refinement algorithm for quality 2-dimensional mesh generation. *Journal of algorithms*, 18(3):548–585, 1995.
- [35] N. Savinov, C. Hane, L. Ladicky, and M. Pollefeys. Semantic 3d reconstruction with continuous regularization and ray potentials using a visibility consistency constraint. In *Computer Vision and Pattern Recognition (CVPR)*, pages 5460–5469. IEEE, 2016.

- [36] T. Schöps, J. L. Schönberger, S. Galliani, T. Sattler, K. Schindler, M. Pollefeys, and A. Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017.
- [37] Q. Shan, B. Curless, Y. Furukawa, C. Hernandez, and S. M. Seitz. Occluding contours for multi-view stereo. In *Computer Vision and Pattern Recognition (CVPR)*, pages 4002–4009. IEEE, 2014.
- [38] J. R. Shewchuk. Delaunay refinement mesh generation. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA SCHOOL OF COMPUTER SCIENCE, 1997.
- [39] H. Si. Tetgen, a delaunay-based quality tetrahedral mesh generator. *ACM Transactions on Mathematical Software (TOMS)*, 41(2):11, 2015.
- [40] C. Strecha, W. Von Hansen, L. Van Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008.
- [41] G. Taubin. A signal processing approach to fair surface design. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 351–358. ACM, 1995.
- [42] C. J. Taylor and D. J. Kriegman. Structure and motion from line segments in multiple images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(11):1021–1032, 1995.
- [43] D. Teney and J. Piater. Sampling-based multiview reconstruction without correspondences for 3d edges. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, pages 160–167. IEEE, 2012.
- [44] C. Topal and C. Akinlar. Edge drawing: a combined real-time edge and segment detector. *Journal of Visual Communication and Image Representation*, 23(6):862–872, 2012.
- [45] B. Ummenhofer and T. Brox. Point-based 3d reconstruction of thin objects. In *International Conference on Computer Vision (ICCV)*, pages 969–976. IEEE, 2013.
- [46] A. Usumezbas, R. Fabbri, and B. B. Kimia. From multiview image curves to 3d drawings. In *European Conference on Computer Vision (ECCV)*, pages 70–87. Springer, 2016.
- [47] A. Usumezbas, R. Fabbri, and B. B. Kimia. The surfacing of multiview 3d drawings via lofting and occlusion reasoning. *arXiv preprint arXiv:1707.03946*, 2017.
- [48] H.-H. Vu, P. Labatut, J.-P. Pons, and R. Keriven. High accuracy and visibility-consistent dense multiview stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(5):889–901, 2012.
- [49] M. Wan, Y. Wang, and D. Wang. Variational surface reconstruction based on delaunay triangulation and graph cut. *International journal for numerical methods in engineering*, 85(2):206–229, 2011.
- [50] K. Yücer, A. Sorkine-Hornung, O. Wang, and O. Sorkine-Hornung. Efficient 3d object segmentation from densely sampled light fields with applications to 3d reconstruction. *ACM Transactions on Graphics (TOG)*, 35(3):22, 2016.