Denoising Point Sets via L_0 Minimization

Yujing Sun[a,*], Scott Schaefer[b], Wenping Wang[a]

[a] *University of Hong Kong, Department of Computer Science, Pokfulam Road, Hong Kong*
[b] *Texas A & M University, Department of Computer Science and Engineering, 3112 Texas A & M University College Station, TX 77802, USA*

Abstract

We present an anisotropic point cloud denoising method using L_0 minimization. The L_0 norm directly measures the sparsity of a solution, and we observe that many common objects can be defined as piece-wise smooth surfaces with a small number of features. Hence, we demonstrate how to apply an L_0 optimization directly to point clouds, which produces sparser solutions and sharper surfaces than either the L_1 or L_2 norms. Our method can faithfully recover sharp features while at the same time smoothing the remaining regions even in the presence of large amounts of noise.

Keywords: point set, denoising, L_0 minimization, L_0 sparsity

1. Introduction

Surface reconstruction is a widely-used geometry processing tool for digitizing real-world objects. In many cases, the input to a reconstruction algorithm is a point set acquired from the object in question. However, despite new methods and acquisition hardware, errors such as noise and outliers inevitably appear in these point sets. Moreover, the quality of the reconstructed surface strongly depends on the quality of the input point set. Yet denoising point sets is inherently a challenging problem since, by definition, there is no connectivity information to guide the denoising process. Denoising point sets with sharp features is even more problematic, especially in the presence of large noise, because these features are difficult to distinguish from noise.

Our observation is that many common surfaces are piece-wise smooth; that is, the surface is smooth almost everywhere except at some small number of sparse features that form sharp features. Hence, we can explicitly take advantage of that sparsity and optimize for such a surface. The idea of directly optimizing the sparsity of a solution is a key idea in the field of compressed sensing [1, 2]. The main insight to optimize for sparse solutions is to measure error in a norm related to the sparseness of the solution. The L_0 norm is one such metric that measures the number of non-zero entries in a vector and is directly related to sparsity. However, this norm is difficult to optimize because of its discrete, combinatorial nature. Thus, as a compromise, many researchers use the sparsest convex norm, the L_1 norm, to generate a tractable optimization that still produces a sparse solution.

In Computer Graphics, the L_1 norm has been used in several areas including image smoothing and deblurring [3, 4, 5], mesh denoising [6], and point set reconstruction [7]. More recently several techniques have been developed that optimize the L_0 norm directly to smooth [8] and deblur [9] images. He et al. [10] also extended L_0 smoothing for images [8] to surfaces.

Yet, to the best of our knowledge, there are no existing methods that directly denoise point clouds using the L_0 norm, which produces the sparsest solutions. Inspired by the recent work on L_0 minimization for

*Corresponding author
Email addresses: yjsun@cs.hku.hk (Yujing Sun), schaefer@cs.tamu.edu (Scott Schaefer), wenping@cs.hku.hk (Wenping Wang)

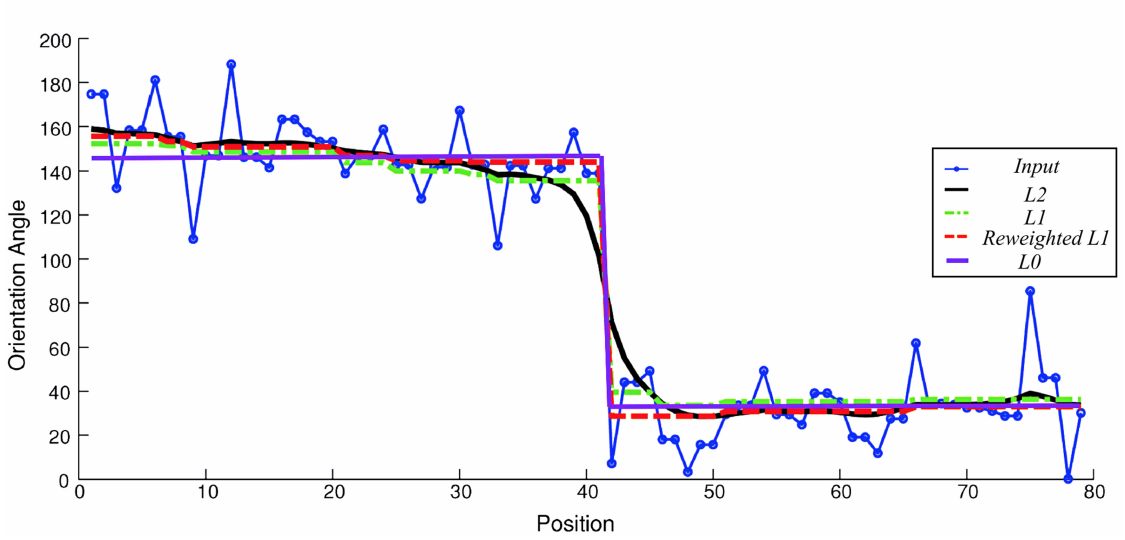

Figure 1: Comparison of smoothing a 1D signal, one set of point orientations, using different norms: L_2, L_1, reweighted L_1, and L_0. We use the same input data set as appearing in [7].

images [8] and meshes [10], we present an algorithm to denoise point sets via L_0 minimization. Our method can effectively eliminate noise to maximize smooth regions as well as recover sharp features.

To summarize the contributions of our work, we first extend L_0 minimization from images and meshes to unstructured point clouds. Moreover, our method faithfully recovers point positions as well as point orientations and enhances the performance of Edge Aware Upsampling (EAR) [11] even in the presence of high amounts of noise. Finally, we develop a projection operator to recover sharp features.

2. Related Work

Denoising point clouds has been studied by many researchers, especially in the context of surface reconstruction. Locally Optimal Projector (LOP) related methods [12, 11, 13, 14, 15] have recently attracted much attention for its robustness against outliers. The core of LOP operator is to project an arbitrary number of particles to a point set to represent the local L_1 median of the original point set. Weighted LOP (WLOP) [12] improved the original LOP [14] by producing a more evenly distributed point set; Kernel LOP (KLOP) reduced the computation cost of the original LOP; Anisotropic LOP [11] can better preserve sharp feature than WLOP and KLOP by anisotropically projecting points to local L_1 median according to point orientations. However, LOP methods use local operators, which can affect the quality of the output especially when, locally, high noise-to-signal ratios yield redundant features or, in the other extreme, oversmoothing [7].

The idea of directly optimizing in a space of sparse solutions is a concept from signal processing that is beginning to have a significant influence on Computer Graphics. These optimizations are typically global in nature, and do not suffer from the problems of locality that LOP methods can. The L_0 norm directly measures sparsity, but direct minimization of the L_0 norm is a highly non-convex problem and is difficult to optimize due to its discrete, combinatorial nature [10]. Hence, many researchers use a convex norm such as L_1 that still tends to generate sparse solutions. In the content of image processing, L_1 norm has been adopted to denoise and deblur using the sparsity of the total variation of images [3, 4, 5]. For 3D surface reconstruction, Avron et al. [7] incorporated the notion of L_1-Sparsity to denoise point sets by directly applying a re-weighted L_1 minimization procedure to restore point orientations followed by restoring point positions in order to preserve sharp features. Our method is also a global method but can handle higher level noise than [7], since L_0 is a sparser solution than L_1. Although their method can produce reasonable

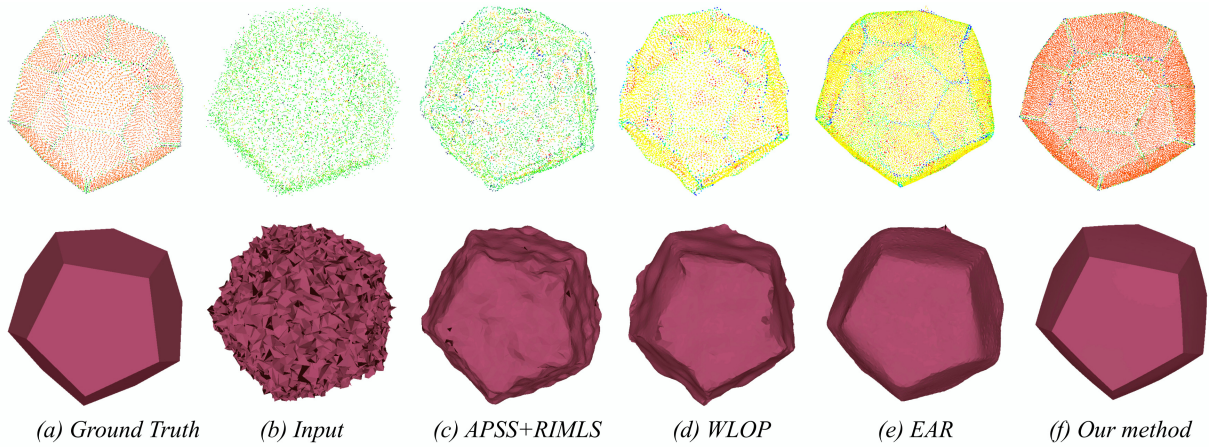

Figure 2: Comparison between our method and various state-of-the-art point set denoising methods. The first row from left to right: (a) ground truth (7582 Points), (b) input noisy point cloud (7582 Points), (c) APSS + RIMLS (7287 Points), (d) WLOP (7062 Points), (e) EAR (AWLOP + UPSAMPLE_EAR) (18K Points), and (f) our method (L_0 Minimization + UPSAMPLE_EAR) (18K Points). Points are colored by curvature. The second row shows the surface reconstructed from the corresponding point set using the "ball pivoting" algorithm [37].

results, points on an edge are sometimes not faithfully recovered, and the corresponding reconstructed edge is not straight or smoothly curved. Our projection operator can better recover points along the edges.

More recently several techniques have been developed that directly attempt to optimize the L_0 norm. Such optimizations have been applied to image smoothing [8], image deblurring[9], and even anisotropic surface denoising [10].

While not directly related to point cloud denoising, surface reconstruction methods often use point clouds, and the quality of the reconstructed surface strongly depends on that of the input point cloud. There are two main classes of surface reconstruction techniques: parametric and implicit. Parametric methods explicitly build the topology of the reconstructed surface and often use Delaunay triangulations or Voronoi diagrams [16, 17, 18]. Implicit methods construct a function whose level set is the reconstructed surface. These implicit functions can take a variety of forms such as signed distance fields [19, 20, 21]. Radial basis functions [22] and even solutions to the Poisson equation [23, 24] have also been used to create these implicit functions. However, all the methods mentioned either perform poorly in the presence of noisy data or oversmooth the output surface.

Another related method is the idea of performing surface reconstruction from point clouds using Moving Lease Squares (MLS) [25, 26, 27, 28, 29]. These procedures perform surface reconstruction by iteratively projecting points to a locally fit polynomial. MLS is designed by nature to reconstruct surfaces that are smooth everywhere. In order to overcome the limitation, many approaches have been incorporated to MLS to better preserve sharp features such as cell complexes [30], tagged point clouds [29], and robust statistics [31, 32]. However, like LOP methods, the locality of the operations can lead to artifacts that global methods avoid when the amount of noise is large.

Our work uses L_0 optimization to directly smooth point clouds and, hence, can handle large amounts of noise. Part of our smoothing process involves estimating normals for each point. There are variety of approaches for reconstructing point normals, ranging from simply fitting local tangent plane [21] to more robust methods such as outlier removal [12], L_1 minimization [7], randomized Hough transform [33], and robust statistics based methods [34, 35, 32, 36]. In this paper, we will show that our L_0 minimization method can not only robustly restore point positions, but also faithfully reconstruct point orientations.

2.1. L_0 Optimization

The L_0 norm directly measures sparsity but is difficult to optimize directly. Recently, Xu et al. [8] provide an algorithm for directly optimizing the L_0 norm in the context of image smoothing to create piece-wise

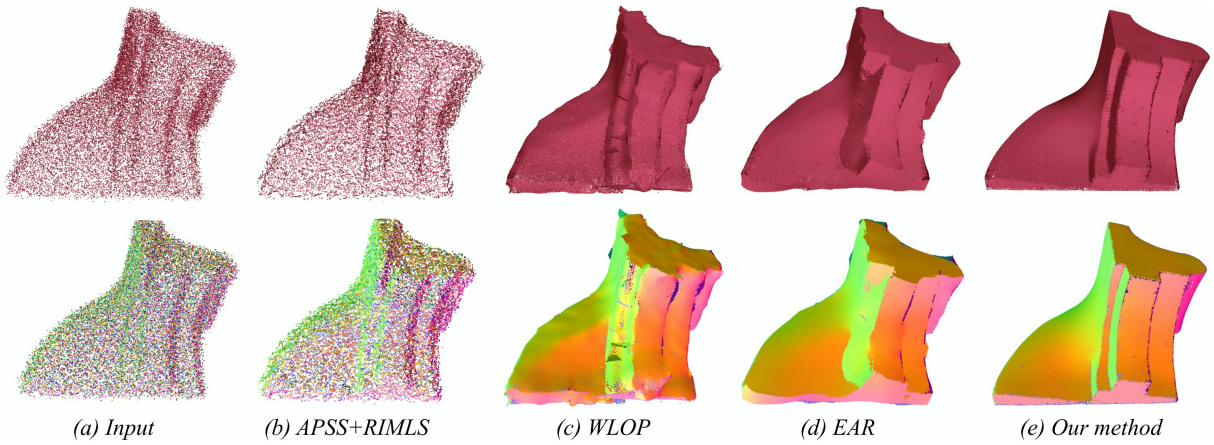

Figure 3: Comparison between our method and various state-of-the-art point set denoising methods. The first row from left to right: (a) Input noisy point cloud (27K points), (b) APSS + RIMLS (26K points), (c) WLOP (127K points), (d) EAR (AWLOP + UPSAMPLE_EAR) (182K points), and (e) our method (L_0 Minimization + UPSAMPLE_EAR) (182K points). The second row shows the corresponding normal colorization.

constant images and He et al. [10] use a similar L_0 minimization strategy to denoise meshes. Our work extends the L_0 minimization concept from image smoothing and mesh denoising to point set denoising. In this section, we review L_0 minimization in the context of one dimensional signal smoothing, two dimensional image smoothing, and three dimensional mesh denoising.

The L_0 norm of a vector v is defined as the number of non-zero entries. That is,

$$|v|_0 = \sum_i \#\{v_i \ \neq 0\}.$$

Given a 1D signal $\hat{S}$ and a differential operator D, we can denoise $\hat{S}$ by optimizing for the values S that minimize

$$\min_S |S - \hat{S}|^2 + \lambda |D(S)|_0 \tag{1}$$

where the first term is a data fidelity term to ensure that the output values do not stray too far from the input and λ controls how smoothed the output will be. $D(S)$ is an operator that returns a vector of values. For example, if the output S should be piecewise constant, $D(S)$ should be chosen to annihilate constant functions. For example, $D(S)_i = S_{i+1} - S_i$. Figure 1 shows a comparison using this operator and smoothing a 1D signal with different norms, demonstrating that L_0 norm produces sparsest solution.

Xu et al. [8] use this optimization to smooth images and define S to be the pixel colors in an image and $D(S)$ to be a vector of color gradients. He et al. [10] also use this framework to perform anisotropic smoothing of surfaces. The authors choose S to be the positions of vertices in a mesh and derive an edge-based Laplacian operator $D(S)$ that annihilates linear functions.

To minimize Equation 1, both [8] and [10] introduce an auxiliary variable σ to form

$$\min_{S,\sigma} |S - \hat{S}|^2 + \beta |D(S) - \sigma|^2 + \lambda |\sigma|_0 \tag{2}$$

where β controls how quickly this minimization problem approaches Equation 1 and is initially set to a small value. Then, given an initial guess for S, the authors minimize this problem by first holding S constant and solving for σ, which gives the following minimization.

$$\min_\sigma \beta |D(S) - \sigma|^2 + \lambda |\sigma|_0$$

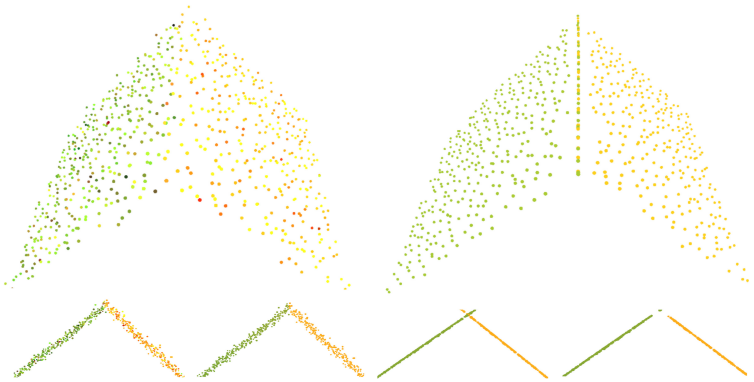

Figure 4: A simple demonstration of our L_0 Minimization approach. Points are colored by normal information. First row from left to right: a noisy point cloud (727 points) and the result of our L_0 approach. The second row shows the results, from a side view, of denoising the input point cloud after each stage. From left to right: initial noisy point cloud with orientations computed from a local PCA, the result after Normal Estimation, the result after Point Denoising, and the result after Edge Recovery.

The solution of this optimization is given by $\sigma_i = 0$ if $\frac{\lambda}{\beta} > |D(S)_i|^2$ or $\sigma_i = D(S)_i$ otherwise. Next, the authors hold σ constant and solve for S in Equation 2, resulting in the following minimization

$$\min_S |S - \hat{S}|^2 + \beta |D(S) - \sigma|^2. \tag{3}$$

This equation is quadratic and, thus, has a global minimum that is easy to find. These two optimizations are then repeated with $\beta = 2\beta$ to force $D(S)$ to approach σ as β approaches infinity.

3. L_0 Minimization for Point Cloud Denoising

One significant difference that distinguishes point clouds from images and meshes is that point clouds do not have connectivity information, while images or meshes have pixels and vertices with fully defined neighborhoods. Similar to Section 2.1, we define S as a vector of points. However, $D(S)$ is harder to define due to the lack of well-defined neighbors. As a consequence, it is difficult to apply the techniques from Section 2.1 directly to point clouds.

To simplify the problem, we decouple orientations and positions. Using a similar L_0 optimization, we start by solving for normals and then, based on this normal information, restore point positions. Our algorithm can be decomposed into 3 steps. First, we estimate normals by observing that normals between close points should be varying smoothly except near sharp features. Then, we modify point positions based on the estimated normal information with the observation that if a point lies on a smooth region, the point and its k nearest neighbors should form a plane perpendicular to the point normal. Finally, we recover points along edges to better represent the underlying geometry. The procedure is repeated until convergence. A simple demonstration of our algorithm is shown in Figure 4 and a more complicated example appears in Figure 12.

3.1. Normal Estimation

To estimate normals for each point in the point cloud, we first compute initial normals $\hat{N}$ using a local PCA. Then we minimize the objective function

$$\min_{N, |N_i|=1} |N - \hat{N}|^2 + \eta |D(N)|_0$$

where $D(N)_{ik+j} = N_i - N_{M(i,j)}$ and $M(i,j)$ gives the j^{th} entry in the set of k nearest neighbors of point i. Therefore, $D(N)$ is a vector whose length is k times the number of input points. Similar to Section 2.1, we

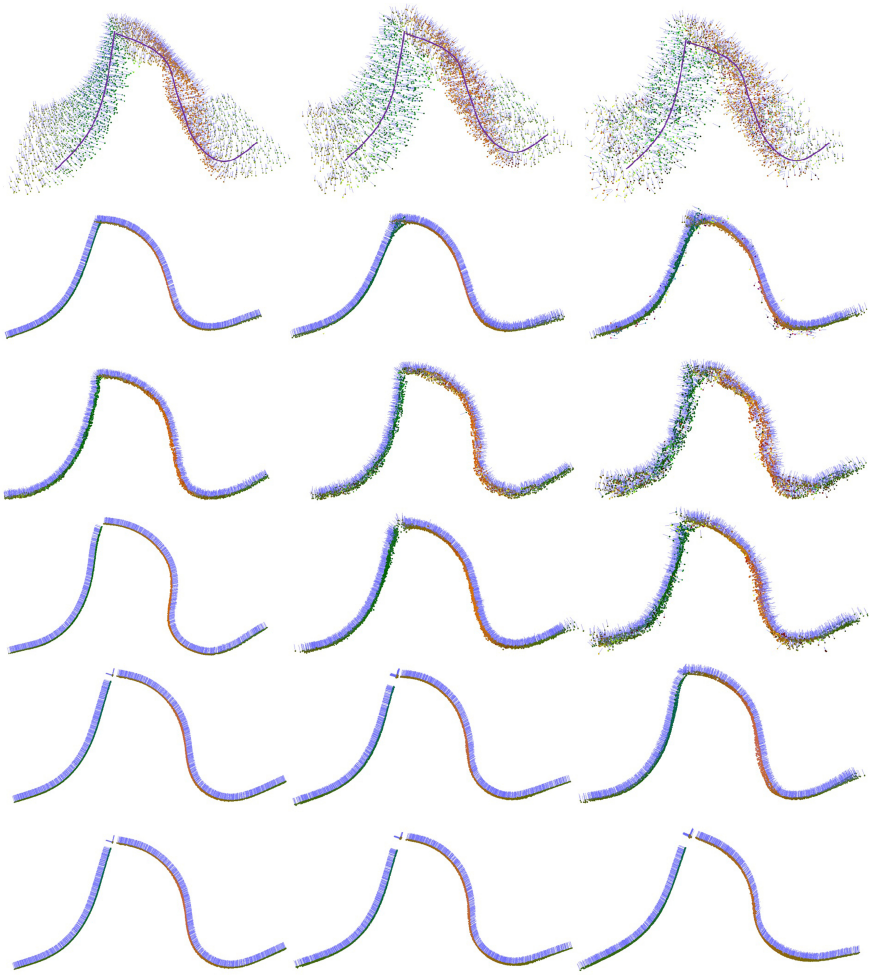

Figure 5: Comparison between our L_0 method and existing methods in estimating normals. First row is the input data corrupted with 1.5%, 2.5%, and 3.5% Gaussian noise respectively. Rows 2 - 6 are the results of: Bilateral Filtering, RIMLS, Anisotropic WLOP, L_0 with default parameters, and L_0 with refined parameters.

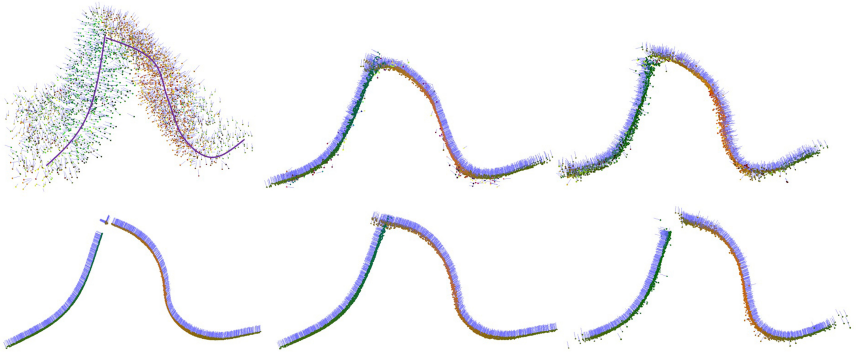

Figure 6: Comparison of existing methods in recovering point positions based on our L_0 denoised normals. First row from left to right: input point cloud corrupted with 3.5% noise (the same model as in Figure 5), Bilateral Filtering and Anisotropic WLOP using normal computed with a local PCA. Second row from left to right: L_0, Bilateral Filtering and Anisotropic WLOP using normal computed with L_0.

introduce auxiliary variables θ, a vector with the same length as $D(N)$, to the optimization

$$\min_{\theta,N,|N_i|=1} |N - \hat{N}|^2 + \beta |D(N) - \theta|^2 + \eta |\theta|_0,$$

which can also be written as

$$\min_{\theta,N,|N_i|=1} |N - \hat{N}|^2 + \beta \sum_i \sum_j^k \mid D(N)_{ik+j} - \theta_{ik+j} \mid^2 + \eta |\theta|_0.$$

We solve this equation using the alternating optimization method as before except the $|N_i| = 1$ constraint leads to a constrained quadratic minimization in Equation 3.

3.2. Point Denoising

Next we use the estimated normals N to reposition the points. Since normals encode higher order geometric information of a shape [38], shifting points along normal directions can smooth the underlying surface. Therefore, to reduce the number of degrees of freedom in our optimization without significantly affecting the quality of the output surface, we restrict each point to only move in the direction of its normal. Let $\hat{P}$ be the initial point set. We then optimize

$$\min_P |P - \hat{P}|^2 + \delta |D(P)|_0.$$

Here $D(P)$ will measure the deviation of each point in the k nearest neighbors from the plane defined by each point and normal; that is, $D(P)_{ik+j} = (P_i - P_{M(i,j)}) \cdot N_i$. Hence, the above objective function becomes

$$\min_P |P - \hat{P}|^2 + \delta \sum_i \sum_j^k \mid D(P)_{ik+j} \mid_0.$$

Since we restrict each P_i to only move along its normal direction N_i, then $P = \tilde{P} + \alpha N$ where α is a diagonal matrix of coefficients and $\tilde{P}$ are the positions of the points from the previous iteration of the full optimization procedure and, initially, $\tilde{P} = \hat{P}$. As a result, we can optimize P with respect to the entries in α. Therefore,

$$\min_\alpha \sum_i |\tilde{P}_i + \alpha_i N_i - \hat{P}_i|^2 + \delta \sum_i \sum_j^k |((\tilde{P}_i + \alpha_i N_i) - (\tilde{P}_{M(i,j)} + \alpha_{M(i,j)} N_{M(i,j)})) \cdot N_i|_0$$

where α_i refers to the i^{th} diagonal entry of the matrix α. Expanding this expression yields

$$\min_\alpha \sum_i |\tilde{P}_i + \alpha_i N_i - \hat{P}_i|^2 + \delta \sum_i \sum_j^k |D(\tilde{P})_{ik+j} + \alpha_i - (N_i \cdot N_{M(i,j)}) \alpha_{M(i,j)}|_0$$

where $D(\tilde{P})_{ik+j}$ refers to $(\tilde{P}_i - \tilde{P}_{M(i,j)}) \cdot N_i$. Again, this problem can be solved by adding auxiliary variables and applying the alternating optimization approach in Section 2.1.

3.3. Edge Recovery

Figures 4 and 7 show that when the point cloud contains a large amount of noise, a cross artifact will be produced after the Point Denoising Phase in Section 3.2. To reduce this artifact and better present the sharp features, we use a projection operator to reposition points near sharp features.

We observe that points on opposite sides of sharp features have normals with different orientation and points on smooth regions have small variation between their normals. For each vertex P_i we classify the vertex as being close to a sharp feature if there exists another vertex $P_{M(i,j)}$ that is one of the k nearest

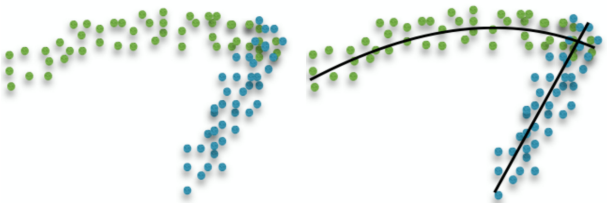

Figure 7: When the point set is noisy, Point Denoising Phase may produce a "cross" artifact.

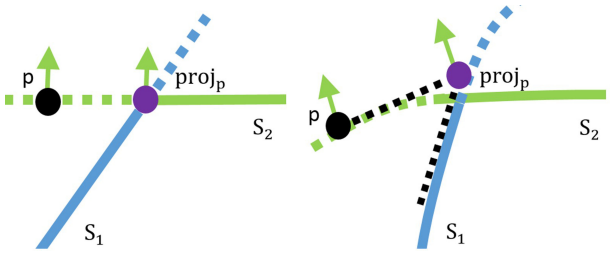

Figure 8: Illustration of Edge Recovery: projecting a point p to the intersection of planar surfaces and curved surfaces.

neighbors of P_i and $|N_i - N_{M(i,j)}| > \epsilon$. Next, for each such vertex P_i, we reposition P_i such that it minimizes the distance to all of the planes defined by a point $P_{M(i,j)}$ in P_i's neighborhood and the point's corresponding normal $N_{M(i,j)}$. Therefore, the new position x of the point P_i is

$$\min_x \sum_j^k |N_{M(i,j)} \cdot (x - P_{M(i,j)})|^2 + |N_i \cdot (x - P_i)|^2.$$

Note that it is possible that this minimization is under-constrained in the presence of a sharp edge. In that case we find the point x that is closest to P_i in the solution space. The effect of this projection is that the points P_i will be projected to the sharp feature.

Figure 8 demonstrates this projection operator with a simple example where two surfaces meet. In Figure 8*a*, the surfaces are both planar and the projected point $proj_p$ will lie on the edge. However, in Figure 8*b*, the surfaces are curved. As a consequence, the projected point $proj_p$ will not lie exactly on the sharp feature. However, our optimization is iterative and, at each iteration, $proj_p$ moves closer to the sharp feature. Figures 4 and 12 show results of the projection operator on noisy point sets.

Our Edge Recovery method works well for both straight and curved sharp features. Figures 3, 11, 12, and 13 all demonstrate shapes containing curved sharp edges.

4. Point Cloud Upsampling

Our L_0 minimization produces piece-wise smooth point sets with points lying on edges and robust normal estimation. Yet, the Edge Recovery Process described in Section 3.3 suffers from a side-effect that gaps appear near edges since nearby vertices have been projected to the edge as shown in Figures 4 and 12. To bridge the gaps near sharp features, we upsample results of L_0 minimization using the second step of the EAR algorithm [11].

The original EAR method [11] consists of two steps. The first phase (AWLOP_EAR) is to resample points away from edges based on an anisotropic WLOP operator in order to produce reliable point orientations. Then, based on these normals, new points are inserted and projected on to the unknown underlying surfaced defined by the point set (UPSAMPLE_EAR). Since the point insertion procedure relies on high-quality normals near edges, the authors state that poor normal estimation leads to unacceptable upsampling [11].

We post-process our L_0 minimization result using UPSAMPLE_EAR. Since our L_0 method can generate more reliable point orientations, upsampling our result using UPSAMPLE_EAR produces more accurate result than does the original EAR. We show comparisons between UPSAMPLE_EAR on our L_0 and the original EAR in Figures 9 and 11, indicating our method can better preserve sharp features.

5. Results and Discussion

In all of the examples in the paper, if with no specification, we corrupted point sets with Gaussian noise using a standard deviation of 2% of the length of the bounding box diagonal. The armadillo in Figure 15

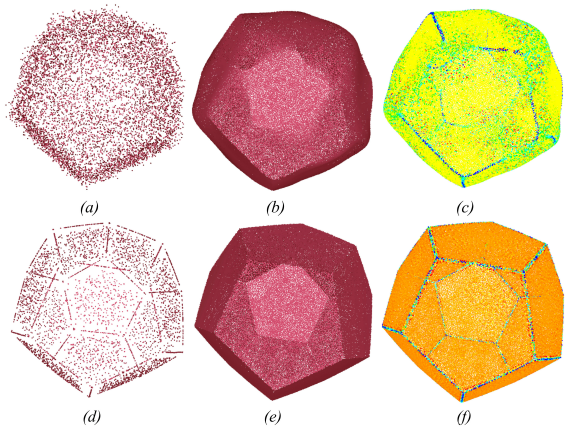

Figure 9: We show comparison of UPSAMPLE_EAR on our L_0 method and the original EAR method. The figures are (a) Input (7582 points), (b) EAR on Figure 9a (42K points), (c) Curvature colorization of Figure 9b, (d) L_0 Minimization on Figure 9a (7582 points), (e) UPSAMPLE_EAR on Figure 9d (42K points), and (f) curvature colorization of Figure 9e.

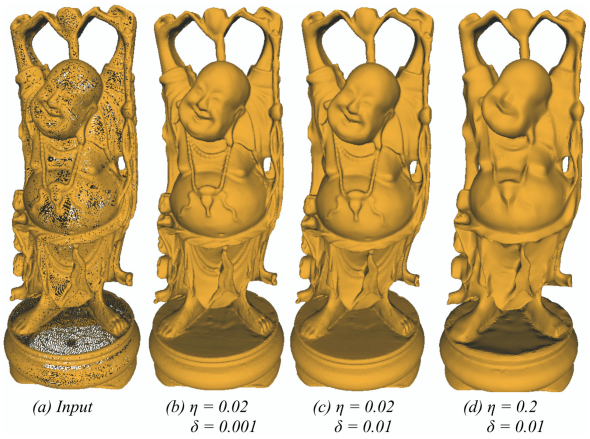

Figure 10: Illustration of how smoothing parameters η and δ affect our L_0 Minimization. We apply L_0 Minimization to a clean input point set (543K points) with different parameter settings.

is a real point scan. In Figure 4 we show the results of our method on a V-Shape surface. In this simple example, our L_0 minimization can reconstruct a clean point set denoting a piece-wise smooth surface with accurate normals.

Figure 12 shows our method on a more complex shape and demonstrates how our results evolve as we iterate. As shown in the middle row (2 iterations) our results are piecewise smooth after the Position Denoising phase, but the points may contain some "cross" artifacts near sharp features. After the Edge Recovery phase, these artifacts are removed. The following row shows the same steps after 4 iterations and the bottom shows our results after 5 iterations indicating that our results converge to a piecewise smooth point set with points sampling the sharp features.

We also evaluate the three different stages of our method (normal estimation, point denoising, and edge recovery) individually. For fairness, we use relatively the same neighborhood size for all methods. Figure 5 shows a comparison of estimating point orientations between our L_0 method and existing methods including Bilateral Filtering, RIMLS [32], and Anisotropic WLOP [11]. As the noise level increases, the performance of all of the methods decreases though our L_0 method is fairly resilient even to significant noise.

In Figure 6 we show the effectiveness of our point denoising step. The first row demonstrates the results of Bilateral Filtering and Anisotropic WLOP using normals computed with a local PCA. Below, we show the same methods using the normals from our L_0 normal estimation step. In each case, the result improves. However, Bilateral Filtering suffers from the "cross" artifact and Anisotropic WLOP produces a large gap around the sharp feature. Neither problem appears in our L_0 result.

Finally, Figures 16 and 17 show that our method preserves edges better than Anisotropic WLOP. Using our L_0 denoised clean point set as input, Anisotropic WLOP performs worse in representing the edges with the gaps between smoothing regions getting smaller. UPSAMPLE_EAR on our L_0 denoised point set better preserves the edges than on the Anisotropic WLOP result as well.

Our L_0 minimization produces piece-wise smooth point sets with points lying directly on sharp edges. However, our projection operator produces gaps near sharp features since nearby points are projected onto the edges as shown in Figures 4 and 12. To bridge the gaps near sharp features, we upsample results of L_0 minimization using the second step (UPSAMPLE_EAR) of EAR algorithm [11]. Because our method approximates point orientations more faithfully, upsampling our L_0 results better preserves sharp features than the original EAR. In Figures 9 and 11, the Dodecahedron example and the Trim-Star example are colored by curvature and normal orientation, respectively. We show the results before and after the upsampling process. Compared with the original EAR method, upsampling using our L_0 results produces higher

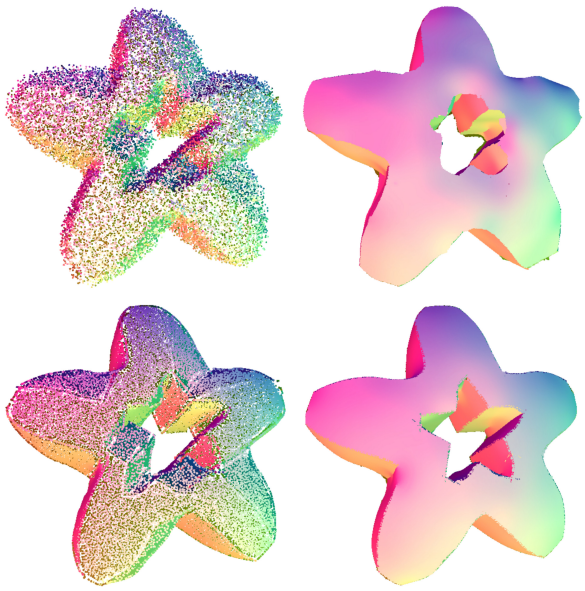

Figure 11: Comparison between EAR and L_0 Minimization. The top row shows the input noisy point cloud and the result of applying original EAR. The bottom row shows the result of L_0 Minimization and UPSAMPLE_EAR on L_0 result.

quality surfaces. In Figure 9 we upsample the point cloud by 250%, while in Figure 11 we upsample the point cloud by 1600%. Figures 13 and 14 show more examples to demonstrate that our method can handle surfaces with or without sharp features.

Figures 2 and 3 show comparisons between our method (L_0 Minimization + UPSAMPLE_EAR) and various state-of-the-art feature aware point cloud denoising methods including APSS [28] + RIMLS [32], WLOP [12], and EAR (AWLOP + UPSAMPLE_EAR) [11]. In each case we used the parameters suggested in the original papers and followed the instructions provided in the authors' code distributions. For our method and EAR, we upsample the point cloud by 250% and 650% in the Dodecahedron example (Figure 2) and the Fandisk example (Figure 3), respectively. In these examples, the WLOP and EAR methods require a large neighborhood size and smoothing parameters, resulting in the poor results in the figures. If the parameters are set larger, small scale features will be smoothed out; while if the parameters are set smaller, the point clouds will still appear noisy. In each example our method provides sharper edges as previous methods either over-smooth sharp features or perform poorly when reconstructing sharp features.

Table 1: Parameter Settings

Parameter	Range (Default)
numNN in Normal Estimation (k)	15 - 35 (20)
numNN in Position Denoising (k)	5 - 15 (10)
numNN in Edge Recovery (k)	4 - 12 (8)
Normal Estimation (η)	0.05 - 0.1 (0.075)
Position Denoising (δ)	0.002-0.008 (0.005)

Parameters. Our algorithm is not very sensitive to its parameters. We list detailed information about the parameters we use in Table 1. In our method, we mainly have five parameters: the size of neighborhood $numNN$ for Normal Estimation, Position Denoising and Edge Recovery as well as the L_0 minimization smoothing parameter for Normal Estimation (η) and Position Denoising (δ). We chose these parameters from a small range starting with the default parameters listed for all of our examples.

In general, more correction is needed to denoise point sets with the presence of large amount of noises

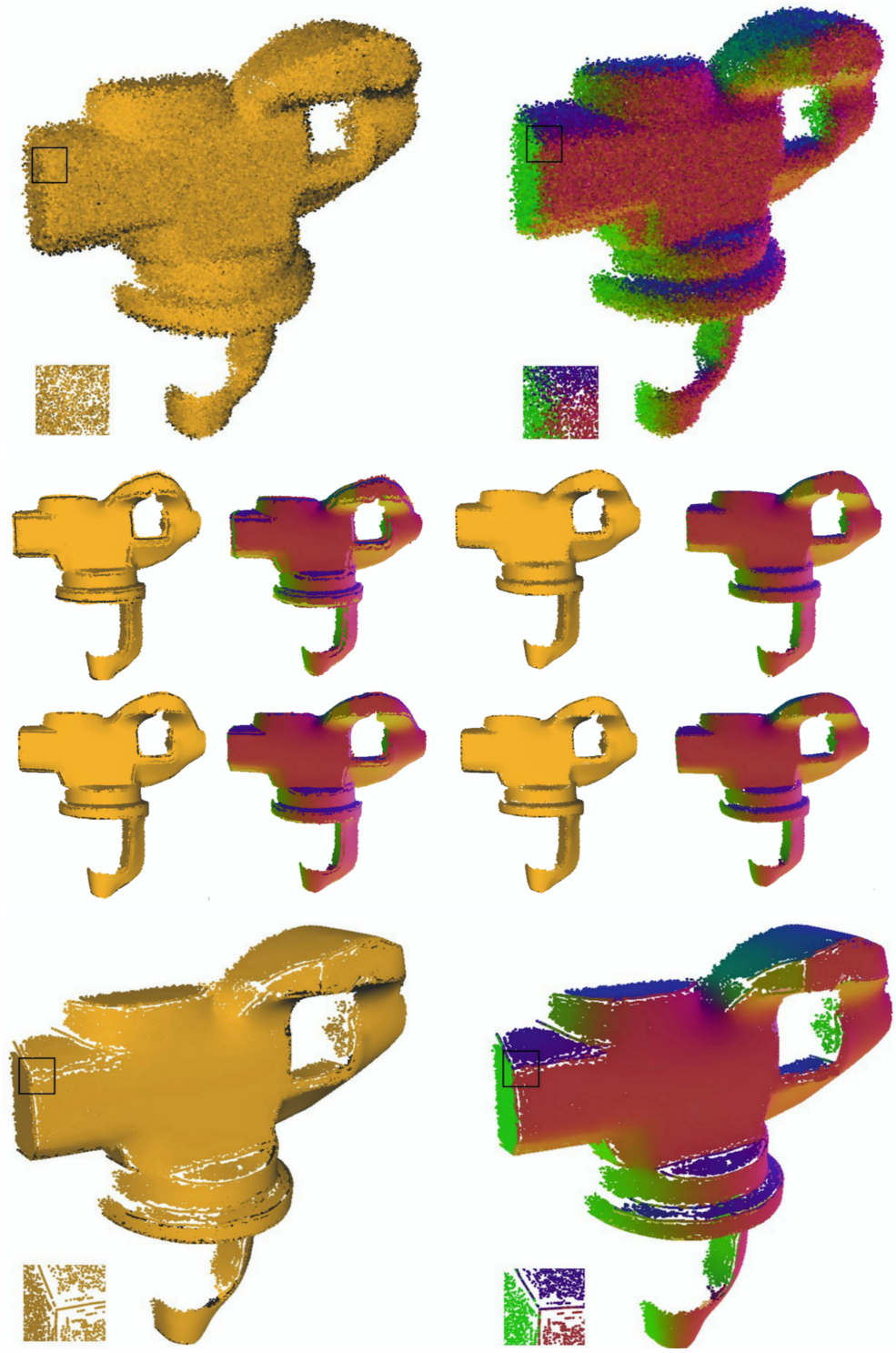

Figure 12: A demonstration of our L_0 Minimization Denoising procedure on a complicated Iron-Vise model (161K points). The first row shows the corrupted input point set, and its corresponding normal colorization. The second row and the third row show the results after 2 and 4 iterations, respectively, from left to right: result after Point Denoising Stage, its corresponding normal colorization, result after Edge Recovery Stage, and its corresponding normal colorization. The last row shows the final result after 5 iterations and its corresponding normal colorization.

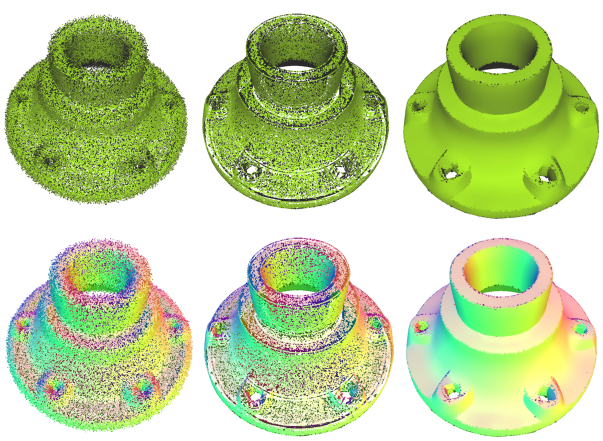

Figure 13: Denoise a model with sharp features. From left to right: Input (125K points), L_0 Minimization, and UPSAMPLE_EAR on L_0 result (764K points).

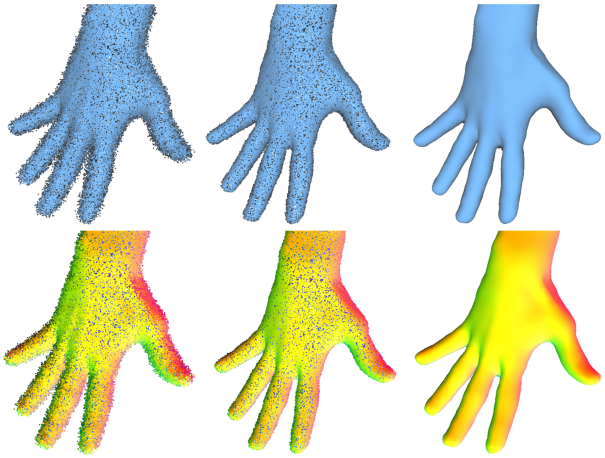

Figure 14: Denoise a model without sharp features. From left to right: Input (59K points), L_0 Minimization, and UPSAMPLE_EAR on L_0 result (143K points).

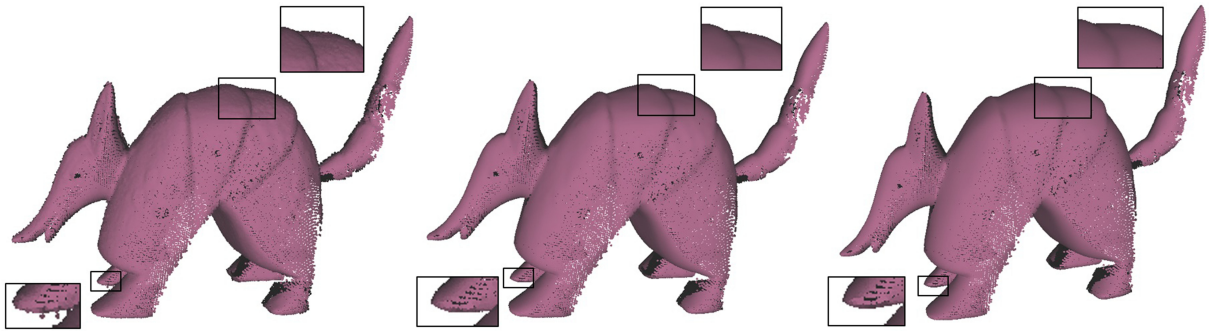

Figure 15: Applying our method to a real point scan, from left to right: Input scanned data (99K Points), L_0 with smaller smoothing parameters, and L_0 with larger smoothing parameters.

and thus larger values will be chosen for η, δ, and $numNN$ while less correction is preferred when more sharpness is required and therefore smaller values of the parameters η, δ, and $numNN$ will be used. As a result, there is a tradeoff between sharpness and amount of noise. Moreover, the neighborhood size $numNN$ needs to be small enough to catch the small-scale features. To produce the results in the paper, we started with the default values of all the parameters and then fine-tuned the parameters based on the presence of noise and sharpness of features. While such tuning helps to yield high-quality results, our method tends to perform well even with the default parameters as shown in Figure 5 where we show results with both default and tuned parameters.

Figure 10 illustrates how the parameters affect our optimization. We applied our method to a clean point cloud with 543k points using different settings for parameters η and δ, which shows that increasing the values of η and δ will gradually remove details. Figure 15 shows a real scanned model processed by our L_0 method with different smoothing parameters.

Convergence and Runtime. Our L_0 Optimization converges quickly. Generally, more iterations are needed for large point clouds or when the noise level is high. For all the models tested in this paper, our optimization converged within 10 iterations with very minor changes to either the positions or normals after 5 iterations. Our tests were run on a 3.16 GHz Intel Xeon X5460. Table 2 shows the number of iterations and running time for all of the models in the paper.

Limitations. Our method cannot handle boundaries well in that we do not produce a clean curve at

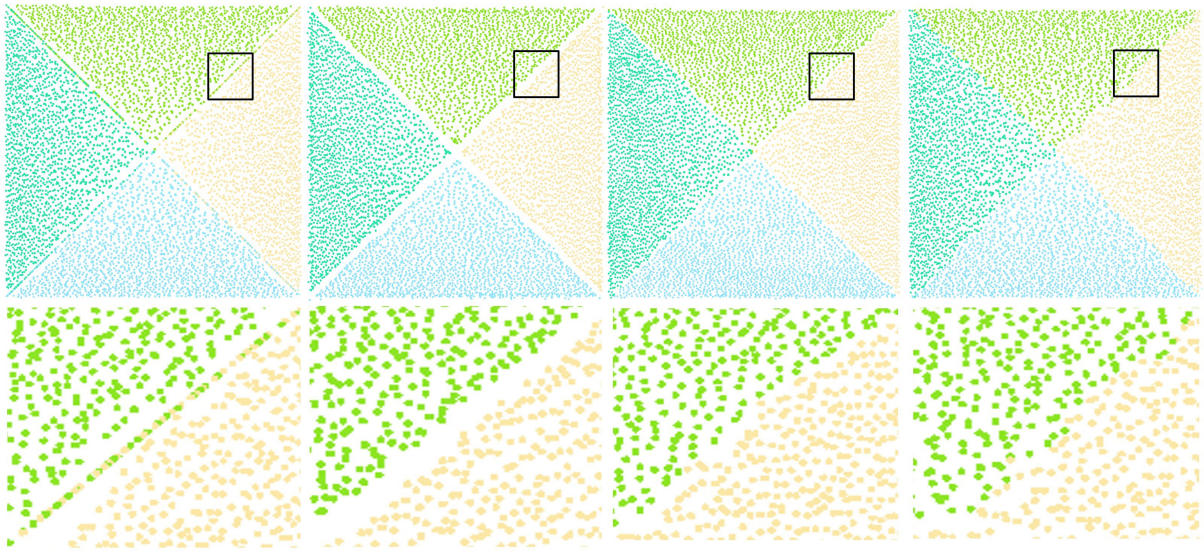

Figure 16: Application of anisotropic LOP (AWLOP) to a L_0 denoised point set. We apply AWLOP with different neighborhood size Δ to a L_0 denoised point set (Top view of a pyramid). First row from left to right: L_0 denoised point set, AWLOP on the L_0 denoised point set with $\Delta = 2$, $\Delta = 1.5$, and $\Delta = 0.5$, respectively. When the gaps between smooth regions become smaller, AWLOP performs worse in terms of representing the edges. The second row shows the corresponding zoomed-in details.

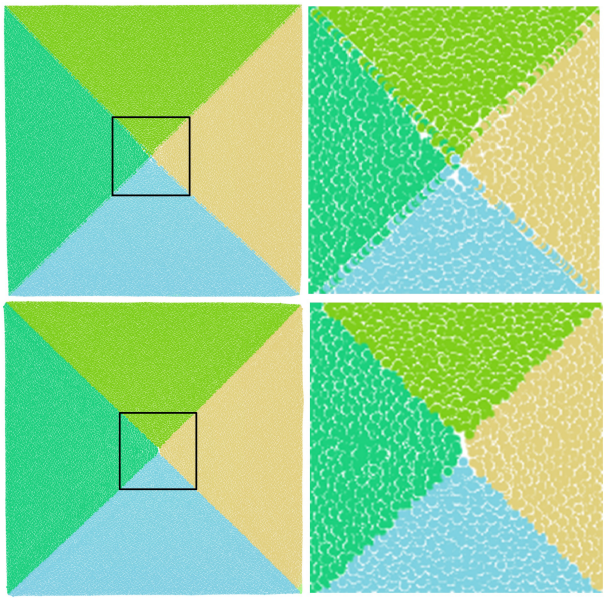

Figure 17: Comparison between UPSAMPLE_EAR on L_0 and on Anisotropic WLOP. First row from left to right: UPSAMPLE_EAR on L_0 and corresponding zoomed-in details. Second row from left to right: UPSAMPLE_EAR on Anisotropic WLOP and corresponding zoomed-in details.

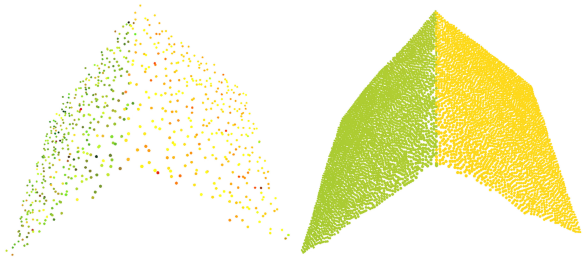

Figure 18: One limitation of our algorithm is that boundaries are not smooth after optimization.

Table 2: Running Times and Optimization Iterations

Model	Points	Iters	Time (Min)
V-Shape Surface	727	1	0.20
Dodecahedron	7,582	3	1.33
Trim-Star	24,402	6	6.58
Fandisk	27,097	5	5.16
Hand	59,485	3	6.07
Armadillo	99,416	3	9.52
Carter	125,804	7	23.17
Iron Vise	161,004	5	21.49
Budda	543,652	2	18.34

boundary edges as shown in Figure 18. In addition, our method may fail when the noise level is extreme. In this case, the initial PCA normal estimation can be so poor that the L_0 norm cannot distinguish features from noises and our results tend to be over-smoothed or over-sharpened in this situation.

6. Future Work and Conclusion

Our method can be improved in several ways. First of all the parameters, including neighborhood size, are currently fixed. We believe that automatically adapting these parameters could improve the performance of our method. Secondly, without the upsampling procedure, our method produces gaps near sharp features. It may be possible to add a repulsion term in the position optimization and projection procedures to distribute points in a more even fashion.

In conclusion, we introduce an efficient L_0-Minimization approach to denoise point sets with sharp features. This denoised point cloud can then be used to further improve the performance of surface reconstruction techniques. In Computer Graphics, a growing body of works aims at enhancing sparsity, and we believe that our approach can help inspire solutions to other problems.

Acknowledgements

We are grateful to all the reviewers for their constructive comments. We would like to thank the authors of [11] to share their real scanned data. We also thank AIM Shape Repository and the Stanford Repository for providing the models used in this paper. This work is supported by National 973 Basic Research Program of China (2011CB302400), National Nature Science Foundation of China (No.61272019) and Science and Technology projects of Shenzhen City (JCYJ20140903112959962).

References

[1] D. Donoho, Compressed sensing, IEEE Transactions on Information Theory 52 (4) (2006) 1289–1306.
[2] E. Candes, J. Romberg, T. Tao, Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information, IEEE Transactions on Information Theory 52 (2) (2006) 489–509.
[3] T. F. Chan, S. Osher, J. Shen, The digital tv filter and nonlinear denoising, Trans. Img. Proc. 10 (2) (2001) 231–241.
[4] A. Levin, R. Fergus, F. Durand, W. T. Freeman, Image and depth from a conventional camera with a coded aperture, ACM Trans. Graph. 26 (3) (2007) 70:1–70:9.
[5] L. I. Rudin, S. Osher, E. Fatemi, Nonlinear total variation based noise removal algorithms, Phys. D 60 (1-4) (1992) 259–268.
[6] R. Wang, Z. Yang, L. Liu, J. Deng, F. Chen, Decoupling noise and features via weighted l1-analysis compressed sensing, ACM Trans. Graph. 33 (2) (2014) 18:1–18:12.
[7] H. Avron, A. Sharf, C. Greif, D. Cohen-Or, L-1 sparse reconstruction of sharp point set surfaces, ACM Trans. Graph. 29 (5) (2010) 135:1–135:12.
[8] L. Xu, C. Lu, Y. Xu, J. Jia, Image smoothing via l0 gradient minimization, ACM Trans. Graph. 30 (6) (2011) 174:1–174:12.

[9] L. Xu, S. Zheng, J. Jia, Unnatural l0 sparse representation for natural image deblurring, in: Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition, CVPR '13, The University of Hong Kong, IEEE Computer Society, Washington, DC, USA, 2013, pp. 1107–1114.
[10] L. He, S. Schaefer, Mesh denoising via l0 minimization, ACM Trans. Graph. 32 (4) (2013) 64:1–64:8.
[11] H. Huang, S. Wu, M. Gong, D. Cohen-Or, U. Ascher, H. R. Zhang, Edge-aware point set resampling, ACM Trans. Graph. 32 (1) (2013) 9:1–9:12.
URL `http://web.siat.ac.cn/~huihuang/EAR/EAR_page.html`
[12] H. Huang, D. Li, H. Zhang, U. Ascher, D. Cohen-Or, Consolidation of unorganized point clouds for surface reconstruction, in: ACM SIGGRAPH Asia 2009 Papers, SIGGRAPH Asia '09, ACM, New York, NY, USA, 2009, pp. 176:1–176:7.
[13] B. Liao, C. Xiao, L. Jin, H. Fu, Efficient feature-preserving local projection operator for geometry reconstruction., Computer-Aided Design 45 (5) (2013) 861–874.
[14] Y. Lipman, D. Cohen-Or, D. Levin, H. Tal-Ezer, Parameterization-free projection for geometry reconstruction, ACM Trans. Graph. 26 (3) (2007) 22:1–22:5.
[15] R. Preiner, O. Mattausch, M. Arikan, R. Pajarola, M. Wimmer, Continuous projection for fast l1 reconstruction, ACM Trans. Graph. 33 (4) (2014) 47:1–47:13.
[16] N. Amenta, M. Bern, M. Kamvysselis, A new voronoi-based surface reconstruction algorithm, in: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '98, ACM, New York, NY, USA, 1998, pp. 415–421.
[17] J.-D. Boissonnat, Geometric structures for three-dimensional shape representation, ACM Trans. Graph. 3 (4) (1984) 266–286.
[18] H. Edelsbrunner, E. P. Mücke, Three-dimensional alpha shapes, ACM Trans. Graph. 13 (1) (1994) 43–72.
[19] C. L. Bajaj, F. Bernardini, G. Xu, Automatic reconstruction of surfaces and scalar fields from 3d scans, in: Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '95, ACM, New York, NY, USA, 1995, pp. 109–118.
[20] J.-D. Boissonnat, F. Cazals, Smooth surface reconstruction via natural neighbour interpolation of distance functions, Comput. Geom. Theory Appl. 22 (1-3) (2002) 185–203.
[21] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle, Surface reconstruction from unorganized points, in: Proceedings of the 19th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '92, ACM, New York, NY, USA, 1992, pp. 71–78.
[22] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, T. R. Evans, Reconstruction and representation of 3d objects with radial basis functions, in: Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '01, ACM, New York, NY, USA, 2001, pp. 67–76.
[23] M. Kazhdan, M. Bolitho, H. Hoppe, Poisson surface reconstruction, in: Proceedings of the fourth Eurographics symposium on Geometry processing, Eurographics Association, 2006, pp. 61–70.
[24] M. Kazhdan, H. Hoppe, Screened poisson surface reconstruction, ACM Trans. Graph. 32 (3) (2013) 29:1–29:13.
[25] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, C. T. Silva, Computing and rendering point set surfaces, IEEE Transactions on Visualization and Computer Graphics 9 (1) (2003) 3–15.
[26] N. Amenta, Y. J. Kil, Defining point-set surfaces, in: ACM SIGGRAPH 2004 Papers, SIGGRAPH '04, ACM, New York, NY, USA, 2004, pp. 264–270.
[27] D. Levin, Mesh-independent surface interpolation, Geometric Modeling for Scientific Visualization (2004) 37–49.
[28] G. Guennebaud, M. Germann, M. Gross, Dynamic sampling and rendering of algebraic point set surfaces, Comput. Graph. Forum 27 (2) (2008) 653–662.
[29] G. Guennebaud, M. Gross, Algebraic point set surfaces, ACM Trans. Graph. 26 (3) (2007) 23:1–23:9.
[30] A. Adamson, M. Alexa, Point-sampled cell complexes, in: ACM SIGGRAPH 2006 Papers, SIGGRAPH '06, ACM, New York, NY, USA, 2006, pp. 671–680.
[31] S. Fleishman, D. Cohen-Or, C. T. Silva, Robust moving least-squares fitting with sharp features, in: ACM SIGGRAPH 2005 Papers, SIGGRAPH '05, ACM, New York, NY, USA, 2005, pp. 544–552.
[32] A. C. Oztireli, G. Guennebaud, M. H. Gross, Feature preserving point set surfaces based on non-linear kernel regression., Comput. Graph. Forum 28 (2) (2009) 493–501.
[33] A. Boulch, R. Marlet, Fast and robust normal estimation for point clouds with sharp features, Comp. Graph. Forum 31 (5) (2012) 1765–1774.
[34] E. Kalogerakis, P. Simari, D. Nowrouzezahrai, K. Singh, Robust statistical estimation of curvature on discretized surfaces, in: Proceedings of the Fifth Eurographics Symposium on Geometry Processing, SGP '07, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2007, pp. 13–22.
[35] B. Li, R. Schnabel, R. Klein, Z. Cheng, G. Dang, J. Shiyao, Robust normal estimation for point clouds with sharp features, Computers & Graphics 34 (2) (2010) 94–106.
[36] Q. Zheng, A. Sharf, G. Wan, Y. Li, N. J. Mitra, D. Cohen-Or, B. Chen, Non-local scan consolidation for 3d urban scenes, in: ACM SIGGRAPH 2010 Papers, SIGGRAPH '10, ACM, 2010, pp. 94:1–94:9.
[37] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, G. Taubin, The ball-pivoting algorithm for surface reconstruction, IEEE Transactions on Visualization and Computer Graphics 5 (4) (1999) 349–359.
[38] I. Guskov, W. Sweldens, P. Schröder, Multiresolution signal processing for meshes, in: Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '99, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1999, pp. 325–334.