

Efficient Point Cloud Skeletonization with Locally Adaptive L_1 -Medial Projection

Stefan Lengauer, Peter Houska and Reinhold Preiner

Institute of Computer Graphics and Knowledge Visualization, Graz University of Technology
Inffeldgasse 16c, 8010 Graz, Austria

s.lengauer@cgv.tugraz.at; p.houska@cgv.tugraz.at; r.preiner@cgv.tugraz.at

ABSTRACT

3D line skeletons are simplistic representations of a shape's topology which are used for a wide variety of geometry-processing tasks, including shape recognition, retrieval, and reconstruction. Numerous methods have been proposed to generate a skeleton from a given 3D shape. While mesh-based methods can exploit existing knowledge about the shape's topology and orientation, point-based techniques often resort to precomputed per-point normals to ensure robustness. In contrast, previously proposed techniques for unprocessed point clouds either exhibit inferior robustness or require expensive operations, which in turn increases computation time. In this paper, we present a new and highly efficient skeletonization approach for raw point cloud data, which produces overall competitive results compared to previous work, while exhibiting much lower computation times. Our algorithm performs robustly in the face of noisy and fragmented inputs, as they are usually obtained from real-world 3D scans. We achieve this by first transferring the input point cloud into a Gaussian mixture model (GMM), obtaining a more compact representation of the surface. Our method then iteratively projects a small subset of the points into local L_1 -medians, yielding a rough outline of the shape's skeleton. Finally, we present a new branch detection technique to obtain a coherent line skeleton from those projected points. We demonstrate the capabilities of our proposed method by extracting the line skeletons of a diverse selection of input shapes and evaluating their visual appearance as well as the efficiency compared to alternative state-of-the-art methods.

Keywords

point cloud, curve skeleton, Gaussian mixture, geometric computation

1 INTRODUCTION

Line skeletons – infinitely thin and centered representations of a shape's topology – have been a heavily researched topic since the original concept definition by Blum [Blu67] in 1967. This compact approximation captures the essence of the topological structure of even highly detailed 3D shapes, and acts as an efficient proxy for applications such as shape identification, classification and retrieval.

3D skeletonization approaches typically require a surface mesh or volumetric mesh as input and extract the line skeletons through shrinking, contracting, projecting or other geometry processing techniques. However, in some cases, e.g. scanning of real world objects, shapes are available only as raw point clouds. Obtaining line skeletons from such inputs is possible but entails additional challenges [OI92; CSM07; HLZ*09]. A skeletonization technique able to process such input data was proposed by Huang *et al.* [HLZ*09]. It builds on the *Locally Optimal Projection* (LOP) operator proposed by Lipman *et al.* [LCLT07], which allows for a parametrization-free resampling of the surface given by a raw input point cloud. Let $P = \{p_j\}_{j \in J} \subset \mathbb{R}^3$ be an unordered set of input points and $X^{(0)} = \{x_i^{(0)}\}_{i \in I} \subset \mathbb{R}^3$

an arbitrary set of projected points with $|X^{(0)}| \ll |P|$ and I, J denoting index sets. While $X^{(0)}$ can contain any points from \mathbb{R}^3 , in practice faster convergence can be achieved with the initialization $X^{(0)} \subset P$. The LOP operator tries to determine the set of optimally projected points X by satisfying

$$\arg \min_X \sum_{i \in I} \sum_{j \in J} \|x_i - p_j\| \theta(\|x_i - p_j\|) + R(X). \quad (1)$$

Here, the first term describes the locally weighted L_1 -median [Web09], driving the points in X towards local centers of P . The repulsion force $R(X)$ prevents projected points from accumulating in clusters. Huang *et al.* [HLZ*09] adopted this operator to account for non-uniform particle distributions with a locally adaptive density weighting, referred to as *Weighted LOP* (WLOP). To this end, they assign weights to each point p_j and x_i , based on the local point cloud density. Preiner *et al.* [PMA*14] show how to reformulate this operator to work on a compact mixture of anisotropic Gaussians \mathcal{M} , representing the input point cloud P . Their projection scheme is referred to as *Continuous LOP* (CLOP). They reformulate the attraction force accordingly, where $|\mathcal{M}| \ll |P|$, thereby greatly enhancing the efficiency of the projection.

While this family of projection operators was originally designed to reconstruct surfaces from noisy and incomplete point clouds, Huang *et al.* [HWC*13] showed that with minimal adaptations WLOP can also be used to project the surface points into local centers of the entire shape, thus constituting the basis for a line skeleton. They use a weighted *Principal Component Analysis* (PCA) to detect point connections and branches. The advantage of this approach is that it has few requirements regarding the quality of the input point set as it is very robust *w.r.t.* outliers. Also, neither a surface mesh nor vertex normals are required.

We present a novel method that combines this WLOP-based skeletonization with the much more efficient CLOP operator that operates on a Gaussian mixture representation of the input data. Moreover, we replace the point-based branch detection technique, proposed by Huang *et al.*, with a more robust probability-based approach. Finally, we show that by decoupling projection and branch detection, a further increase of computational efficiency can be achieved.

2 RELATED WORK

The topic of skeletonization is vast and multi-faceted. Several different strategies have emerged over the years, ranging from *shrinking ball methods* and *distance field methods* to *medial-surface-based methods* and *contraction methods*, each with a large number of publications. Among these methods, various approaches can be also be classified based on the type of input data they process, such as (watertight) surface meshes, voxel grids, or point clouds. In this paper, we primarily focus on skeletonization techniques that process point clouds as inputs. For a general discussion on skeletonization techniques, we refer the reader to a number of surveys. Cornea *et al.* [CSM07] give a broad overview of methods for obtaining curve skeletons. Sobiecki *et al.* [SYJT13] focus specifically on the comparison of contraction-based skeletonization methods, and they also look into the comparison of curve- and surface-skeletons operating on voxel shapes [SJT14]. The most recent survey from 2016 by Tagliasacchi *et al.* [TDS*16] constitutes an encompassing report on all types and aspects of skeletonization.

While the vast majority of approaches requires surface meshes as input, Ogniewicz and Ilg [OI92] show how line skeletons can be obtained from point clouds based on a Voronoi diagram of boundary points. A weakness of this approach is that it requires a uniform sampling of an intact input shape, which is not generally provided by many point clouds, especially when they are obtained by scanning real-world physical objects.

Sharf *et al.* [SLSK07] propose growing a watertight genus-0 mesh inside the input point cloud with multiple competing evolving fronts. While this approach

manages to capture various degrees of detail and is able to cope with missing data, it is computationally expensive. Tagliasacchi *et al.* [TZC09] achieve robustness *w.r.t.* holes in the point cloud by inferring a generalized rotational symmetry axis (ROSA) from the points, which, however, requires normals provided along with the point positions as input. Cao *et al.* [CTO*10] show how skeletons can be obtained from point clouds by pairing a Laplacian-based contraction – as proposed by Au *et al.* [ATC*08] for surface meshes – with a local Delaunay triangulation.

3 GMM-BASED SKELETONIZATION

In this section we discuss our method and its main components in detail. Our method performs a continuous optimal projection (CLOP) (Sec 3.1) with dynamically increasing kernel radii (Sec 3.2). The subsequent branch detection (Sec. 3.3) takes the projected points as input and yields a connected 1D structure capturing the essential geometric shape of these points.

3.1 L_1 -Medial Projection

Based on the condition given in Eqn. (1), Huang *et al.* [HLZ*09] use the following update rule for the projected points $x_i^{(k)} \in X^{(k)}$ at iteration k :

$$x_i^{(k+1)} = F_1(x_i^{(k)}, P) + \mu F_2(x_i^{(k)}, X_i^{(k)} \setminus \{x_i\}), \quad (2)$$

with the attraction force

$$F_1(x, P) = \sum_{j \in J} p_j \frac{\alpha_j}{\sum_{j' \in J} \alpha_{j'}} \quad (3)$$

and the repulsion force

$$F_2(x, X_i) = \sum_{i' \in I \setminus \{i\}} (x - x_{i'}) \frac{\beta_{i'}}{\sum_{i'' \in I \setminus \{i\}} \beta_{i''}}, \quad (4)$$

where

$$\alpha_j = \frac{\theta(\|p_j - x\|)}{\|p_j - x\|} \quad \text{and} \quad (5)$$

$$\beta_{i'} = \frac{\theta(\|x - x_{i'}\|)}{\|x - x_{i'}\|} \left| \frac{\partial \eta}{\partial r}(\|x - x_{i'}\|) \right|. \quad (6)$$

In Eqn. (2), μ governs the balance between the forces and should be within $[0, 0.5)$ for practical applications [HLZ*09; PMA*14]. $\theta(r) = e^{r^2/(h/4)^2}$ is a fast decaying, isotropic kernel function over support radius h . The projection scheme in this form corresponds to the LOP operator by Lipman *et al.*

In order to account for non-uniform densities in the input point cloud, the WLOP operator introduces additional local density weights

$$v_j = 1 + \sum_{j' \in J \setminus \{j\}} \theta(\|p_j - p_{j'}\|), \quad (7)$$

$$w_i^{(k)} = 1 + \sum_{i' \in I \setminus \{i\}} \theta(\|x_i^{(k)} - x_{i'}^{(k)}\|), \quad (8)$$

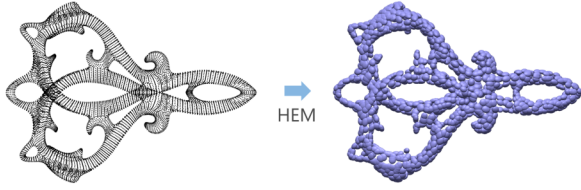


Figure 1: A point cloud is converted to a mixture of anisotropic Gaussians, visualized as blue ellipsoids denoting their 0.5 iso-variance.

acting on Eqn. (5) and Eqn. (6) as $\alpha_j = \alpha_j/v_j$ and $\beta_{i'}^{(k)} = \beta_{i'}^{(k)} w_{i'}^{(k)}$. This adopted scheme results in an improved point regularity and further prevents the forming of point clusters.

With CLOP we approximate the input point set P by a much more compact mixture of anisotropic Gaussians $\mathcal{M} = \{w_j, \mu_j, \Sigma_j\}$, where w_j denotes the Gaussian's convex weights, μ_j their means, and Σ_j their anisotropic covariance matrices. Such a mixture can be computed efficiently by means of a regularized *hierarchical expectation maximization* [VL98; PMA*14]. Accordingly, Eqn. (3) is reformulated to exert the integral attraction force of the continuous density of the mixture \mathcal{M} ,

$$\mathcal{F}_1(q, \mathcal{M}) = \frac{\sum_j w_j \int_{\mathbb{R}^3} x g(x|\mu_j, \Sigma_j) \alpha(x) dx}{\sum_j w_j \int_{\mathbb{R}^3} g(x|\mu_j, \Sigma_j) \alpha(x) dx}, \quad (9)$$

which can be efficiently evaluated in closed form [PMA*14].

3.2 Adaptive Kernel-Growth

Huang *et al.* [HWC*13] propose an interleaved WLOP projection and branch detection scheme in order to obtain a L_1 medial projection of points. To this end, a constantly increasing kernel size h is used to cover features of different diametric extents, while avoiding the degeneration of smaller features in the presence of too large h . During the branch detection step, subsets of the projected points that are found to exhibit a branch-like structure are marked as *fixed* and excluded from the further projection iterations in order to prevent their degeneration.

Opposed to an interleaved execution, we found that the efficiency as well as the robustness of the skeletonization process could be improved by performing the projection and branch detection steps consecutively. That is, we perform CLOP projection until *all* projected points X have converged towards their local L_1 median, and only then, branch detection is performed. As a consequence, we also have to adjust the kernel-growth policy accordingly: Since no points are *fixed* until projection is finished, a globally increasing kernel would destroy the valid medial structures which form at a small h in areas of delicate features. To resolve this issue, we

employ an increasing but locally adaptive kernel size whose growth rate is governed by the local anisotropy of the shape.

Starting from an initial kernel size of $h^{(0)} = 2bbd / \sqrt[3]{|P|}$ (where bbd denotes the bounding box diagonal of P) as proposed by Huang *et al.*, the kernel $h_i^{(k)}$ for point x_i at iteration k is given by

$$h_i^{(k)} = h_i^{(k-1)} + \Delta h v_i^{(k)}, \quad (10)$$

with $\Delta h = h^{(0)}/2$. The local growth factor $v_i^{(k)} \in [0, 1]$ ensures that areas with highly isotropic neighborhood experience a comparably large kernel growth, while regions of highly anisotropic shape experience almost none. The local growth factor is given by

$$v_i^{(k)} = \begin{cases} 1 & \text{if } n_i^{(k)} \leq 2, \\ 1 - \frac{n_i^{(k)} - 2}{n_i^{(k)} - 1} \sigma_i^{(k)} & \text{otherwise.} \end{cases} \quad (11)$$

Here, the measure for local anisotropy $\sigma_i^{(k)} \in [0, 1]$ is given by

$$\sigma_i^{(k)} = \frac{\lambda_{i_2}^{(k)}}{\lambda_{i_0}^{(k)} + \lambda_{i_1}^{(k)} + \lambda_{i_2}^{(k)}}, \quad (12)$$

where $\lambda_{i_0}^{(k)} \leq \lambda_{i_1}^{(k)} \leq \lambda_{i_2}^{(k)}$ are the real-valued eigenvalues of the weighted covariance matrix

$$C_i^{(k)} = \sum_{i' \in I \setminus \{i\}} \theta(\|v_{i,i'}\|) v_{i,i'} v_{i,i'}^\top, \quad v_{i,i'} = x_i^{(k)} - x_{i'}^{(k)}. \quad (13)$$

Note that $\sigma_i^{(k)}$ is scaled relative to the size of x_i 's local neighborhood $n_i^{(k)} = |\{i' \in I \setminus \{i\} : \|x_i^{(k)} - x_{i'}^{(k)}\| < h_i^{(k)}\}|$. This counteracts a known downside of the applied weighted PCA that very sparse neighborhoods could exhibit a very high anisotropy, thus preventing any kernel growth and point projection in the first place.

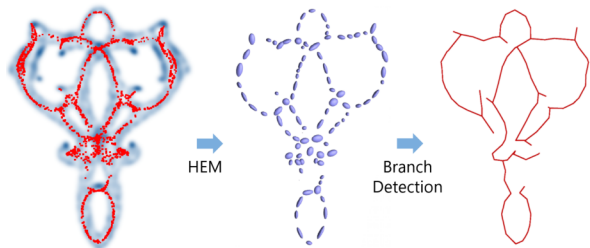


Figure 2: Left: density of \mathcal{M} (blue) and projected skeleton points X (red). Middle: hierarchical EM on X results in a skeleton mixture \mathcal{M}_X . Right: Skeleton after probabilistic branch detection.

3.3 Branch Detection

The iterative CLOP projection (Sec. 3.1) results in a regularized point set X in the local L_1 -medians of the

input shape. Although X resembles the resulting skeleton, the points do not convey any connectivity or branch information. Thus, in a final processing step we extract a set of connected straight line segments constituting the branches of the shape's skeleton.

We circumvent the instabilities we encountered with point-based branch detection approaches [HWC*13] by extracting branches from a probabilistic approximation of the projected points. This way, branch detection is more robust *w.r.t.* outliers and areas with high point density. Similar to the processing of input points P in Section 3.1, we use a regularized hierarchical expectation maximization [PMA*14] to compute a Gaussian mixture $\mathcal{M}_{\mathcal{X}} = \{w_i, (\mu_i, \Sigma_i)\}$ of the final projected points X (Fig. 2, middle). The mean points $\{\mu_i\}$ of the resulting Gaussians are then iteratively connected such that the angle α between adjacent segments (Fig. 3b) does not fall below $\alpha_{min} = \pi/2$, as we assume that the points within a branch are well aligned. Let $M = (d_M(i, j)) \in \mathbb{R}^{|\mathcal{M}| \times |\mathcal{M}|}$, $i, j = 1..|\mathcal{M}|$ denote the matrix of symmetric Mahalanobis distances [Mah36] $d_M(i, j) = \min\{\tilde{d}_M(\mu_i, \mu_j, \Sigma_i), \tilde{d}_M(\mu_j, \mu_i, \Sigma_j)\}$, with $\tilde{d}_M(x_1, x_2, \Sigma) = \sqrt{(x_1 - x_2)^\top \Sigma^{-1} (x_1 - x_2)}$, and let $E = (||\mu_i - \mu_j||)$ denote the equal-sized matrix of Euclidean distances between components' means. Our greedy branch detection method employs these distances as outlined in Algorithm 1, with the minimum number of components per branch $b_{min} = 5$, and the maximum Euclidean distance between components $d_{max} = 0.1\%$ of the bounding box diagonal.

As this approach can result in multiple disjoint branches, we complete this stage by merging these segments to a single connected set of branches (Fig. 3c). To this end, we perform a connected components analysis to identify unconnected clusters. Branches are merged pairwise by inserting a link between the two points from the respective clusters with the lowest Mahalanobis distance d_M . This merging is conducted iteratively until only one cluster remains, which constitutes the final skeleton (Fig. 2, right).

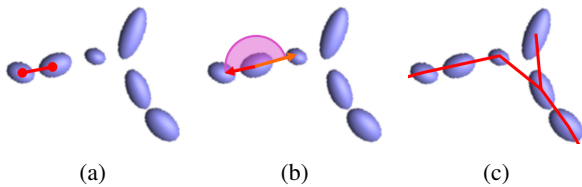


Figure 3: Starting from a seed pair of mixture components with minimal distance (a), an angle criterion (b) is used to grow the branch in both directions. Finally all branches are merged (c) to obtain a fully connected line skeleton.

4 RESULTS

We compare our method to five alternative skeletonization methods (Sec. 4.1), both mesh-based and point-

Algorithm 1 Branch detection based on pair-wise Mahalanobis distances M and Euclidean distances E as well as governing thresholds α_{min} , d_{max} and b_{min} .

```

1: function BRANCHDETECTION( $M, E, \alpha_{min}, d_{max}, b_{min}$ )
2:    $B \leftarrow \{\}$  ▷ The set of branches
3:    $A \leftarrow 1..|M_{0*}|$  ▷ The index set
4:    $v \leftarrow \{\}$  ▷ The blacklist of used points
5:    $(\hat{i}, \hat{j}) \leftarrow \arg \min_{i \in A, j \in A} M_{ij}$ 
6:   while  $|A| \geq b_{min}$  and  $E_{\hat{i}\hat{j}} < d_{max}$  do
7:      $b \leftarrow \langle \hat{i}, \hat{j} \rangle$  ▷ Init new branch
8:      $prev \leftarrow \arg \min_{i \in A \setminus b} (M_{ib_0})$ 
9:     while  $\angle(b_0 b_1, b_0 prev) > \alpha_{min}$  do
10:       $b \leftarrow \langle prev, b \rangle$  ▷ Grow head
11:       $prev \leftarrow \arg \min_{i \in A \setminus b} (M_{ib_0})$ 
12:    end while
13:     $next \leftarrow \arg \min_{i \in A \setminus b} (M_{ib_{|b|-1}})$ 
14:    while  $\angle(b_{|b|-1} b_{|b|-2}, b_{|b|-1} next) > \alpha_{min}$  do
15:       $b \leftarrow \langle b, next \rangle$  ▷ Grow tail
16:       $next \leftarrow \arg \min_{i \in A \setminus b} (M_{ib_{|b|-1}})$ 
17:    end while
18:    if  $|b| \geq b_{min}$  then
19:       $B \leftarrow B \cup \{b\}$  ▷ Append to branches
20:    end if
21:     $A \leftarrow A \setminus b$  ▷ Update index set
22:     $(\hat{i}, \hat{j}) \leftarrow \arg \min_{i \in A, j \in A} M_{ij}$  ▷ Update start
23:  end while
24:  return  $B$ 
25: end function

```

based approaches, and evaluate the obtained results regarding their visual appearance and capability to capture the essence of a shape (Sec. 4.2), their computational efficiency (Sec. 4.3) and their robustness *w.r.t.* noise and outliers (Sec. 4.4).

4.1 Reference Methods

We compare our method to the original LOP-based approach (L1-WLOP) by Huang *et al.* [HWC*13], the ROSA approach by Tagliasacchi *et al.* [TZC09], and the Laplacian-based contraction of point clouds (Lap-Con10) by Cao *et al.* [CTO*10]. The former is given as a C++ implementation, while the latter two are available as MATLAB scripts. Besides these point-based methods we additionally compare our method to several mesh-based approaches (Wave Fronts, Geometry Contraction, TEASAR and Tangent Ball). Those are provided within a Python library¹ by Au *et al.* [ATC*08], which we also employ to preprocess the inputs and fix common mesh deficiencies, like duplicate or unreferenced vertices, degenerated faces and the like.

Wave Fronts

This method requires a surface mesh as input. Starting from a randomly selected seed vertex, a wave is prop-

¹ <https://navis-org.github.io/skeletor/>

agated across the mesh based on a given n -ring neighborhood, where n defines the propagation step. Vertices that are hit by the wave at the same step are considered rings and are subsequently collapsed to their common center. This approach is particularly well-suited and highly efficient for tubular structures.

Geometry Contraction

Au *et al.* [ATC*08] describe a skeletonization method based on an iterative geometric contraction (henceforth referred to as *LapCon08VC*) of the surface mesh. To this end, the mesh is subjected to an implicit Laplacian smoothing [SCL*04] until the mesh converges to a zero-volume skeleton-like shape. The remaining surface topology is then transformed to a one dimensional curve using a connectivity surgery process. Contraction weights control the efficiency of contraction, while attraction weights dynamically adapt to the local neighborhood. We used the contraction parameter $s_L = 2.0$ recommended by the authors. For inputs where these result in numerical instabilities, a smaller value was used. We also applied a sufficiently large convergence threshold $\epsilon_{vol} = 0.1$ which lets the contraction stop if the contracted mesh is reduced to less than 10% of its original volume. To obtain a line skeleton from the contracted mesh, a subsequent vertex clustering algorithm is applied which joins vertices within a maximum geodesic distance of $s_{d_{max}}$. As the value of this parameter recommended by the authors resulted in instabilities on several of our inputs, we have increased this parameter for those cases.

Besides vertex clustering, Au *et al.* propose a second, alternative connectivity surgery method (*LapCon08EC*). Here, edges are iteratively collapsed in a greedy fashion based on a cost function that follows established mesh geometry preservation metrics [GH97]. An advantage of this approach is that it can be directly applied without the computational expensive prior contraction of the input (as we do in our experiments), while on the downside, it is more sensitive to the scale of the input.

TEASAR

The TEASAR algorithm by Sato *et al.* [SBB*00] starts from a randomly determined root voxel V_0 of a volumetric input mesh, and determines the voxel that maximizes the shortest-path length to V_0 , invalidating all voxels within a given distance along its path. Next, the second longest shortest-path between still valid voxels is picked up repeatedly until all voxels are invalidated. This approach is well suited for shapes with an underlying tree structure and tubular sections like vessels, rib cages and the like. Although the original approach is

conceptualized for volumetric meshes, the only available code is an adaption by Dorkenwald *et al.* [DS22], operating on surface meshes. Note however, that this adaption produces skeletons that lie on the surface of the input shape.

Tangent Ball
















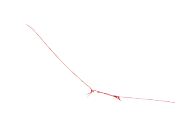

















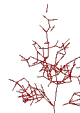
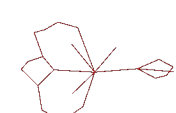













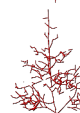

With their medial axis point approximation method (henceforth referred to by *Tangent Ball*) Ma *et al.* [MBC12] leverage vertex normals to determine the line skeleton. A set of randomly determined seed points are used to generate spheres, whose center points are located along their inverse vertex normal axis, such that the respective seed points rest on their sphere surface. The spheres' radii are decreased iteratively via nearest neighbor queries until they contain no other sampling point than their respective seed point. The authors show that for an infinite number of seed points, the sphere centers converge towards the true medial axis. The method is highly sensitive *w.r.t.* the quality of vertex normals and works generally well for smooth surfaces, while errors in the mesh and noisy surfaces pose a limitation.

4.2 Visual Comparison

In order to conduct a qualitative evaluation of our skeletonization approach, we compiled a diverse set of input shapes, posing different challenges to skeletonization: (i) a *Dino* shape which is a simple standard model of a creature with a finite number of extremities, (ii) a statue of *Neptune*, a humanoid figure with several high-frequency structures like fingers, hair and a trident, (iii) an abstract smooth model of *Dancers*, (iv) a *Tree* model comprising a large number of thin details, and (v) a topologically complex *Filigree* model of genus 10. The inputs, as well as their number of vertices, faces and mixtures, are presented in Fig. 4, top row. All models have been skeletonized using the reference skeletonization methods listed in Sec. 4.1.

For the *Dino* shape (Fig. 4, first column) the Wave Fronts, TEASAR, Tangent Ball, ROSA, LapCon10, L1-WLOP, and our method result in plausible line skeletons with clearly distinguishable extremities like legs, tail and neck. The LapCon08EC and LapCon08VC approaches lead to an oversimplification.

For the *Neptune* model (Fig. 4, second column) all methods yield plausible results. However, in contrast to the *Dino* model, the LapCon08EC skeleton exhibits various superfluous branches. The same can be observed for the result generated by the TEASAR method in the torso region of the model. Note that LapCon08VC, ROSA, LapCon10, L1-WLOP and our method only partially preserve the thin tip of the trident.

Input Shapes	Req.	Dino	Neptune	Dancers	Tree	Filigree
	Surface Mesh Volumetric Mesh Vertex Normals	 (21K, 41K, 2354)	 (28K, 56K, 3472)	 (4K, 8K, 687)	 (51K, 84K, 5678)	 (20K, 41K, 1331)
Wave Fronts	● ○ ○					
LapCon08EC	● ○ ○					
LapCon08VC	● ○ ○					
TEASAR	● ● ○ †					
Tangent Ball	○ ○ ●					
ROSA	○ ○ ●					
LapCon10	○ ○ ○					
L1-WLOP	○ ○ ○					
Our method	○ ○ ○					

† The original concept requires a volumetric mesh, but the given results were obtained with an adaptation for surface meshes.

Figure 4: Qualitative comparison of our method to eight related approaches using five diverse input shapes. The numbers in the table header denote the models' number of vertices, faces and mixture components after conversion to a GMM. Filled circles on the left indicate the type of input data and attributes required by a specific approach.

The *Dancers* model (Fig. 4, third column) poses a serious challenge to all of the applied algorithms, as none of them are able to capture the genus of the input shape within their resulting skeleton. Nonetheless, all results appear to be plausible, and one could argue that the

skeleton calculated by L1-WLOP comes closest to the desired result.

The *Tree* shape (Fig. 4, fourth column) yields very diverse results. While the Wave Fronts, LapCon08EC and TEASAR methods obtain an overly detailed skeleton

with a branch for every leaf, the LapCon08VC, Tangent Ball, and L1-WLOP methods return an oversimplified skeleton. ROSA, LapCon10 and our method provide a good trade off and are able to capture the stem as well as the thicker tree branches. Note that the input mesh is comprised of two intersecting but unconnected components (lower stem and tree crown). LapCon10 and our method are the only ones which are able to return a single connected skeleton component for this input.

The *Filigree* model (Fig. 4, fifth column) also exhibits tubular structures of higher genus. Similar to the Dancers model, none of the algorithms is able to capture the shape’s essential structure accurately, and the achieved skeleton quality of the individual algorithms is comparable to the skeletons calculated from the Dancers model.

4.3 Efficiency

Table 1 lists the skeleton computation times for all presented methods and models. All computations were performed on commodity hardware. Note that our method was implemented in C++, while the reference methods were provided as either Python scripts, Matlab scripts or C++ implementations.

Table 1: Computation times for different models and approaches in seconds.

Model	Dino	Neptune	Dancers	Tree	Filigree
Wave Fronts	0.16	0.22	0.04	1.25	0.21
LapCon08EC	42.28	80.23	4.32	173.12	52.62
LapCon08VC	100.13	207.93	10.49	226.36	123.01
Contraction [†]	99.55	206.96	10.32	221.06	122.32
Clustering	0.59	0.97	0.17	5.30	0.69
TEASAR	0.45	1051.85	0.66	1.18	0.99
Tangent Ball	39.70	7.50	0.42	6.18	4.17
ROSA	5101.80	5479.90	7308.00	2236.60	411.29
LapCon10	195.00	525.00	1168.00	977.00	252.00
L1-WLOP	40.00	50.00	161.00	51.00	12.00
Our Method	8.37	13.54	12.98	17.63	3.89
Init [†]	1.93	3.06	5.99	7.55	1.26
HEM [‡]	0.54	1.05	5.41	2.11	0.69
Projection	5.67	9.17	1.49	5.93	1.81
Branching [‡]	0.23	0.26	0.09	2.05	0.12

[†] Mixture computation of the input point set.

[‡] Mixture computation of the projected points plus branch detection.

^{*} Times can vary significantly, depending on parameterization.

From Table 1 we can observe a correlation of efficiency between model sizes and computation times. In particular, the skeletonization of the *Tree* model takes significantly longer with almost all approaches. We can also see that the computation times with different methods vary greatly. LapCon08VC, Tangent Ball and LapCon10 are generally slower, with computation times up to a few minutes. In general, the ROSA technique exhibited the slowest performance, with timings far outside the comparable range for most of the models. Note that the bottleneck of the LapCon08VC method is given by the contraction step, whose efficiency is very sensitive *w.r.t.* the governing parameters. The authors state that these techniques could be orders of magnitude faster with optimal parameterization [ATC*08], which

we aimed to find manually to the best of our means. In contrast, the Wave Fronts and TEASAR methods exhibit very short computation times. As an exception, the skeletonization of the Neptune model with TEASAR took more than 17 minutes on our reference platform.

Our method aligns itself in-between the fast and the slow reference methods with computation times settled around several seconds. As it can be seen from the performance break-down, our computation time is mostly composed of the computation of a Gaussian mixture of input points, necessary for CLOP (Sec. 3.1) and the iterative projection (Sec. 3.2). Interestingly, we can observe that the computation time with our approach does not significantly increase with increasing model size and complexity (c.f. Tree model), as the number of mixture components does not increase equally. For a proper interpretation of the presented timings, note that our method receives a raw non-parameterized point cloud as input, while most reference methods receive a fully connected surface mesh or point clouds with pre-processed vertex normals. If the actual input of those methods were given by raw points cloud as well, additional computation times for normal and surface reconstruction would have to be factored in. These timings were omitted in our comparison, since the input shapes were already given as surface meshes and reconstruction timings are themselves known to depend heavily on the applied method and parameters.

4.4 Stability

In most practical applications of line skeletonization it is generally required that similar input shapes result in similar skeletons. However, it has been shown that even very small perturbations on the surface of an input shape can lead to a substantially different skeleton [TDS*16]. This phenomenon has been referred to as *skeletal noise* [RVT08] or *spurious points* [SBTZ02]. Since 3D models captured from real world objects typically suffer from a multitude of different errors [BLN*13], the robustness of a skeletonization algorithm *w.r.t.* surface perturbations is a major quality characteristic.

We analyze the stability of our method by looking at the Filigree model, as it exhibits a multigenus geometry with structures and features of varying sizes. We subject the model to different levels of noise in order to mimic the artefacts present on models obtained by scanning real-world physical objects. In terms of noise, we apply white noise following the Gaussian distribution $\mathcal{N}(0, \sigma^2)$ to all of the vertices. Different magnitudes σ^2 of noise, 0.002, 0.005, and 0.01 times the object’s bounding box diagonal, have been investigated, in the following referred to as $\mathcal{N}_{0.002}$, $\mathcal{N}_{0.005}$ and $\mathcal{N}_{0.01}$. The resulting noisy inputs are shown in the top row of Fig. 5. Rows 2 to 7 of Fig. 5 present the skeletons resulting from our method as well as the reference methods

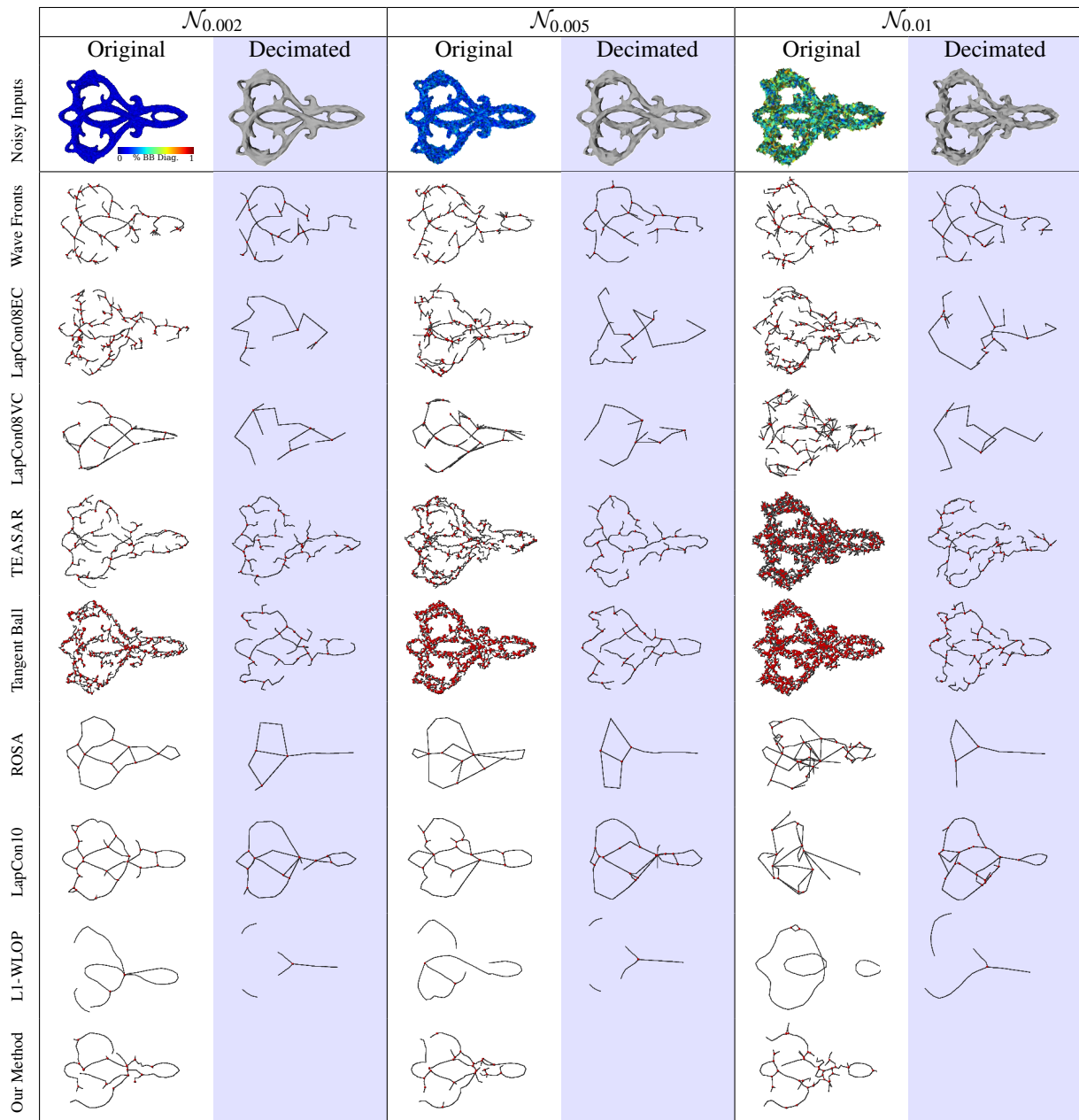


Figure 5: Top: Filigree model subjected to various levels of noise and (shaded in blue) additional decimation step. Rows 2-7: Skeletonization with different methods based on the respective noisy (and decimated) inputs. The red dots in the line skeletons mark junction points.

(Sec. 4.1), based on the different noised inputs. It is immediately visible that the results degenerate with increasing levels of noise, independent from the applied method. Moreover, we can observe clear differences in the quality of the approaches. The LapCon08EC, TEASAR and especially the Tangent Ball methods exhibit a major increase in skeletal noise, even with visually inconspicuous surface perturbations. The skeletons obtained with the Wave Fronts and Iterative Contraction methods do not significantly differ from the results obtained with the noiseless model (Fig. 4). With our method no significant deterioration is apparent at the

lower two noise levels $\mathcal{N}_{0.002}$ and $\mathcal{N}_{0.005}$, and only at level $\mathcal{N}_{0.01}$ slight perturbations become visible.

Additionally to this visual comparison, we provide a quantitative evaluation of robustness. To this end, we count the number of junction points (points where branches are joined) of the resulting skeletons as an indicator for skeletal noise. For the *Filigree* model we expect around 20 junction points as visualized in Fig. 6, left. The figure shows the number of junction points resulting from different algorithms and noise levels. The comparably good performance of our method is attributed to the fact that the Gaussian mixtures,

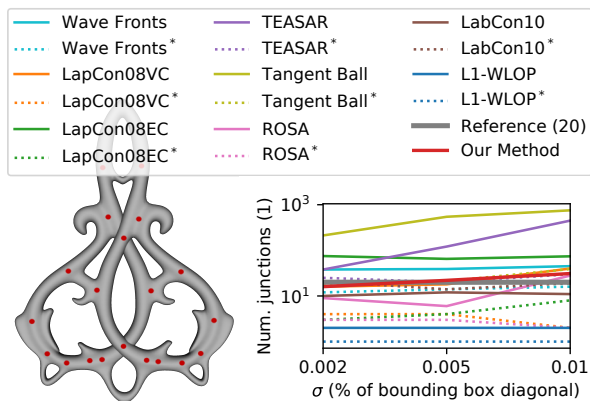


Figure 6: Number of junction points obtained with different approaches and different levels of additive noise. Algorithms marked with an asterisk are based on the decimated inputs.

which the input points are converted to (Sec. 3.1), allow to automatically model the inherent noise level of the data, thereby providing an implicit pre-filtering of the data. To investigate the impact of pre-filtering on the results obtained with the comparison methods, we apply a variant of the *Quadratic Edge Collapse Decimation* by Garland *et al.* [GH98] to the noisy inputs. We choose the parameters such that the resulting number of vertices is equal to the number of mixture components, which do not vary greatly across noise levels. The decimated models can be seen in Fig. 5, top row, shaded in blue. The results for the different methods based on these decimated inputs are given in Fig. 5, rows 2-7. While results became worse with the LapCon08EC and Iterative Contraction skeletonizations, an improvement can be clearly observed for the Wave Fronts, TEASAR and Tangent Ball approaches, where the skeletons obtained with the decimated noisy input is even better than with the unperturbed original input (Fig. 4).

5 LIMITATIONS AND FUTURE WORK

Like many other techniques, the results of the proposed skeletonization approach can be sensitive *w.r.t.* governing parameters. That is, empirically determined parameters are required in three separate steps of our pipeline: (i) the computation of the Gaussian mixture for CLOP and balancing of attraction and repulsion forces (Sec. 3.1), (ii) the speed of the adaptive kernel growth (Sec. 3.2), and (iii) the thresholds governing the branch detection (Sec. 3.3). Some of these parameters could possibly be defined over properties intrinsic to the input point cloud (diameter, density, feature sizes) which can be part of future work.

The PCA-driven kernel growth (Sec. 3.2) and repulsion force (Eqn. (2)) face typical issues of PCA, which are also discussed in Huang *et al.* [HLZ*09]: erroneous

directivity estimates near thick point clouds and unwanted propagation between particles at close-by surfaces. The latter can be observed at the trident of the Neptune model (Fig. 4) where the relative proximity of the three parallel prongs leads to their collapse to a single tubular structure. This poses a severe disadvantage compared to mesh-based approaches, which can rely on the surface topology to avoid the merging of close-by surfaces.

Another aspect not investigated is the determination of the appropriate number of projection steps necessary for a successful projection for all shape granularities. 20 to 30 iterations turned out to be a good choice for all inputs. Introducing an automatic halting criterion would increase the flexibility of our method. A naive approach is to stop the projection if the overall points movement $m^{(k)} = \sum_{i \in I} \|x_i^{(k+1)} - x_i^{(k)}\|$, falls below a certain threshold. In practice however, m does not gradually decrease with k , but instead can level off and run into a local minimum, or even continue to decrease after two surface branches merge. Finding a simple yet robust halting criterion in the face of these complex processes is thus still open for further investigation.

6 CONCLUSION

We present an efficient projection-based point cloud skeletonization approach. To this end, we marry the concept of the WLOP-based skeletonization by Huang *et al.* [HWC*13] with the CLOP operator [PMA*14]. In contrast to Huang *et al.*, we grow projection kernels for each point individually, based on a local anisotropy measure, and conduct the branch detection after the projection instead of interleaved. We present a probabilistic branch detection approach for the projected point set, which is more robust *w.r.t.* outliers. While point-based approaches generally seem to be inferior to surface or even volume-based approaches – as mesh topologies incorporate much vital information – the strength of our approach is that it does not depend on high quality or manifold input data, and even performs robustly in the face of noisy and incomplete data. Hence, we think that our approach is a reasonable alternative in cases where just the raw point cloud data is available, e.g. in the case of 3D scans if no surface reconstruction is conducted. We have demonstrated the feasibility of our method by different experiments conducted on a wide variety of 3D shapes, and analysed limitations and problem cases to indicate directions for future work.

REFERENCES

- [ATC*08] AU, OSCAR KIN-CHUNG, TAI, CHIEW-LAN, CHU, HUNG-KUO, *et al.* “Skeleton Extraction by Mesh Contraction”. *ACM Trans. Graph.* 27.3 (2008) 2, 4, 5, 7.

- [BLN*13] BERGER, MATTHEW, LEVINE, JOSHUA A., NONATO, LUIS GUSTAVO, et al. "A Benchmark for Surface Reconstruction". *ACM Trans. Graph.* 32.2 (Apr. 2013). DOI: 10 . 1145 / 2451236.2451246 7.
- [Blu67] BLUM, HARRY. *Models for the Perception of Speech and Visual Form*. MIT Press, 1967, 362–380 1.
- [CSM07] CORNEA, NICU D., SILVER, DEBORAH, and MIN, PATRICK. "Curve-skeleton properties, applications, and algorithms". *IEEE Transactions on visualization and computer graphics* 13.3 (2007), 530 1, 2.
- [CTO*10] CAO, JUNJIE, TAGLIASACCHI, ANDREA, OLSON, MATT, et al. "Point cloud skeletons via laplacian based contraction". *2010 Shape Modeling International Conference*. IEEE. 2010, 187–197 2, 4.
- [DS22] DORKENWALD, SVEN and SCHNEIDER-MIZELL, CASEY. *MeshParty*. <https://github.com/sdorkenw/MeshParty>. 2022 5.
- [GH97] GARLAND, MICHAEL and HECKBERT, PAUL S. "Surface simplification using quadric error metrics". *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. 1997, 209–216 5.
- [GH98] GARLAND, MICHAEL and HECKBERT, PAUL S. "Simplifying Surfaces with Color and Texture Using Quadric Error Metrics". *Proceedings of the Conference on Visualization '98*. VIS '98. Research Triangle Park, North Carolina, USA: IEEE Computer Society Press, 1998, 263–269. ISBN: 1581131062 9.
- [HLZ*09] HUANG, HUI, LI, DAN, ZHANG, HAO, et al. "Consolidation of Unorganized Point Clouds for Surface Reconstruction". *ACM Trans. Graph.* 28.5 (Dec. 2009), 1–7. DOI: 10 . 1145 / 1618452.1618522 1, 2, 9.
- [HWC*13] HUANG, HUI, WU, SHIHAO, COHEN-OR, DANIEL, et al. "L1-Medial Skeleton of Point Cloud". *ACM Trans. Graph.* 32.4 (July 2013). DOI: 10 . 1145 / 2461912.2461913 2–4, 9.
- [LCLT07] LIPMAN, YARON, COHEN-OR, DANIEL, LEVIN, DAVID, and TAL-EZER, HILLEL. "Parameterization-Free Projection for Geometry Reconstruction". *ACM Trans. Graph.* 26.3 (July 2007), 22–es. DOI: 10 . 1145 / 1276377.1276405 1.
- [Mah36] MAHALANOBIS, PRASANTA CHANDRA. "On the generalized distance in statistics". National Institute of Science of India. 1936 4.
- [MBC12] MA, JAEHWAN, BAE, SANG WON, and CHOI, SUNGHEE. "3D medial axis point approximation using nearest neighbors and the normal field". *The Visual Computer* 28.1 (2012), 7–19 5.
- [OI92] OGNIEWICZ, ROBERT L and ILG, MARKUS. "Voronoi skeletons: theory and applications." *CVPR*. Vol. 92. 1992, 63–69 1, 2.
- [PMA*14] PREINER, REINHOLD, MATTAUSCH, OLIVER, ARIKAN, MURAT, et al. "Continuous Projection for Fast L_1 Reconstruction". *ACM Trans. Graph.* 33.4 (July 2014). DOI: 10 . 1145 / 2601097.2601172 1–4, 9.
- [RVT08] RENIERS, DENNIE, VAN WIJK, JARKE, and TELEA, ALEXANDRU. "Computing multiscale curve and surface skeletons of genus 0 shapes using a global importance measure". *IEEE Transactions on Visualization and Computer Graphics* 14.2 (2008), 355–368 7.
- [SBB*00] SATO, MIE, BITTER, INGMAR, BENDER, MICHAEL A, et al. "TEASAR: Tree-structure extraction algorithm for accurate and robust skeletons". *Proceedings the Eighth Pacific Conference on Computer Graphics and Applications*. IEEE. 2000, 281–449 5.
- [SBTZ02] SIDDIQI, KALEEM, BOUIX, SYLVAIN, TANNENBAUM, ALLEN, and ZUCKER, STEVEN W. "Hamilton-Jacobi Skeletons". *Int. J. Comput. Vision* 48.3 (July 2002), 215–231. DOI: 10 . 1023/A:1016376116653 7.
- [SCL*04] SORKINE, OLGA, COHEN-OR, DANIEL, LIPMAN, YARON, et al. "Laplacian surface editing". *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*. 2004, 175–184 5.
- [SJT14] SOBIECKI, ANDRÉ, JALBA, ANDREI, and TELEA, ALEXANDRU. "Comparison of curve and surface skeletonization methods for voxel shapes". *Pattern Recognition Letters* 47 (2014), 147–156 2.
- [SLSK07] SHARF, ANDREI, LEWINER, THOMAS, SHAMIR, ARIEL, and KOBBELT, LEIF. "On-the-fly Curve-skeleton Computation for 3D Shapes". *Computer Graphics Forum*. Vol. 26. 3. Wiley Online Library. 2007, 323–328 2.
- [SYJT13] SOBIECKI, ANDRÉ, YASAN, HALUK C, JALBA, ANDREI C, and TELEA, ALEXANDRU C. "Qualitative comparison of contraction-based curve skeletonization methods". *International Symposium on Mathematical Morphology and Its Applications to Signal and Image Processing*. Springer. 2013, 425–439 2.
- [TDS*16] TAGLIASACCHI, ANDREA, DELAME, THOMAS, SPAGNUOLO, MICHELA, et al. "3d skeletons: A state-of-the-art report". *Computer Graphics Forum*. Vol. 35. 2. Wiley Online Library. 2016, 573–597 2, 7.
- [TZC09] TAGLIASACCHI, ANDREA, ZHANG, HAO, and COHEN-OR, DANIEL. "Curve skeleton extraction from incomplete point cloud". *ACM SIGGRAPH 2009 papers*. 2009, 1–9 2, 4.
- [VL98] VASCONCELOS, NUNO and LIPMAN, ANDREW. "Learning Mixture Hierarchies". *Proceedings of the 11th International Conference on Neural Information Processing Systems*. NIPS'98. Denver, CO: MIT Press, 1998, 606–612 3.
- [Web09] WEBER, ALFRED. *Ueber den Standort der Industrien*. Mohr, 1909. ISBN: 9785880850846 1.