

Visibility of Noisy Point Cloud Data

Ravish Mehra^{1,2}
¹ IIT Delhi

Pushkar Tripathi^{1,3}
² UNC ³ GaTech

Alla Sheffer⁴
⁴ UBC

Niloy J. Mitra^{1,5}
⁵ KAUST

Abstract—We present a robust algorithm for estimating visibility from a given viewpoint for a point set containing concavities, non-uniformly spaced samples, and possibly corrupted with noise. Instead of performing an explicit surface reconstruction for the points set, visibility is computed based on a construction involving convex hull in a dual space, an idea inspired by the work of Katz et al. [26]. We derive theoretical bounds on the behavior of the method in the presence of noise and concavities, and use the derivations to develop a robust visibility estimation algorithm. In addition, computing visibility from a set of adaptively placed viewpoints allows us to generate locally consistent partial reconstructions. Using a graph based approximation algorithm we couple such reconstructions to extract globally consistent reconstructions. We test our method on a variety of 2D and 3D point sets of varying complexity and noise content.

1. INTRODUCTION

Unorganized point cloud data (PCD) is the natural output of many 3D scanning systems. Despite its simplicity, recent research [1], [33], [22] demonstrates that PCD can be an effective and powerful shape representation, suitable for model editing and manipulation. The simplicity of PCD data-structures, the easy availability of 3D scanners as its source, and the promise of generalization to higher dimensions have all contributed to the popularity of this representation.

Unlike with other surface representations, the notion of visibility for a point set is ambiguous. However, once we have a well defined surface based on a point set, we can uniquely define the notion of visibility and identify its hidden points, from any viewpoint. The problem of reconstructing a (smooth) surface from a noisy point set, under moderate sampling requirements, has been extensively studied by the computer graphics [23], [27] and by the computational geometry community (see recent monograph [11]).

Recently Katz et al. [26] introduced the hidden point removal (HPR) operator (see Figure 1), a simple and elegant algorithm for determining point set visibility *without* explicitly reconstructing an underlying surface. The operator uses the convex hull of a point set obtained by mapping the original PCD to a dual domain to estimate the visible points in the input point set. This allows meaningful visibility computation from a desired viewpoint without the need to explicitly reconstruct a surface.

Given an input point-set, the original HPR operator determines the set of visible points from a specified viewpoint in two steps: First, the input PCD is inverted using a suitable function about the given viewpoint. Next, the convex hull of the inverted points along with the viewpoint is computed, and the points lying on the convex hull are marked as visible (see Figure 1). Thus, even though visibility for point sets is defined using a reconstructed surface definition, the HPR operator allows to estimate this visibility while operating directly on the points. Unfortunately, the simple operator performs poorly in presence of noise and fails to handle regions involving high curvature (see Figures 2 and 5, respectively).

We observe that such problems arise as slight input perturbations can result in significant changes in the structure of the corresponding convex hull. As a result, several points can be labeled incorrectly. However, in such cases the corresponding inverted points stay *close* to the new convex hull instead of being exactly on it. We quantify this observation by analyzing the effect of noise on the inversion and bound the deviation from the convex hull. We provide theoretical bounds on the stability of the HPR operator and identify *guard bands* around given point clouds, which are regions in space from where visibility cannot be reliably estimated using the algorithm. By suitably relaxing the condition of points lying on the convex hull to include points near the convex hull, we arrive at a robust visibility operator.

Using this understanding, we propose simple algorithms for consistent curve and surface reconstructions. The ability to reliably extract local connectivity information, enables the direct application of various geometry processing tools on the input PCD. Piecing together local connectivity inferred from various adaptively placed

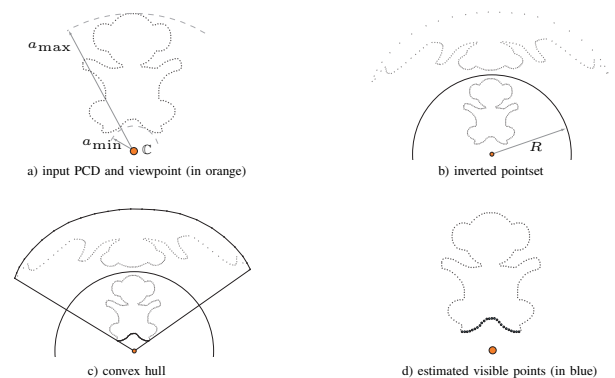


Figure 1. Stages of the basic HPR operator [26]. Figures have been suitably scaled for illustration.

viewpoints we build a connectivity graph. A special sub-graph of this graph implicitly encodes a reconstruction for the input PCD. Since extracting this sub-graph turns out to be an instance of the maximal weight-cycle problem, known to be NP hard, we propose an approximate solution. Besides curve or surface reconstructions, the connectivity graphs can also be used for smoothing the noisy input point set.

Contributions. We analyze the effect of noise in the inverted domain and obtain a bound for the distortion of the convex hull for HPR operator. We introduce the concept of *guard bands* around the point set from where visibility cannot be reliably estimated. This understanding leads to a robust HPR operator, for 2D and 3D point sets, that is used to infer local connectivity, which is subsequently collated using a graph based approximation algorithm to extract a consistent manifold.

2. BACKGROUND

Given a polygonal model, the problem of correctly and efficiently identifying the hidden faces or determining the visible parts of a model from a specified viewpoint has received significant attention since the early days of computer graphics [5], [7]. The problem is essentially an instance of sorting the model primitives according to their depth values [36]. Given the importance of the problem, hardware solutions like z-buffer [8], [35] have been proposed and widely deployed. The equivalent question of detecting visibility for a point set is ill-posed since the likelihood of a point being exactly occluded by other points is negligible. However, since a point set typically corresponds to an underlying surface, one can first reconstruct this surface, identify the visible part from the specified viewpoint, and then mark points as visible if they lie on the visible surface parts.

Surface reconstruction from unorganized points is a fundamental problem in geometry processing with applications in CAD, computer graphics and geometric modeling [11]. It is complicated by the fact that the input point sets typically contain noise and are non-uniformly sampled. Several algorithms have been proposed under different assumptions on the input data set [3], [14], [29], [27]. Notably surface fitting using moving least square (MLS) surfaces [28] and its variants [4], [17], [24] are popular for their simplicity and effectiveness.

An alternate approach, proposed by Katz et al. [26], estimates visibility of point sets directly *without* explicit surface reconstruction. Given a point cloud $\mathbf{P}$ and a camera viewpoint $\mathbb{C}$, the hidden point removal (HPR) operator determines the set of points that are visible from $\mathbb{C}$ by a two step process of inversion and convex hull computation (see Figure 1). For simplicity, assume that $\mathbb{C}$ is at the origin. First, each point $\mathbf{p}_i \in \mathbf{P}$ is inverted about the origin, i.e., the viewpoint, using an inversion function $f(\mathbf{p}_i)$, which monotonically decreases with $\|\mathbf{p}_i\|$ (where $\|\cdot\|$ is a norm). This results in points closer to the viewpoint becoming far and vice-versa. Typical inversion functions include spherical and exponential, which for any point $\mathbf{p}_i$, are defined as:

- Spherical inversion

$$\hat{\mathbf{p}}_i := f(\mathbf{p}_i) = \mathbf{p}_i + 2(R - \|\mathbf{p}_i\|)\mathbf{p}_i/\|\mathbf{p}_i\| \quad (1)$$

where, R is the radius of inversion.

- Exponential inversion

$$\hat{\mathbf{p}}_i := f(\mathbf{p}_i) = \mathbf{p}_i/\|\mathbf{p}_i\|^\gamma \quad (2)$$

where, $\gamma > 1$ is the inversion parameter and $\|\mathbf{p}_i\| < 1$.

Let $\hat{\mathbf{P}}$ denote the inverted point cloud. In the second step, we take the convex hull of $\hat{\mathbf{P}} \cup \mathbb{C}$. All the points of the original point set that get mapped onto the convex hull constitute the set of points visible from $\mathbb{C}$. For a point cloud with n points, it has an asymptotic complexity of $O(n \log n)$. Radius of inversion is found by maximizing the number of distinct points visible from $\mathbb{C}$ and reflection of $\mathbb{C}$ about center of mass of the PCD. For the rest of the paper we refer to this radius as R_{opt} .

An important drawback of this operator is its susceptibility to noise (Figures 2 and 3). The number of *false negatives*, i.e., visible points that are declared as hidden, for a radius R quickly increase as the noise magnitude gets larger (Figure 3, column 2). In presence of noise, the optimized radius R_{opt} proposed in [26] may converge at a much larger radius resulting in large number of *false positives*, i.e., hidden points declared visible (Figure 3, column 3).

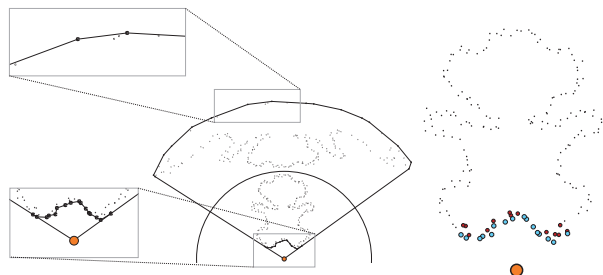


Figure 2. For noisy point cloud data, basic HPR operator can result in unreliable visibility estimates (shown in blue). Red points denote samples that should have been marked as visible.

Another drawback of the original HPR operator is the inability to consistently resolve visibility in regions of high curvature. Based on the radius of inversion and the point set, a curvature threshold exists such that all concave regions with curvature below the threshold are correctly resolved (see Lemma 4.3 in [26]). This threshold can be increased by changing the radius of inversion for spherical inversion function. Unfortunately, this increases the number of *false positives* i.e., hidden point declared visible (see Figure 5), thus restricting the amount of concavity that can be reliably handled.

3. ROBUST VISIBILITY OPERATOR

We overcome the deficiencies of the original HPR operator by introducing a robust visibility operator. Noise in the original PCD may amount to large structural perturbations of the convex hull in the inverted domain.

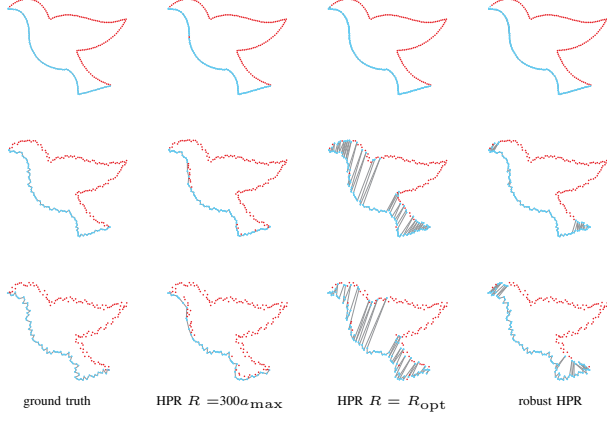


Figure 3. Visibility of noisy point cloud under varying noise amount from a fixed view point (not shown in the figure). Blue points denote the points identified as visible, while red denote the hidden ones. Rows from top down indicate points with no, 1%, and 1.7% additive uniform noise in normal direction, respectively measured in terms of length of diagonal of the bounding box. Polyline indicate recovered local connectivity. Observe the robustness of the proposed HPR operator. For comparison, we use the original connectivity and visibility information (top-left) as ground truth for the noisy cases.

As a result, some visible points may get displaced from the perturbed hull. We bound the maximum deviation of such points, and define a thin band around the convex hull which contains these points. This helps us specify the robust HPR operator.

Let $\mathbf{P}$ denote the original noise free point set. Let $\mathbf{P}^\sigma := \{\mathbf{p}_i + \sigma \mathbf{n}_i | \forall \mathbf{p}_i \in \mathbf{P}\}$ be the noisy point set, where σ is a uniform random variable over the range $[0, a]$, and $\mathbf{n}_i$ is a unit vector oriented in a uniformly chosen random direction. Intuitively, we assume that in the noisy point cloud every point is perturbed to a point chosen uniformly at random from a ball of radius a centered at it (similar noise model was used by Mitra et al. [32]) and Pauly et al. [34].)

3.1 Noise Robustness

In this section we present the main results, while the corresponding proofs can be found in the Appendix. Here we consider spherical inversion, while the exponential inversion case can be found in the Appendix.

Theorem 1. *Given a noisy point cloud $\mathbf{P}^\sigma$, a viewpoint $\mathbb{C}$, the maximum noise in the inverted domain under spherical inversion function with radius of inversion R , is bounded by*

$$\epsilon_{\max} = \left(\frac{4R}{a_{\min} - \sigma} - 1 \right) \sigma \quad (3)$$

where, $a_{\min}$ is the distance from $\mathbb{C}$ to the closest point in $\mathbf{P}$.

This implies that the convex hull of $\mathbf{P}^\sigma$ will not be displaced more than $\epsilon_{\max}$ with respect to the convex hull of $\mathbf{P}$. In the worst case, it could move out by $\epsilon_{\max}$ while previously visible points could move in by $\epsilon_{\max}$. Therefore, the false negative points can be separated by a maximum of $2\epsilon_{\max}$ from the convex hull. We mark such points as visible, project them onto the convex

hull, and recover local connectivity in the process.

Given a noisy point cloud $\mathbf{P}^\sigma$, diameter of point cloud $D (= (a_{\max} - a_{\min}))$, we use projection if

$$\epsilon_{\max} \leq \alpha D / 2 \quad (4)$$

where, α is the projection parameter and $a_{\max}$ is the distance from $\mathbb{C}$ to the farthest point in $\mathbf{P}$. We use $\alpha = 0.15$ for all examples in this paper. The allowed range of R for which such a projection can be performed is given as follows.

Lemma 1. *Given a noisy point cloud $\mathbf{P}^\sigma$, projection can be applied if*

$$ma_{\max} \leq R \leq \left(\frac{\alpha D}{2\sigma} + 1 \right) \left(\frac{a_{\min} - \sigma}{4} \right). \quad (5)$$

See Appendix for meaning of variable m . Further simplification leads to the following bound on the noise margin that can be reliably handled:

$$\sigma < \frac{\alpha D}{2(4m - 1)}. \quad (6)$$

This shows that if the noise margin exceeds $\alpha D / 2(4m - 1)$, then distinguishing points of different surface parts from noisy points becomes ambiguous, and hence projection can fail. If noise is less than the threshold, we can apply projection and recover visibility along with local connectivity.

Theorem 2. *Given a noisy point cloud $\mathbf{P}^\sigma$, the minimum distance above which viewpoints can be placed $a_{\min}$ is*

$$a_{\min} \geq \frac{(4m + \frac{\alpha}{2})D + \sigma}{\left(\frac{\alpha D}{2\sigma} - (4m - 1) \right)}. \quad (7)$$

As we cannot place a viewpoint at distance closer than the threshold, this defines a *guard zone* around the noisy point cloud (see Figure 4). When $\sigma \geq \alpha D / 2(4m - 1)$, Equation 7 is no longer valid and the guard zone spans the entire space.

In summary, for determining the robust visibility of noisy point cloud $\mathbf{P}^\sigma$ from viewpoint $\mathbb{C}$, we first check the position of $\mathbb{C}$ w.r.t the *guard band*. If $\mathbb{C}$ lies outside the *guard zone*, we apply the inversion and convex hull steps of the original HPR operator to $\mathbf{P}^\sigma$ while keeping R inside the radius range (Equation 5). For $\mathbf{p}_i \in \mathbf{P}^\sigma$, if the inverted point $\hat{\mathbf{p}}_i$ is closer than $2\epsilon_{\max}$ from the



Figure 4. For a noisy point cloud (left), we define the corresponding *guard zone* (right) as the region in space from which visibility cannot be reliably estimated (see Equation 7).

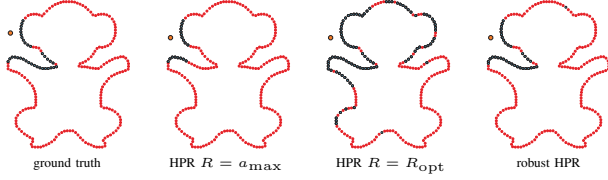


Figure 5. For concave regions, small values of radius of inversion results in missing several visible points, while large values lead to many false positives. Even manually tuned R produces unsatisfactory result. The robust HPR operator, using a weighted definition of visibility, yields results comparable with the ground truth.

convex hull, we project $\hat{\mathbf{p}}_i$ back onto the convex hull, update the convex hull connectivity and mark $\mathbf{p}_i$ as visible. This gives us robust visibility of noisy PCD $\mathbf{P}^\sigma$ from viewpoint $\mathbb{C}$. Otherwise if $\mathbb{C}$ lies inside the *guard zone*, the noise ambiguity of the data prevents us from correctly determining the visibility from this viewpoint.

3.2 Concavity Robustness

The original HPR operator correctly estimates the set of visible points for oblique planar and convex regions of the point cloud. The method can also handle limited concavity (Lemma 4.3 in [26]) with the limit or the curvature threshold being directly proportional to the radius of inversion R . In order to handle regions of higher curvature one could potentially increase R , thereby increasing the curvature threshold. However, this leads to higher *false positives* (see also Figure 7 in [26]).

We introduce a softer notion of visibility that allows the handling of high curvature regions. We draw upon the observation that convex, oblique planar and low curvature concave regions are consistently visible at lower range of R , whereas high curvature concave regions become visible only at higher R values. Keeping the viewpoint fixed, we estimate visibility using various R in the range $[a_{\max}, R_{\max}]$.

For noise-free point cloud $\mathbf{P}$, $R_{\max} = R_{\text{opt}}$ as given by the original HPR operator in [26]. In case of noisy point cloud $\mathbf{P}^\sigma$, $R_{\max} = (\frac{\alpha D}{2\sigma} + 1) (\frac{a_{\min} - \sigma}{4})$ as shown in Equation 5.

Points are weighted depending on the number of times they are tagged visible. Unlike visible points, false positives are not *persistently* estimated to be visible in any sizable interval in the radius range. Using this observation, we filter out false positives across the curve/surface (see Figure 5).

The robust visibility operator inherits convex hull connectivity from the construction step. For 2D points connectivity is in the form of edges, while in 3D it consists of triangles. Similar to weighted visibility of points, we assign weights to such connectivity edges/triangles proportional to the number of times they appears in the convex hull over different values of R . These (normalized) weights are subsequently used for curve/surface reconstruction and for noise smoothing.

4. VISIBILITY BASED RECONSTRUCTION

The notion of visibility for point clouds is defined with respect to a surface reconstructed from the point

set. Being able to estimate visibility without explicitly reconstructing the surface, we now propose how to extract a surface reconstruction stringing together visibility estimates from various viewpoints. From each viewpoint, along with visibility estimates, we get a partial reconstruction in the form of local connectivity, where a pair of points is considered neighbors if both the points are marked visible and share an edge along the convex hull constructed in the inverted domain. We begin by placing multiple viewpoints around the point cloud while staying out of its guard zone. Using the robust HPR operator we generate locally consistent partial reconstructions from each of these viewpoints. We globally couple such reconstructions using a graph theoretic formulation to extract a curve/surface reconstruction. First, we note some desirable properties of such a reconstruction:

Curve. Reconstruction of a 2D point cloud sampled from a *simple* closed curve should be a set of edges connecting the points such that they form a closed loop without self-intersections.

Surface. Reconstruction of a 3D point cloud sampled from a manifold surface should consist of triangles such that: Each edge of a triangle is shared with at most one other triangle, each triangle shares edges with at most three others, and triangles intersect only at edges.

4.1 Curve Reconstruction

Given a noisy point cloud $\mathbf{P}^\sigma$, we determine the extent of the *guard zone* surrounding it using Equation 7 and place a collection of viewpoints uniformly, say $\mathcal{C}$, outside this zone. For all our 2D experiments we used $|\mathcal{C}| = 120$. For every viewpoint $\mathbb{C} \in \mathcal{C}$, the robust HPR operator estimates a set of visible points along with local connectivity information, even in the presence of noise or concavity. For each viewpoint, each edge of the local connectivity graph is assigned a *visibility weight* proportional to the number of times the edge is visible from the viewpoint for different choices of R (see Section 3.2). Final weight of an edge is equal to the sum of the weights from each viewpoint. Union of all such edges represents the global connectivity information. We call this the *visualization* step.

The connectivity information is encoded as a graph $G = (V, E)$, where the vertex set V consists of points of $\mathbf{P}^\sigma$ and the edge set E is the union of all edges found in the visualization step. This weighted graph defines the relative importance of edges for the final reconstruction. Based on the intuition that the edges belonging to the reconstructed curve should be visible from multiple viewpoints, and thus have higher weights, we formulate the curve reconstruction problem of 2D point cloud as follows: Given a graph $G = (V, E)$ that encodes the connectivity information extracted in the visualization step, find the maximum weighted subgraph of G that forms a valid curve.

NP Hardness. Even though the formulation is intuitive, the optimization problem defined above is NP hard. Finding the *maximum weight cycle* in a weighted graph

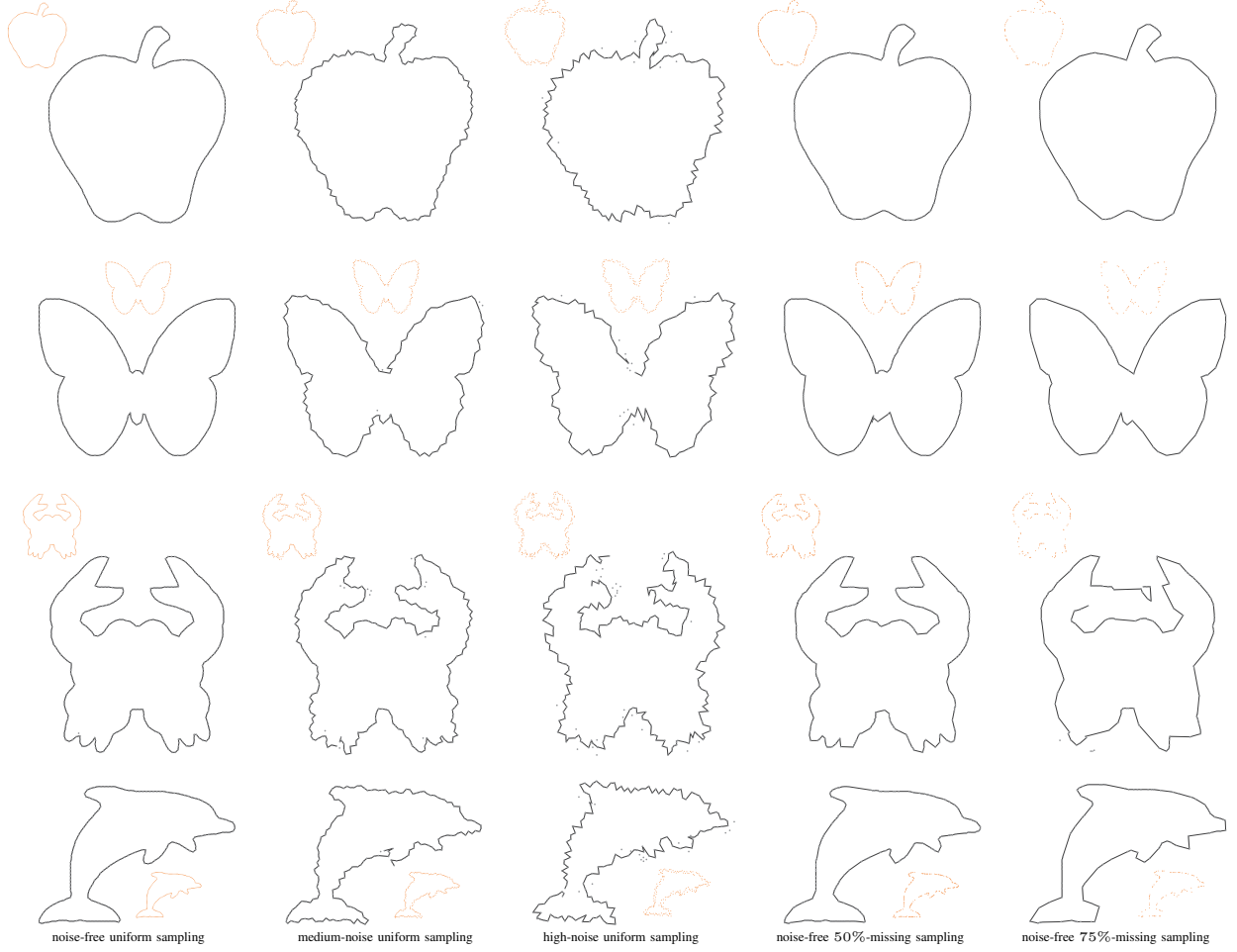


Figure 6. Curve reconstruction results for points sampled off apple, butterfly, crab, and dolphin models with different noise amounts and non-uniform sampling. For illustration, sample points are also shown in orange in corresponding insets. Medium-noise and high-noise correspond to uniform random perturbations in the normal direction respectively by 1% and 2% of the diagonal length of the original bounding boxes.

is known to be a NP hard problem [21]. The hardness results is true even for an uniformly weighted (unit weight) undirected graph [25]. We look for approximation algorithms that lead to near optimal results. Note

that although we found our algorithms, both in 2D and in 3D, to give good results experimentally, at this point we do not have any reasonable factor to bound their approximation quality.

Approximation algorithm. We propose an approximation algorithm (see Algorithm 1) to find the maximum

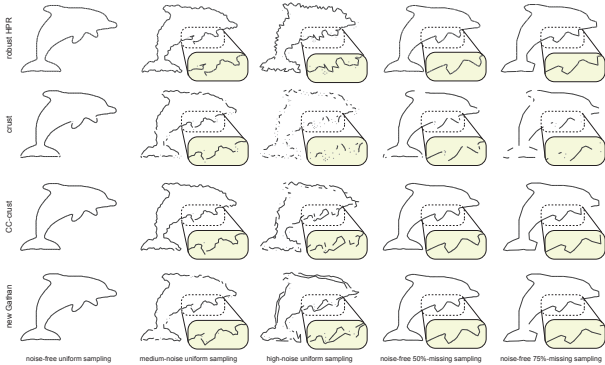


Figure 7. Reconstruction result using robust HPR, crust [2], CC-crust [16], and Gathan [19]. For noise-free uniform sampling all algorithms perform well. The crust reconstruction degenerates in presence of noise or on non-uniformly sampled point sets. On noise-free but non-uniformly sampled point sets, CC-crust, Gathan, and robust HPR give similar results. However, in presence of noise, robust HPR yields better connectivity, while CC-crust and Gathan reconstructions contain many small curves. All algorithms yield results within 1-2 seconds.

Algorithm 1 Curve Reconstruction

Input: A graph $G = (V, E)$ with set of vertices V and weighted edges E

Output: Maximum weight cycle $S \subseteq E$

- 1: $S \leftarrow \text{max_weight_edge}(E)$
 - 2: {Add heaviest edge incident on an end point}
 - 3: **repeat**
 - 4: $\mathbf{v}_l \leftarrow \text{left_most_vertex}(S)$
 - 5: $\mathbf{v}_r \leftarrow \text{right_most_vertex}(S)$
 - 6: $\mathbf{e}_{\text{left}} \leftarrow \text{max_weight_edge}(\mathbf{e} \mid \mathbf{e} \in E(\mathbf{v}_l) - S \text{ and } \text{NotIntersect}(\mathbf{e}, s) \forall s \in S)$
 - 7: $\mathbf{e}_{\text{right}} \leftarrow \text{max_weight_edge}(\mathbf{e} \mid \mathbf{e} \in E(\mathbf{v}_r) - S \text{ and } \text{NotIntersect}(\mathbf{e}, s) \forall s \in S)$
 - 8: $\mathbf{e} = \text{max_weight_edge}(\mathbf{e}_{\text{left}}, \mathbf{e}_{\text{right}})$
 - 9: $S \leftarrow S \cup \mathbf{e}$
 - 10: **until** $\mathbf{e} = \emptyset$ or $\mathbf{v}_l = \mathbf{v}_r$
-

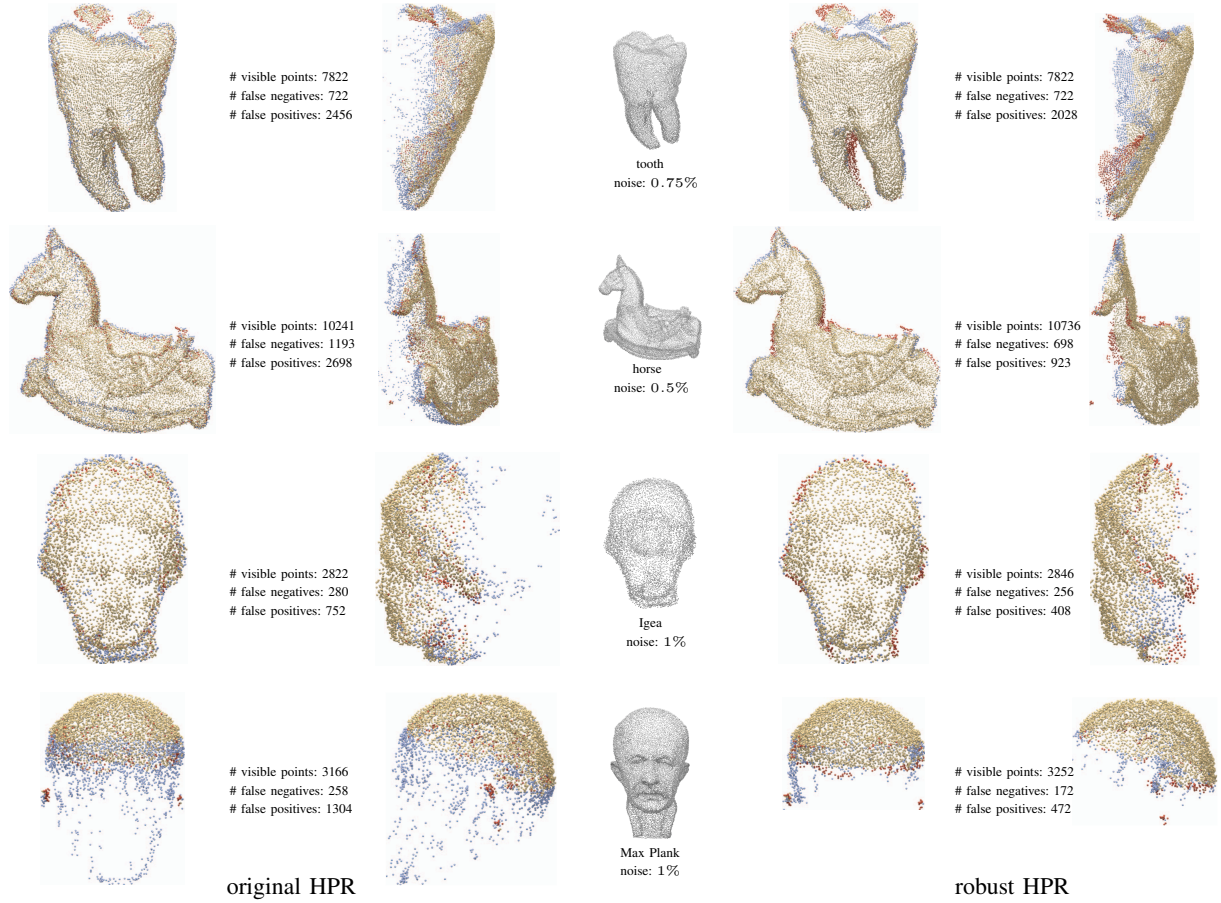


Figure 8. Comparison of visibility results by original HPR operator and robust HPR operator on point clouds from various models corrupted with uniform random perturbations in the normal direction (view points are not shown). Noise margins as indicated with the data sets are measured with respect to the diagonal lengths of the respective original bounding boxes. Yellow points denote correctly marked visible points, blue points denote those falsely marked as visible(false positives), and red points denote those which are falsely marked as invisible(false negatives). Most false points for robust HPR operator lie near the respective silhouettes. False negatives and positives are marked using ground truth computed based on a simple z-buffer visibility test using the connectivity of the (respective) original 3D models.

weight cycle in a *sparse* graph $G = (V, E)$. The central idea is to iteratively build a solution by greedily augmenting the current solution with new edges. We refer to the partial solution at the end of the i -th iteration as the set $S_i \subseteq E$. The invariant that we maintain throughout the algorithm is that edges in S_i form a path. We initialize the set S_0 with the heaviest edge of E (Algorithm 1: step 1). In the i -th iteration, the solution is augmented with the heaviest edge incident on one of the two end-points of the path S_i , which does not intersect any other edge in S_i (Algorithm 1: steps 6 to 8). This prevents the formation of cycles. The process is repeated until we form a closed curve or we run out of candidate edges. Next, we analyze its complexity.

Running Time. The number of edges in G is at

model	a_{min}	a_{max}	σ	ϵ_{max}	R_{max}
tooth	1.69	2.27	0.010	0.046	2.44
horse	1.34	1.76	0.007	0.031	1.77
Igea	2.54	3.55	0.016	0.074	3.67
Max P.	2.42	3.43	0.014	0.067	5.00

TABLE 1
ROBUST VISIBILITY OPERATOR PARAMETERS FOR THE TEST
SCENARIOS SHOWN IN FIGURE 8.

most $|V||C|$. Typically this bound is very loose and the number of edges is usually only two-three times $|V|$ since the convex hulls of neighboring viewpoints overlap considerably, sharing a lot of edges. Thus the number of edges is $O(|V|)$ with edge weights being bounded by $|C|$. Every edge that is chosen must be checked to see if it does not intersect with any previously chosen edge. The time spent in testing for edge intersections (Algorithm 1: steps 6 and 7) dominates the running time of the algorithm. For efficiency we pre-compute all edge intersections in $O(|E| \log(|V|) + k)$ time [9] where, k is the total number of intersecting edge pairs in the graph. Typically k is small leading to an asymptotic run time of $O(|V| \log(|V|))$, i.e., $O(n \log n)$.

4.2 Surface Reconstruction

A generalization of the curve reconstruction problem to 3D handles the surface reconstruction problem. The underlying idea is to greedily extend the solution using weighted atomic elements while maintaining the desired properties using a graph theoretic formulation. As in the 2D case, this guides the neighborhood search for new elements. Visibility for triangles are tracked, like

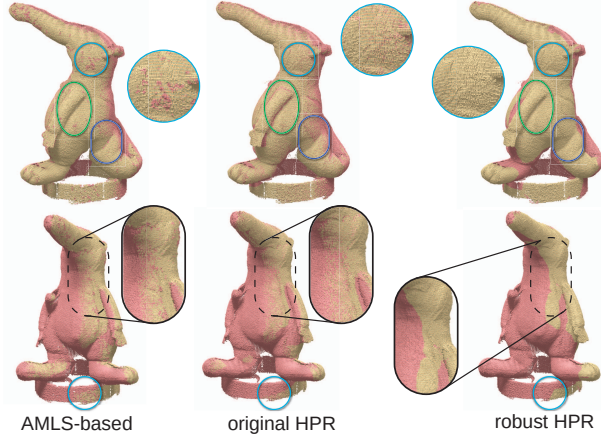


Figure 9. Typical behavior of original HPR and robust HPR on raw scanned data from a given viewpoint (not shown), with the hidden points in pink and visible points in yellow. Notice that the boundary between visible and hidden regions is sharp with robust HPR, indicating few misclassifications. In absence of ground truth, we show the AMLS [18] + cocone [12] based reconstruction and use the surface for visibility computation (left column) for comparison. Notice the result also has misclassifications. We highlight some regions where original HPR clearly produces a large number of misclassifications. The scanner noise parameter σ is estimated using a calibration plane.

edges in 2D, since triangles are the building blocks of the convex hull in 3D.

Given a noisy point set $\mathbf{P}^\sigma$, we first determine the *guard zone* and place multiple viewpoints outside it uniformly at random. We used $|\mathcal{C}| \approx 500$ in our experiments. For each viewpoint, the robust HPR operator is applied, and for each triangle in the local connectivity graph we maintain a count of the number of viewpoints from which it is visible. The intuition being that triangles that belong to an actual surface triangulation would be visible more often.

We encode the results from the visualization phase in a graph $G = (V, E)$ where the vertex set V consists of the set of all triangles encountered during the visualization phase. Two vertices are defined to be adjacent if their corresponding triangles share an edge. Additionally, a pair of vertices in V are connected by an *anti-edge* if the corresponding triangles intersect.

We define a weight function on the vertices as follows. The weight of a vertex, i.e., triangle in the original graph, consists of two components, the first of which is fixed and is proportional to the number of times it was included in a convex hull during visualization. The second contribution to the weight of a vertex is not statically determined at the start of the algorithm. It is a function of the state of the algorithm and is proportional to the number of neighbors of the vertex that are already a part of the partial solution. Thus the weight of a vertex not only depends on the visibility information but also on its connectivity to other vertices that have already been chosen, and may change during the course of the algorithm. We use a stronger notion of weight function because the surface reconstruction affords higher degrees of freedom than the curve reconstruction problem. A greedy augmentation algorithm similar to Algorithm 1, would need to choose between

triangles that share different number of edges with the triangles that have already been chosen. This is different from the possibilities during curve reconstruction where every candidate edge is incident on exactly one of the end-points of the partial solution. The weight function defined above provides a single measure of the utility of a triangle both from connectivity and visibility standpoints.

We wish to find the maximum weight subset of V such that no two vertices in the induced subgraph are connected by anti-edges (to prevent self-intersection) and the degree of every vertex in the induced subgraph is exactly three (each triangle shares an edge with three adjacent triangles).

NP Hardness. Even if we ignore the anti-edges the given problem is NP hard [20]. So we seek an approximate algorithm (see Algorithm 2) to this problem.

Approximation algorithm. As in 2D, we greedily extend the solution set, while maintaining the desired surface properties. Note that unlike the 2D case, here the solution set refers to a subset of vertices in G . Another departure from the curve reconstruction is that while earlier we initialized the algorithm by a single seed (heaviest edge), here we use multiple seeds in the *initialization phase* of the algorithm to account for varying sampling densities across different parts of the

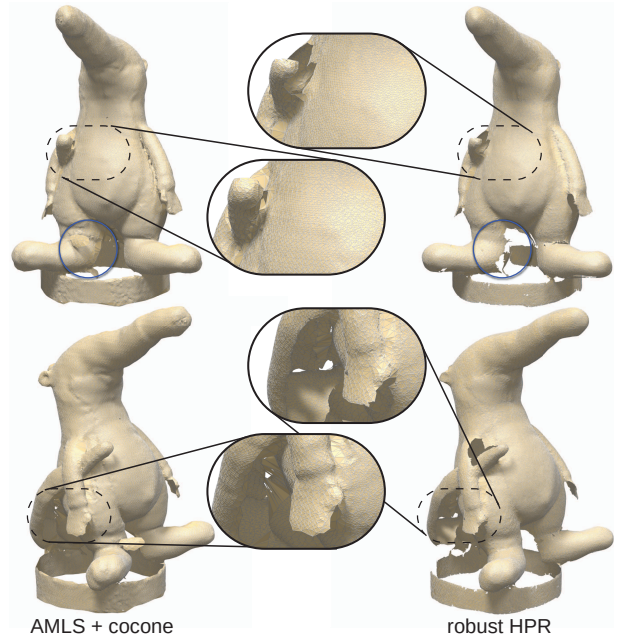


Figure 10. Reconstruction outputs on raw scanned data for the coati model using a combination of AMLS [18] and cocone [12] (left), and robust HPR based smoothing (Section 4.4) with reconstruction (Algorithm 2). While AMLS-cocone generate large triangles in poorly sampled regions our approach provides a more natural result leaving such regions empty. Similarly, by construction, our method avoids self intersections present in AMLS-cocone based reconstruction. The raw scan is *thick* due to noise and a direct reconstruction is unnatural. Hence instead of an interpolatory solution, we choose approximation as with AMLS method. For smoothing the point cloud for robust HPR, we use connectivity based on robust HPR (see also Figure 14). On such noisy scans original HPR based quick-and-dirty reconstruction breaks down, since visibility and convex-hull based connectivity estimates from single views have significant errors (see Figure 9).

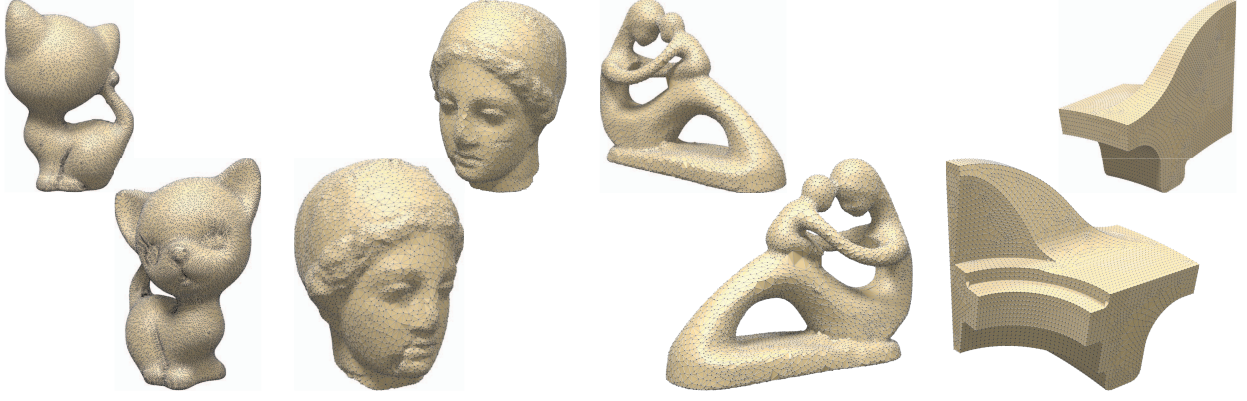


Figure 11. Surface reconstruction results on various models using Algorithm 2. (Left to right) A well-sampled kitten model, Igea face model with varying levels of detail, mother-and-child model having narrow features and large holes, fandisk model containing sharp feature curves and edges. Note that while state-of-the-art surface reconstruction methods can handle such inputs equally well or better, a visibility based surface reconstruction method is interesting as it links two apparently different problems in computer graphics and computational geometry.

object. We start with an independent set of vertices in G that have a weight greater than a preset threshold as seed points (Algorithm 2: steps 4 to 7). In the next phase, the algorithm proceeds by iteratively adding vertices to extend the surface while ensuring that no pair of vertices in G have an anti-edge between them and that no vertex has degree more than three in the induced subgraph. The order in which vertices are added is determined by their weight, with heavier vertices getting higher priority. Recall that the weight of a vertex is not fixed but is a dynamic function of the state of the algorithm. We refer to this as the *greedy selection* phase (steps 9 to 13). The algorithm terminates when more vertices cannot be added. A formal description of the algorithm is given below.

Running Time: Like in 2D, the total number of can-

Algorithm 2 Surface Reconstruction

Input: A dual graph $G = (V, E)$ with set of weighted vertices V and edges E

Output: Maximum weight triangulation $S \subseteq V$

```

1: {Initialization: Build independent set}
2:  $S \leftarrow \phi$ 
3: for all  $v \in V$  do
4:   if  $W(v) \geq \text{THRESHOLD}$  and notAdjacent( $u, v$ )
     and noAntiEdge( $u, v$ )  $\forall u \in S$  then
5:      $S \leftarrow S \cup v$ 
6:      $V \leftarrow V - v$ 
7: {Greedy Selection: Add vertices greedily}
8:  $v \leftarrow \text{max\_weight\_vertex}(w \mid w \in V, \exists s \in S \text{ s.t.}$ 
   Adjacent( $w, s$ ))
9: while  $v \neq \text{NULL}$  do
10:  if noAntiEdge( $u, v$ ) and degree( $u, S \cup v$ )  $\leq 3$ 
     $\forall u \in S$  then
11:     $S \leftarrow S \cup v$ 
12:     $V \leftarrow V - v$ 
13:  for all ( $u \mid u \in V$  and Adjacent( $u, v$ )) do
14:    update_weight( $u$ )
15:   $v \leftarrow \text{max\_weight\_vertex}(w \mid w \in V, \exists s \in S \text{ s.t.}$ 
    Adjacent( $w, s$ ))
```

didate vertices after the visualization stage with $|C|$ viewpoints is at most $|V||C|$. The running time consists of the time spent in maintaining the priority queue, which can be done in $O(|V||C| \log |V||C|)$ time using a heap. Since the weight of any vertex is updated at most twice and the total number of vertices is bounded by $|V||C|$, there are at most $2|V||C|$ operations on the heap. Another component of the running time deals with enumerating the set of anti-edges, i.e., to check for intersection between triangles corresponding to the end points of the edge. Since checking for intersection of two triangles is an expensive operation, though constant time, we avoid it wherever possible. Using the observation that two triangles whose circumcenters are *reasonably* far do not intersect, we only check for triangle pairs whose circumcenters are *close*. By calculating the circumcenters of all triangles (vertices in G), and using efficient space partitioning data structure techniques [6] we make efficient enumeration of the anti-edges. In our experiments we have observed that every triangle intersects with few other triangles thus the number of anti-edges is reasonably small (typically fewer than 10 per triangle in our experiments). Hence the total running time is dominated by that spent in managing the priority queue which is asymptotically $O(|V||C| \log(|V||C|))$, i.e., $O(n \log n)$ for a constant number of viewpoints.

4.3 Improvements

- **Filtering after Visualization:** Triangles with long edges can significantly slow down the algorithm, since they can intersect with a large number of edges. As a solution we filter out such aberrations after the visualization step by removing triangles with edges more than $5\times$ than the average sampling distance s_{avg} . Furthermore, we normalize the weight of every edge e by multiplying it with $\exp(-2 \cdot ||e|| - s_{\text{avg}}|/s_{\text{avg}})$. Thus, triangles with too long or too short edges get lower priority. Similar edge filtering is less critical for 2D curve reconstruction, where the algorithm is fast to start

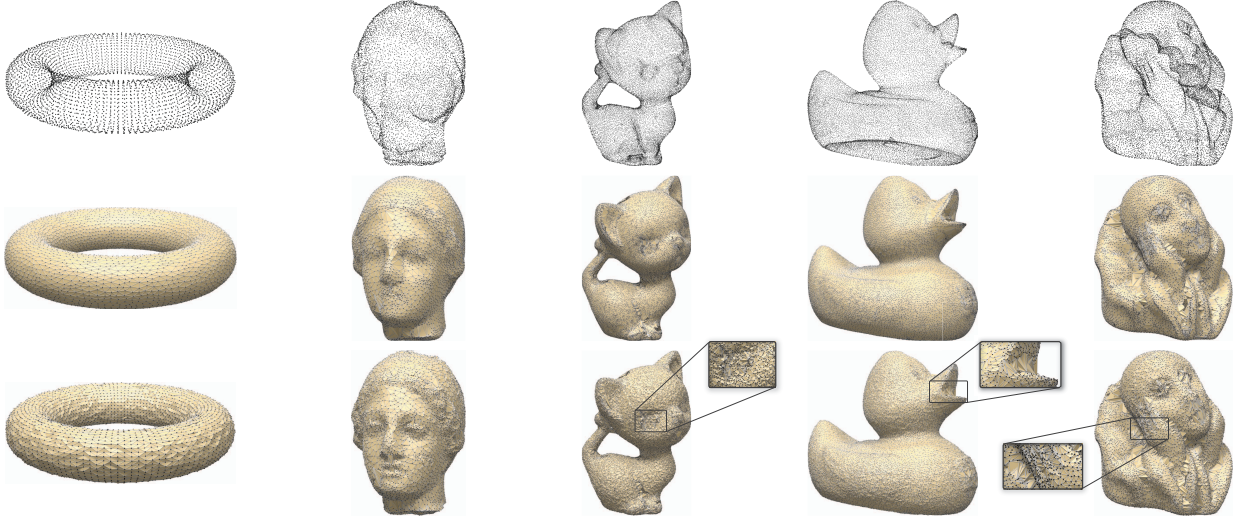


Figure 12. (Top row) Input noisy point sets are smoothed and reconstructed using the robust HPR operator on torus, Igea, kitten, duck and Pierrot models, respectively. Connectivity information from the smoothed reconstructions (middle row) are mapped back to the original noisy data sets (bottom row). Insets highlight artifacts due to poor sampling or concavity.

with.

- **Multi-iteration Reconstruction:** In the reconstruction step, if the placed viewpoints do not cover the entire point cloud, certain sections can be poorly reconstructed. Also placing suitable viewpoints is challenging since we do not know how the underlying curve/surface looks like. As a solution, we use multiple iterations of the algorithm. In each iteration, missing regions in the curve/surface are located and viewpoints placed adaptively around such regions. We decrease the number of additional viewpoints exponentially by a factor of two resulting in $\log |\mathcal{C}|$ iterations. Small holes are fixed using diffusion based hole filling [10].

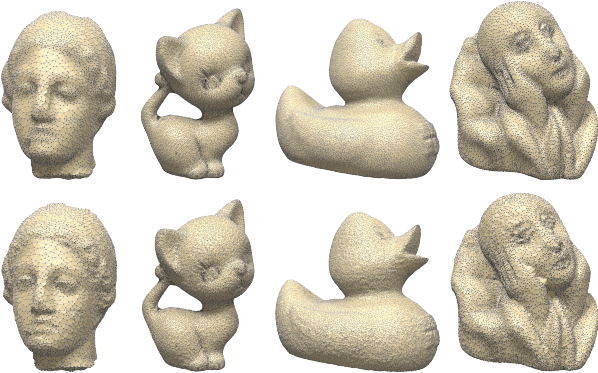


Figure 13. (Top row) AMLS smoothing [18] followed by tight cocone reconstruction [13] (compare with Figure 12-middle row); (bottom row) reconstruction results with robust cocone [15] (compare with Figure 12-bottom row). Robust HPR based reconstruction produces results of similar quality to these state-of-the-art methods on uniformly sampled data (all these methods may exhibit artifacts in areas of high-curvature). On non-uniformly sampled data robust HPR produces more desirable results as shown in Figure 10. Note that in presence of noise, the original HPR based method is not applicable.

4.4 Noise-smoothing

Since we start with noisy point cloud $\mathbf{P}^\sigma$, our reconstruction algorithm can output a rough and non-uniform surface. However, the global connectivity information E allows us to apply Laplacian smoothing without explicitly reconstructing the surface. For each $\mathbf{p}_i \in \mathbf{P}^\sigma$, we determine the set of its neighbors using connectivity E . We then calculate the new position of $\mathbf{p}_i$ by applying HC Laplacian smoothing [30] as follows :

$$\mathbf{l}_i = (1 - \rho)\mathbf{p}_i + \rho \left(\frac{\sum_{j \in \text{Adj}(i)} w_j^i \mathbf{p}_j}{\sum_{j \in \text{Adj}(i)} w_j^i} \right) \quad (8)$$

where w_j^i = weight of edge connecting $\mathbf{p}_i$ and $\mathbf{p}_j$ and $\rho = 0.1$ is the rate of Laplacian smoothing.

We denote the points in the previous iteration by $\mathbf{q}_i$ and the original points by $\mathbf{o}_i$.

$$\begin{aligned} \mathbf{b}_i &= \mathbf{l}_i - (\alpha \mathbf{o}_i + (1 - \alpha) \mathbf{q}_i) \\ \mathbf{d}_i &= -\beta \mathbf{b}_i - \frac{1 - \beta}{|\text{Adj}(i)|} \sum_{j \in \text{Adj}(i)} \mathbf{b}_j \\ \mathbf{p}_i^{\text{new}} &= \mathbf{p}_i + \mathbf{d}_i \end{aligned}$$

where $\alpha = 1$ and $\beta = 0.6$.

The resulting smoothed point cloud, after 80 to 100 iterations, is used as input for the next step. We apply alternate steps of visualization and smoothing, typically 4 to 5, and return the smoothed point cloud as output (see Figure 14). Alternating between visualization and

model	# points	noise margin	time (in msec)
tooth	21.9k	0.75 %	531
horse	32.3k	0.50 %	681
Igea	8.3k	1.00 %	171
Max Planck	20.0k	1.00 %	375

TABLE 2
ROBUST VISIBILITY OPERATOR TIMINGS FOR THE TEST SCENARIOS SHOWN IN FIGURE 8.

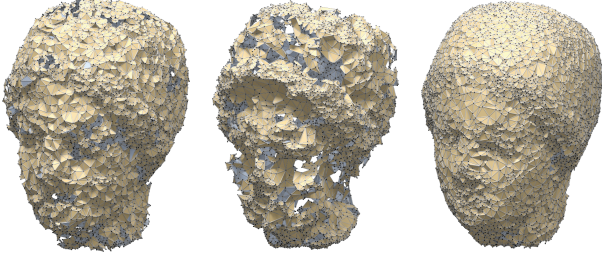


Figure 14. Reconstruction results for point set from Igea model corrupted with uniform noise in normal direction of magnitude 1% of bounding box diagonal length. (Left to right) Surface reconstruction of original point set, of point set obtained by smoothing the corrupted point set using connectivity obtained from original HPR operator, and of point set obtained by smoothing the corrupted point set using connectivity obtained from proposed robust HPR operator. Both in the case of original HPR and robust HPR, we avoid pruning of long edges in the connectivity graphs based on thresholds and heuristics. Due to long incorrect connections, the middle model degrades due to ‘smoothing’. Interior or back facing triangles are in blue.

smoothing results in improved connectivity estimates for $G(V, E)$ resulting in improved smoothing.

5. RESULTS

We tested our proposed robust HPR operator for visibility and reconstruction of 2D points. Figure 3 compares our robust HPR operator with the original HPR operator [26]. Our operator detects the correct set of visible points and has fewer false positives for noisy point sets. By placing multiple viewpoints around the PCD, we generate consistent partial reconstructions that are coupled using a graph based approach (Algorithm 1) to extract a consistent curve passing through the input point set. We tested the algorithm on samples extracted from different curves under varying sampling conditions and noise perturbations as seen in Figure 6. The algorithm is robust and degrades gracefully under adverse conditions.

Next we tested the robust operator to estimate visibility for 3D point sets that were corrupted with noise. Our operator consistently performed better than the original HPR operator as seen in Figure 8. Corresponding statistics about false negatives and false positives are indicated alongside the figure, while respective timings on a 2.8GHz Xeon desktop with 3GB RAM running Windows XP are listed in Table 2. The graph based technique for curve reconstruction can be suitably modified to perform surface reconstruction (see Algorithm 2). Figure 11 demonstrates the performance of the method on a variety of noise-free point-sets. The algorithm performs well even in presence of sharp features and non-uniform sampling.

In case of noisy point sets, an algorithm that interpolates the input, which often appears in a band around the original surface, produces significant fold-overs and artifacts. A better approach is to use the local connectivity estimated in the visualization step for smoothing the point set (see Section 4.4) and reconstruct the smoothed input (see Figure 12, middle row). The resultant connectivity can then be mapped to the original data (see Figure 12, bottom row). Notice the artifacts in

regions of concavity or poor sampling. Figure 14 compares the performance of original HPR operator and the proposed HPR algorithm when used for extracting local neighborhood information and smoothing, followed by reconstruction. For both methods we avoid the use of thresholds to filter out *long edges*. Notice that for the original HPR operator large number of false positives lead to a degradation of the noisy point sets under ‘smoothing’.

6. CONCLUSIONS

We analyzed the effect of noise on the performance of the HPR operator and introduced the concept of *guard bands* around the point set from where visibility cannot be reliably estimated. Using this we developed a robust HPR operator to estimate visibility and infer local connectivity for 2D and 3D point sets, without the need to explicitly perform curve or surface reconstruction. We also presented a graph based approach to collate partial connectivity information extracted from multiple visibility queries to extract a consistent manifold. Application to smoothing on noisy point sets was also demonstrated.

ACKNOWLEDGEMENTS

We thank Tamal Dey for making code for AMLS smoothing [18], robust cocone [15], and tight cocone [13] publicly available. We also thank Yongliang Yang for proofreading, and the anonymous reviewers for their extensive comments and suggestions that helped to improve the paper. The Coati scan was obtained from Mitra and colleagues [31], while the tooth, horse, Igea, and Max Planck models were obtained from the AIM@SHAPE database.

APPENDIX

Let $\mathbf{P}$ denote the original noise free point set. Let $\mathbf{P}^\sigma := \{\mathbf{p}_i + \sigma \mathbf{n}_i \mid \forall \mathbf{p}_i \in \mathbf{P}\}$ be the noisy point set, where σ is a uniform random variable over the range $[0, a]$, and $\mathbf{n}_i$ is a unit vector oriented in a uniformly chosen random direction. We may assume that the viewpoint is at the origin of the coordinate system. Let $a_{\min}$ and $a_{\max}$ be the shortest and farthest distance from the origin to a point in $\mathbf{P}$ and D be the diameter of point cloud, i.e., $(a_{\max} - a_{\min})$. For any point $\mathbf{p} \in \mathbf{P}$ let $\mathbf{p}^\sigma$ be the corresponding point in $\mathbf{P}^\sigma$, and $\hat{\mathbf{p}}$ be its image in the inverted space.

Spherical inversion function

The spherical inversion function is defined as follows.

$$\hat{\mathbf{p}} = \mathbf{p} + 2(R - \|\mathbf{p}\|)\mathbf{p}/\|\mathbf{p}\| = 2R\mathbf{p}/\|\mathbf{p}\| - \mathbf{p}.$$

For a point $\mathbf{p} \in \mathbf{P}$,

$$\begin{aligned} \hat{\mathbf{p}} &= 2R\mathbf{p}/\|\mathbf{p}\| - \mathbf{p} \\ \hat{\mathbf{p}}^\sigma &= 2R(\mathbf{p} + \sigma\mathbf{n})/\|\mathbf{p} + \sigma\mathbf{n}\| - (\mathbf{p} + \sigma\mathbf{n}). \end{aligned}$$

Next, we quantify the effect of noise on points in the inverted domain. Let ϵ denote the magnitude of noise in the inverted domain, i.e., $\epsilon = \|\hat{\mathbf{p}}^\sigma - \hat{\mathbf{p}}\|$. We have,

$$\begin{aligned}\epsilon &= \|2R \left(\frac{1}{\|\mathbf{p} + \sigma \mathbf{n}\|} - \frac{1}{\|\mathbf{p}\|} \right) \mathbf{p} + \left(\frac{2R}{\|\mathbf{p} + \sigma \mathbf{n}\|} - 1 \right) \sigma \mathbf{n}\| \\ &\leq \|2R \left(\frac{1}{\|\mathbf{p}\| - \sigma} - \frac{1}{\|\mathbf{p}\|} \right) \mathbf{p}\| + \left\| \left(\frac{2R}{\|\mathbf{p}\| - \sigma} - 1 \right) \sigma \mathbf{n} \right\| \\ &\leq 2R \left(\frac{\sigma}{(\|\mathbf{p}\| - \sigma)\|\mathbf{p}\|} \right) \|\mathbf{p}\| + \left(\frac{2R}{\|\mathbf{p}\| - \sigma} - 1 \right) \sigma \|\mathbf{n}\| \\ &\leq \frac{2R\sigma}{\|\mathbf{p}\| - \sigma} + \left(\frac{2R}{\|\mathbf{p}\| - \sigma} - 1 \right) \sigma \\ &\leq \left(\frac{4R}{\|\mathbf{p}\| - \sigma} - 1 \right) \sigma.\end{aligned}$$

So the maximum noise in the inverted domain is bounded by

$$\epsilon_{\max} = \left(\frac{4R}{a_{\min} - \sigma} - 1 \right) \sigma.$$

In our algorithm for robust visibility we project the noisy points onto the convex hull. This projection is robust and consistent only when $2\epsilon_{\max} \leq \alpha D$, where $\alpha \in [0, 1]$. Simplifying we get,

$$R \leq \left(\frac{\alpha D}{2\sigma} + 1 \right) \left(\frac{a_{\min} - \sigma}{4} \right).$$

Now we find a range of value of R over which we can reliably perform this projection. Recall that in the noise-free scenario, convex hull of the inverted point cloud along with the viewpoint gives us the correct visibility estimates. But in presence of noise, the points in the inverted domain may get perturbed by at most $2\epsilon_{\max}$, causing some of the previously visible points to be marked as hidden. To solve this problem, we project points back to the convex hull if they are within $2\epsilon_{\max}$ from the (new) convex hull.

While the projection operator works well for locally convex regions, it may lead to false negatives in concave regions. In this case, the noisy points can be even farther than $2\epsilon_{\max}$ from the convex hulls in the inverted regions and may lead to false negatives. As concave region visibility is proportional to the radius of inversion R (Lemma 4.3 in [26]), we have to select R that is greater than the minimum radius at which the concave region is visible. This imposes a lower bound on R .

1) *Locally convex region*: Locally convex shapes are visible at any radius, we can use any radius above minimum radius possible for inversion, i.e., $a_{\max}$.

2) *Locally concave region*: Concave regions, see Lemma 4.3 in [26], are correctly visible only above a certain radius given by the local curvature κ . For the simple case when the tangent to the surface is perpendicular to the line of sight $R > \kappa r^2/2$. In this case, the lower bound for R is not $a_{\max}$ but some other value say R' , greater than $a_{\max}$. Let $R' = ma_{\max}$ for some $m > 1$.

Lemma 2. *Given a noisy point cloud $\mathbf{P}^\sigma$, projection can be applied if*

$$ma_{\max} \leq R \leq \left(\frac{\alpha D}{2\sigma} + 1 \right) \left(\frac{a_{\min} - \sigma}{4} \right).$$

Since $a_{\max} = a_{\min} + D$, we have

$$m(a_{\min} + D) \leq R \leq \left(\frac{\alpha D}{2\sigma} + 1 \right) \left(\frac{a_{\min} - \sigma}{4} \right).$$

Simplifying we get,

$$\left(4m + \frac{\alpha}{2} \right) D + \sigma \leq \left(\frac{\alpha D}{2\sigma} - (4m - 1) \right) a_{\min}.$$

We now have three scenarios based on the noise level:

- 1) **Case $\sigma = 0$** : No noise, hence, there is no restriction on $a_{\min}$.
- 2) **Case $\sigma < \frac{\alpha D}{2(4m-1)}$** : For locally convex regions, since $m = 1$ the bound becomes $\alpha D/6$. For concave regions, as $m > 1$ the noise threshold is smaller than the $\alpha D/6$. This suggests that in presence of concave regions, the capability of resolving high amounts of noise decreases. Thus,

$$a_{\min} \geq \frac{(4m + \frac{\alpha}{2})D + \sigma}{\left(\frac{\alpha D}{2\sigma} - (4m - 1) \right)}.$$

- 3) **Case $\sigma \geq \frac{\alpha D}{2(4m-1)}$** : Such a large amount of noise cannot be handled with this method without introducing many false positives, since distinguishing points of different surface parts from noisy points becomes ambiguous.

In second case, we can get a minimum distance to $\mathbf{P}^\sigma$ above which robust visibility can be determined. This defines the boundary of the guard zone (see Theorem 2). Note that the guard zone is thicker around concave regions than around convex ones.

Exponential Inversion Function

Equivalent results can also be obtained for the exponential inversion function, which is defined as follows:

$$\hat{\mathbf{p}} = \mathbf{p} / \|\mathbf{p}\|^\gamma$$

where, $\gamma > 1$ is the inversion parameter and $\|\mathbf{p}\| < 1$. For any point $\mathbf{p} \in \mathbf{P}$ we have,

$$\begin{aligned}\hat{\mathbf{p}} &= \mathbf{p} / \|\mathbf{p}\|^\gamma \\ \hat{\mathbf{p}}_\sigma &= (\mathbf{p} + \sigma \mathbf{n}) / \|\mathbf{p} + \sigma \mathbf{n}\|^\gamma.\end{aligned}$$

Let ϵ denote the magnitude of noise in the inverted domain, i.e., $\epsilon = \|\hat{\mathbf{p}}_\sigma - \hat{\mathbf{p}}\|$. Then

$$\begin{aligned}\epsilon &= \left\| \frac{(\mathbf{p} + \sigma \mathbf{n})}{\|\mathbf{p} + \sigma \mathbf{n}\|^\gamma} - \frac{\mathbf{p}}{\|\mathbf{p}\|^\gamma} \right\| \\ &= \left\| \mathbf{p} \left(\frac{1}{\|\mathbf{p} + \sigma \mathbf{n}\|^\gamma} - \frac{1}{\|\mathbf{p}\|^\gamma} \right) + \frac{\sigma \mathbf{n}}{\|\mathbf{p} + \sigma \mathbf{n}\|^\gamma} \right\| \\ &\leq \|\mathbf{p}\| \left(\frac{1}{(\|\mathbf{p}\| - \sigma)^\gamma} - \frac{1}{\|\mathbf{p}\|^\gamma} \right) + \frac{\sigma \|\mathbf{n}\|}{(\|\mathbf{p}\| - \sigma)^\gamma} \\ &\leq \frac{1}{\|\mathbf{p}\|} \gamma^{-1} \left(\left(1 - \frac{\sigma}{\|\mathbf{p}\|} \right)^{-\gamma} - 1 \right) + \frac{\sigma}{(\|\mathbf{p}\| - \sigma)^\gamma} \\ &\leq \frac{\gamma \sigma}{\|\mathbf{p}\|^\gamma} + \frac{\sigma}{(\|\mathbf{p}\| - \sigma)^\gamma} \\ &\leq \left(\gamma \left(1 + \frac{\sigma}{a_{\min}} \right) + 1 \right) \frac{\sigma}{\|\mathbf{p}\|^\gamma}.\end{aligned}$$

So the maximum noise in the inverted domain is bounded by

$$\epsilon_{\max} = \left(\gamma \left(1 + \frac{\sigma}{a_{\min}} \right) + 1 \right) \frac{\sigma}{\|\mathbf{p}\|^\gamma}.$$

While the first factor is linear, the second term is exponential in γ . Variation in $\epsilon_{\max}$ is mainly due to the exponential term. Therefore, we approximate the linear term with a constant ϕ to get,

$$\epsilon_{\max} = \phi\sigma/a_{\min}^{\gamma}.$$

In our algorithm for robust visibility we project the noisy points onto the convex hull. This projection is robust and consistent only when $2\epsilon_{\max} \leq \alpha D$, where $\alpha \in [0, 1]$. Simplifying we get,

$$\gamma \leq \log(\alpha D/(2\phi\sigma))/\log(1/a_{\min}).$$

Lemma 3. *Given a noisy point cloud $\mathbf{P}^{\sigma}$, projection can be applied if*

$$m \leq \gamma \leq \log(\alpha D/(2\phi\sigma))/\log(1/a_{\min}).$$

It follows that,

$$a_{\min} \geq (2\phi\sigma/(\alpha D))^{1/m}.$$

Since $a_{\min} < 1$, we get

$$1 > a_{\min} \geq (2\phi\sigma/(\alpha D))^{1/m}$$

leading to,

$$\sigma < \alpha D/(2\phi).$$

REFERENCES

- [1] Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T. Silva. Point set surfaces. In *VIS '01: Proceedings of the conference on Visualization '01*, pages 21–28, Washington, DC, USA, 2001. IEEE Computer Society.
- [2] Nina Amenta, Marshall Bern, and David Eppstein. The crust and the beta-skeleton: Combinatorial curve reconstruction. In *Graphical Models and Image Processing*, pages 125–135, 1998.
- [3] Nina Amenta, Sunghee Choi, and Ravi Krishna Kolluri. The power crust, unions of balls, and the medial axis transform. *Computational Geometry: Theory and Applications*, 19:127–153, 2000.
- [4] Nina Amenta and Yong Joo Kil. Defining point-set surfaces. *ACM Trans. Graph.*, 23(3):264–270, 2004.
- [5] A. Appel. Some techniques for shading machine renderings of solids. In *AFIPS Spring Joint Computer Conf.*, volume 32, pages 37–45, 1968.
- [6] Sunil Arya and David M. Mount. Approximate nearest neighbor queries in fixed dimensions. In *SODA '93: Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms*, pages 271–280, Philadelphia, PA, USA, 1993. Society for Industrial and Applied Mathematics.
- [7] Jiri Bittner and Peter Wonka. Visibility in computer graphics. *Environment and Planning B: Planning and Design*, 30(5):729–755, 2003.
- [8] Edwin Earl Catmull. *A subdivision algorithm for computer display of curved surfaces*. PhD thesis, 1974.
- [9] Thomas T. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to algorithms*. MIT Press, Cambridge, MA, USA, 1990.
- [10] James Davis, Steven R. Marschner, Matt Garr, and Marc Levoy. Filling holes in complex surfaces using volumetric diffusion. In *First International Symposium on 3D Data Processing, Visualization, and Transmission*, pages 428–438, 2001.
- [11] Tamal Dey. *Curve and Surface Reconstruction : Algorithms with Mathematical Analysis*. Cambridge University Press, 2006.
- [12] Tamal K. Dey and Joachim Giesen. Detecting undersampling in surface reconstruction. In *SCG '01: Proceedings of the seventeenth annual symposium on Computational geometry*, pages 257–263, 2001.
- [13] Tamal K. Dey and Samrat Goswami. Tight cocone: a water-tight surface reconstructor. In *SM '03: Proceedings of the eighth ACM symposium on Solid modeling and applications*, pages 127–134, New York, NY, USA, 2003. ACM.
- [14] Tamal K. Dey and Samrat Goswami. Provable surface reconstruction from noisy samples. In *SCG '04: Proceedings of the twentieth annual symposium on Computational geometry*, pages 330–339, New York, NY, USA, 2004. ACM.
- [15] Tamal K. Dey and Samrat Goswami. Provable surface reconstruction from noisy samples. *Comput. Geom. Theory Appl.*, 35(1):124–141, 2006.
- [16] Tamal K. Dey, Kurt Mehlhorn, and Edgar A. Ramos. Curve reconstruction: connecting dots with good reason. In *SCG '99: Proceedings of the fifteenth annual symposium on Computational geometry*, pages 197–206, New York, NY, USA, 1999. ACM.
- [17] Tamal K. Dey and Jian Sun. An adaptive mls surface for reconstruction with guarantees. In *SGP '05: Proceedings of the third Eurographics symposium on Geometry processing*, page 43, Aire-la-Ville, Switzerland, Switzerland, 2005. Eurographics Association.
- [18] Tamal K. Dey and Jian Sun. An adaptive mls surface for reconstruction with guarantees. In *SGP '05: Proceedings of the third Eurographics symposium on Geometry processing*, page 43, Aire-la-Ville, Switzerland, Switzerland, 2005. Eurographics Association.
- [19] Tamal K. Dey and Rephael Wenger. Fast reconstruction of curves with sharp corners. *Int. J. Comput. Geometry Appl.*, 12(5):353–400, 2002.
- [20] Andrea Francke and Michael Hoffmann. The euclidean degree-4 minimum spanning tree problem is np-hard. In *SCG '09: Proceedings of the 25th annual symposium on Computational geometry*, pages 179–188, New York, NY, USA, 2009. ACM.
- [21] M. R. Garey and D. S. Johnson. *Computers and Intractability : A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences)*. W. H. Freeman, January 1979.
- [22] Markus Gross and Hanspeter Pfister, editors. *Point-Based Graphics*. Elsevier Science and Technology Books, 2007.
- [23] Hugues Hoppe, Tony Deroose, Tom Duchamp, John Alan McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points, 1992.
- [24] H. Huang, D. Li, H. Zhang, U. Ascher, and D. Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. *ACM Transactions on Graphics*, 28(5), 2009. to appear.
- [25] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.
- [26] Sagi Katz, Ayellet Tal, and Ronen Basri. Direct visibility of point sets. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers*, page 24, New York, NY, USA, 2007. ACM.
- [27] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *SGP '06: Proceedings of the fourth Eurographics symposium on Geometry processing*, pages 61–70, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.
- [28] David Levin. The approximation power of moving least-squares. *Math. Comput.*, 67(224):1517–1531, 1998.
- [29] Boris Mederos, Nina Amenta, Luiz Velho, and Luiz Henrique de Figueiredo. Surface reconstruction from noisy point clouds. In *SGP '05: Proceedings of the third Eurographics symposium on Geometry processing*, page 53, Aire-la-Ville, Switzerland, Switzerland, 2005. Eurographics Association.
- [30] Surface Meshes, Surface Meshes, J. Vollmer, J. Vollmer, R. Mencl, R. Mencl, H. Mller, and H. Muller. Improved laplacian smoothing of noisy surface meshes, 1999.
- [31] Niloy J. Mitra, Simon Flory, Maks Ovsjanikov, Natasha Gelfand, Leonidas Guibas, and Helmut Pottmann. Dynamic geometry registration. In *Symposium on Geometry Processing*, pages 173–182, 2007.
- [32] Niloy J. Mitra, Ann Nguyen, and Leonidas Guibas. Estimating surface normals in noisy point cloud data. In *special issue of International Journal of Computational Geometry and Applications*, volume 14, pages 261–276, 2004.
- [33] Mark Pauly, Leif P. Kobbelt, and Markus Gross. Point-based multiscale surface representation. *ACM Trans. Graph.*, 25(2):177–193, 2006.
- [34] Mark Pauly, Niloy J. Mitra, and Leonidas Guibas. Uncertainty and variability in point cloud surface data. In *Symposium on Point-Based Graphics*, 2004.
- [35] Schnelle Kurven Straer Wolfgang. *Flchendarstellung auf graphischen Sichtgeräten*. PhD thesis, 1974.
- [36] Ivan E. Sutherland, Robert F. Sproull, Robert, and A. Schumacker. A characterization of ten hidden-surface algorithms. *ACM Computing Surveys*, 6:1–55, 1974.