Fast Non-Convex Hull Computation

Julián Bayardo Spadafora
Universidad de Buenos Aires
Buenos Aires, Argentina
jbayardo@dc.uba.ar

Francisco Gómez-Fernandez
Universidad de Buenos Aires
Buenos Aires, Argentina
fgomez@dc.uba.ar

Gabriel Taubin
Brown University
Providence, RI, USA
taubin@brown.edu

Abstract

3D surface reconstruction usually begins with a point cloud and aims to build a representation of the object producing that point cloud. There are several algorithms to solve this problem, each with different priors over the point cloud, such as the type of object represented, or the method by which it was obtained. In this work, we focus on an algorithm called Non-Convex Hull (NCH), which reconstructs surfaces through a concept similar to the Medial Axis Transform. A new algorithm called Shrinking Planes is proposed to compute the NCH, based on the Shrinking Ball method with a few improvements. We prove that the new method can approximate surfaces to arbitrarily small error, and evaluate its performance on the surface reconstruction task. The new method maintains the same reconstruction quality as the Naïve Non-Convex Hull method, while achieving a large performance improvement.

1. Introduction

3D objects are often represented as polygon meshes. This representation is chosen because it is very efficient for most common operations in computer graphics applications. In 3D scanning, an unstructured point cloud is obtained as a finite set of points $\mathcal{P} = \{\mathbf{p_1}, \dots, \mathbf{p_n}\} \subset \mathbb{R}^3$ that lie in the boundary $\partial\mathcal{S}$ of the object's surface $\mathcal{S}$. These have peculiarities owed to the specific scanning methodology and the object being scanned. The points are then processed into a polygon mesh via surface reconstruction algorithms, which is what we are going to be concerned with in this work. As will be evident later on, reconstruction is a hard problem, and its many complexities lead algorithms to be difficult to understand, code and parallelize.

The Medial Axis Transform (MAT) is a shape representation due to [6], in which objects are given as a union of balls contained within it, each one called a Medial Atom (MA). The MAT is well-structured and holds useful information about the surface, which is why it has been extensively used for a variety of purposes: surface reconstruction [3], shape simplification [29] [4], point cloud simplification [20] [24], and skeletonization [20] [15], among others.

Non-Convex Hull (NCH) Surface Reconstruction [31] is a method which is very simple to understand, code and parallelize, albeit slow for large point clouds. In this work, we analyze and improve the algorithm in order to make it applicable to large point clouds. We chose the NCH as the focus of this work because it is the first method to consider both spheres and planes as atoms for shape representation; all other algorithms that we are aware of work with either more complex objects fitted locally, such as polynomials and non-explicit functions, or simpler objects (i.e. planes). The NCH merges two approaches that have performed well before: planes [14], and spheres (as in the aforementioned MAT-based reconstructions). This contribution went largely unnoticed, and there have not been any improvements to the method since. The work of several authors in developing the Shrinking Ball (SB) algorithm [20] [15] [24], inspired us to make it work with the large datasets omnipresent today. Based on SB, we built a new fitting algorithm for the NCH called Shrinking Planes (SP). Our method reduces the complexity of estimating the NCH from $\Theta(N^2)$ to expected $\mathcal{O}(N \log N)$.

This work is organized as follows. We introduce the MAT and NCH, as well as several of their important theoretical properties in section 3, where we also explain the relationship between the MAT and the NCH. A new fitting algorithm for the NCH, called Shrinking Planes, is presented in section 4. Results of experimentation over different shapes and operating conditions is presented in section 5. Several conclusions and future work ideas are described in section 6.

2. Related Work

Vast amounts of work have been dedicated to surface reconstruction over the past two decades. For a more complete literature review, [5] is a great survey, and [10] an excellent book on the topic.

[14] wrote one of the first papers in the field. Techniques such as Poisson Surface Reconstruction by [17],

2475-7888/19/$31.00 ©2019 IEEE
DOI 10.1109/3DV.2019.00087

Multi-Level Partition Of Unity by [22], Point-Set Surfaces by [1], Algebraic Point-Set Surfaces by [13], and Smooth Signed Distance (SSD) by [30] arose out of considering smooth priors over the surface being fit. Recent reconstruction methods have become more specialized, in order to target specific shapes, structure, and priors in the missing data or noise. For instance, [7] performs surface reconstruction for point clouds with color information as well as normals.

The work of [3] on the Power Crust presented many interesting theoretical results that can be used to prove NCH's correctness. Substantial work has been dedicated to extracting medial representations from surfaces, see [27] for a thorough overview. There are methods for obtaining medial representations from both meshes and point clouds; only work that applies to the latter will be cited, as the former is considered out of scope for this work.

Signed Distance Functions are a common representation of 3D surfaces: the border $\partial\mathcal{S}$ of the surface $\mathcal{S}$ is represented as the σ level set of a scalar field $f : \mathbb{R}^3 \to \mathbb{R}$, which may be stated explicitly as a function definition or implicitly, for example as a uniform grid of samples. The σ level set is then extracted into a polygonal representation via isosurfacing algorithms. Methods based on this concept have been used for reconstruction from the very beginning [14], and it has been a common theme in many popular techniques such as Poisson Reconstruction [17], MPU [22], and Moving Least Squares-based techniques [1] [13].

The NCH can be seen as an extension of the MAT representation to allow planes: these are thought as spheres of infinite radius and define an implicit surface through the NCH Signed Distance Function. For isoextraction, the function that determines the grid is not needed, only samples of it and maybe its derivative. This is enough to compute a polygonal mesh using popular techniques, such as the classic Marching Cubes [19], or improved methods that work on the dual representation, like Dual Marching Cubes [21] and Dual Contouring [16]. Other approaches can generate a mesh directly from the MAT, such as in [9]; however, the methodology is significantly more complex and less explored.

There are many algorithms for computing the MAT from a point cloud; the most popular are derivations of [2]. Such methods rely on extracting a subset of the Voronoi diagram and properly scaling/filtering the MAs. The diagram is well defined for densely sampled surfaces, but for point clouds with missing data the corresponding Voronoi diagram may no longer resemble the MAT.

If approximate approaches are acceptable, more robust methods are available. Ma et al.[20] built an algorithm to approximate the MAT, called Shrinking Ball (SB), as a preprocessing step for skeletonization; Jalba et al.[15] worked on parallelizing it on the GPU and improving convergence. Recent work [24, 23] adapted the SB to include a denoising heuristic and approximate the θ Simplified Medial Axis (θ-SMA) [11].

3. The Medial Axis Transform and the Non-Convex Hull

In this section we formally introduce the representation of a surface as a union of balls, called the Medial Axis Transform (MAT), as well as the representation as a union of balls and planar half-spaces, called the Non-Convex Hull (NCH).

Given a surface $\mathcal{S}$ of a smooth, bounded, orientable (i.e. with a consistent choice of surface normal vector at every point) and watertight (closed) object, the pair $(\mathcal{P}, \mathcal{N})$ denotes an oriented point cloud for $\mathcal{S}$. Here $\mathcal{P} = \{\mathbf{p_1}, \ldots, \mathbf{p_N}\}$ s.t. $\mathcal{P} \subset \partial\mathcal{S}$ is a sampling of $\mathcal{S}$, and $\mathcal{N} = \{\mathbf{n_1}, \ldots, \mathbf{n_N}\}$ is the set of associated unit length normal vectors for $\mathbf{p_i}$, consistently oriented with respect to the surface.

3.1. Medial Axis Transform

Several equivalent definitions for the MAT exist. Our exposition is based on multiple sources; for a complete overview, the book by [27] can be a valuable resource. The following definitions are due to Choi et al.[8]:

Definition 3.1 (Set of inscribed disks)**.** Given Ω a connected and bounded subset of $\mathbb{R}^k$, $\mathcal{D}(\Omega)$ is the set of inscribed disks inside Ω:

$$\mathcal{D}(\Omega) = \{B_r(\mathbf{p}) \mid \mathbf{p} \in \Omega, r \in \mathbb{R}_{\geq 0}, B_r(\mathbf{p}) \subset \Omega\}, \Omega \subset \mathbb{R}^k$$

where $B_r(\mathbf{p})$ is the closed ball of radius r centered at p (if $r = 0$, then $B_r(\mathbf{p}) = \{p\}$).

Definition 3.2 (Core)**.** $\mathbf{CORE}(\Omega)$ filters $\mathcal{D}(\Omega)$ to maximal inscribed disks in Ω, each one called Medial Atom (MA) and defined as follows:

$$\begin{aligned}\mathbf{CORE}(\Omega) = \quad & \{B_r(\mathbf{p}) \in \mathcal{D}(\Omega) \mid \\ & B_s(\mathbf{q}) \in \mathcal{D}(\Omega) \wedge B_r(\mathbf{p}) \subset B_s(\mathbf{p}) \\ & \implies B_r(\mathbf{p}) = B_s(\mathbf{q})\}\end{aligned}$$

Definition 3.3 (Medial Axis)**.** $\mathbf{MA}(\Omega)$ is the set of centers of the Medial Atoms in $\mathbf{CORE}(\Omega)$:

$$\mathbf{MA}(\Omega) = \text{cl}(\{\mathbf{p} \in \Omega | B_r(\mathbf{p}) \in \mathbf{CORE}(\Omega)\})$$

Note that we take the closure cl of the ball's centers, i.e. including the limit points. This trivially makes the MAT closed, and even compact if Ω is bounded [26]; these facts will be useful later on when considering sharp edges.

Definition 3.4 (Medial Axis Transform)**.** $\mathbf{MAT}(\Omega)$ is the set of pairs consisting of the center and radius of each disks in $\mathbf{MA}(\Omega)$.

Remark 1. definition 3.2 implies that two Medial Atoms cannot have the same center. Thus, the mapping $\mathbf{MA}(\Omega) \rightarrow \mathbf{MAT}(\Omega)$ giving the radius for each center is a well-defined isomorphism. Concretely, they are just different representations of the same object.

In the literature, it is usual to split the MAT into Exterior (or Outer) and Interior (or Inner), where the Interior MAT refers to the definitions given earlier, and the Exterior is defined similarly for Ω^c. Intuitively, the Interior MAT builds a shape out of balls, while the Exterior MAT carves them out of a block of marble: in both cases we generate the same shape when the surface is infinitely sampled, but one method is "additive" and the other is "subtractive".

3.2. Non Convex Hull

The following exposition roughly follows [31] with slight differences leading to the same results.

Definition 3.5 (Half-Space)**.** Given $X \subset \mathbb{R}^k$ and $f : X \rightarrow \mathbb{R}$, we define $H_f = \{x \in X \mid f(x) \leq 0\}$ as a half-space. We call it linear when f is a linear function.

Definition 3.6 (Supporting Half-Space)**.** Given a point cloud $\mathcal{P} = \{\mathbf{p_1}, \dots, \mathbf{p_N}\}$, H_f is a supporting half-space for $\mathcal{P}$, when $\mathcal{P} \subseteq H_f$ and $\exists \mathbf{x} \in \mathcal{P} : f(\mathbf{x}) = 0$. (i.e. $\mathcal{P}$ is contained in it, with at least one point at its border).

Definition 3.7 (Oriented Convex Hull)**.** Let $(\mathcal{P}, \mathcal{N})$ be an oriented point cloud. Define $f_i(\mathbf{x}) = \langle \mathbf{n_i}, \mathbf{x} - \mathbf{p_i} \rangle$. Then:

$$\mathbf{OCH}(\mathcal{P}, \mathcal{N}) = \bigcap_i H_{f_i}$$

Remark 2. $f_i(x)$ as defined in the OCH is positive when x is inside of the half-space given by a plane centered at $\mathbf{p_i}$ with normal $\mathbf{n_i}$. Thus H_{f_i} is both the other side and the border connecting both sides.

Definition 3.8 (Non-Convex Hull)**.** Let $(\mathcal{P}, \mathcal{N})$ be an oriented point cloud, and $\rho = \{\rho_1, \dots, \rho_N\}$ such that $\rho_i \geq 0$. Define $f_i^{\rho_i}(\mathbf{x}) = \langle \mathbf{n_i}, \mathbf{x} - \mathbf{p_i} \rangle - \rho_i ||\mathbf{x} - \mathbf{p_i}||^2$. Then:

$$\mathbf{NCH}(\mathcal{P}, \mathcal{N}, \rho) = \bigcap_i H_{f_i^{\rho_i}}$$

Each f_i is called a basis function for $\mathbf{p_i}$ and note that all the half-spaces involved are supporting by construction. Furthermore, we define the set of Non-Convex Hull Atoms (NCHA) as:

$$\mathbf{NCHA}(\mathcal{P}, \mathcal{N}, \rho) = \{(\mathbf{p_i}, \mathbf{n_i}, \rho_i) \mid 1 \leq i \leq N\}$$

Remark 3. When $\rho_i = 0$, $f_i^0(\mathbf{x})$ is equivalent to the f_i used in the OCH (definition 3.7).

Lemma 3.1. *When $\rho_i > 0$, $f_i^{\rho_i}(\mathbf{x})$ is a SDF for the complement of a sphere centered on $\mathbf{p_i} + r_i \mathbf{n_i}$ with radius $r_i = \frac{1}{2\rho_i}$*

$$f_i^{\rho_i}(\mathbf{x}) = \frac{1}{2r_i}\{r_i^2 - ||\mathbf{x} - (\mathbf{p_i} + r_i \mathbf{n_i})||^2\}$$

Proof. Replace in the formula above the square norm:

$$\begin{aligned} f_i^{\rho}(\mathbf{x}) &= \frac{1}{2r_i}(r_i^2 - (||\mathbf{x} - \mathbf{p_i}||^2 + (-2r_i)\langle \mathbf{n_i}, \mathbf{x} - \mathbf{p_i} \rangle + r_i^2)) \\ &= \frac{1}{2r_i}(-||\mathbf{x} - \mathbf{p_i}||^2 + (2r_i)\langle \mathbf{n_i}, \mathbf{x} - \mathbf{p_i} \rangle) \\ &= \langle \mathbf{n_i}, \mathbf{x} - \mathbf{p_i} \rangle - \frac{1}{2r_i}||\mathbf{x} - \mathbf{p_i}||^2 \end{aligned}$$

Which is exactly the original definition. The important detail here is that the SDF for the sphere is undefined when $r_i = 0$, which is why the other representation is taken. ☐

Definition 3.9 (NCH SDF)**.** $f(\mathbf{x}) = \max_{1 \leq i \leq N} f_i^{r_i}(\mathbf{x})$ is also an SDF, representing the intersection of all $f_i^{r_i}$ (intersection of complements). Each one of these objects may be either a sphere, or a hyperplane.

We have thus far worked with an oriented point cloud $(\mathcal{P}, \mathcal{N})$, we call this NCH SDF $f^+(\mathbf{x})$. We denote by $f^-(\mathbf{x})$ the NCH SDF given by reversing the direction of the normals. The relationship between these two functions is akin to the Inner and Outer MAT. We call $\hat{f}(x) = \frac{f^+(x) - f^-(x)}{2}$ the Symmetric NCH SDF.

Nothing has been said up to this point about how $\rho = \{\rho_1, \dots, \rho_N\}$ should look. [31] defines it so we can always ensure that, for a given oriented point cloud $(\mathcal{P}, \mathcal{N})$ and point $\mathbf{p_i}$: (i) the point is at the border of the SDF, i.e. $f_i^{\rho_i}(\mathbf{p_i}) = 0$, (ii) the normal of the SDF at the point coincides, i.e. $\nabla f_i^{\rho_i}(\mathbf{p_i}) = \mathbf{n_i}$, (iii) other points are left outside of the sphere, or at its border, i.e. $f_i^{\rho_i}(\mathbf{p_j}) \leq 0$ for all $j \neq i$ (iv), r_i is maximal and $\rho_i \geq 0$.

3.3. Relationship with the MAT

The key difference between the MAT and the NCH is that the latter is also able to use planes as "Medial Atoms". If we take away this freedom from the NCH, what we are left is a shape represented as a union of balls (i.e. the MAT). The following lemma proves this:

Lemma 3.2. *Let $(\mathcal{P}, \mathcal{N})$ be an oriented point cloud, with ρ suitably defined such that $\rho_i > 0$ for all i (i.e. no plane fits are allowed). Also, let $\mathcal{S}^c = \mathbf{NCH}(\mathcal{P}, \mathcal{N}, \rho)$. Then, there is a surjective function $g : \mathbf{NCHA}(\mathcal{P}, \mathcal{N}, \rho) \rightarrow \mathbf{MAT}(\mathcal{S})$.*

Proof. We define $g(\mathbf{p_i}, \mathbf{n_i}, \rho_i) = (\mathbf{p_i} + r_i \mathbf{n_i}, r_i)$, s.t. $r_i = \frac{1}{2\rho_i}$. Take any $B_r(\mathbf{p}) \in \mathbf{MAT}(\mathcal{S})$; by construction, $\exists \mathbf{p_i} \in \mathcal{P} : \mathbf{p_i} \in \partial B_r(\mathbf{p})$; thus, observe that $g(\mathbf{p_i}, \mathbf{n_i}, \frac{1}{2r}) = (\mathbf{p}, r)$

as expected since the ball is tangent in $\mathbf{p_i}$ and the unit length normal $\mathbf{n_i}$ points to the center $\mathbf{p}$. Notice that such $\mathbf{p_i}$ must exist: we know that $\exists \mathbf{x} \in (\partial B_r(\mathbf{p}) \cap \partial \mathcal{S})$, this in turn implies that $\exists i \in [N]$ such that the associated ball to $\mathbf{p_i}$ has $\mathbf{x}$ in its boundary, as it is determined by the NCH SDF. $\square$

Concretely, the NCH covers the entire MAT, proving that it is a viable representation for general shapes. Indeed, lemma 3.2 also implies that many properties of the MAT also apply to the constrained NCH (when $\rho_i > 0$ for all i). This last assumption is reasonable in the Inner NCH, as any watertight shape will have at least one point in the direction of its normal when it is sufficiently sampled.

The reason to prefer the NCH over the MAT follows directly from the fact that planes are allowed. It is well known that the MAT has problems representing sharp edges. From a theoretical standpoint, this derives from the fact that the Medial Axis needs to include limit points in order to fit sharp edges [12]. In practice, there cannot be an arbitrarily accurate reconstruction of sharp edges unless more and more points are sampled near edge points. Practical applications that use the MAT usually do not handle this case or wade around this limitation by artificially generating such atoms [28]. One of the significant benefits of the NCH is that it does not suffer from such limitation: it is naturally able to represent sharp edges, simply by fitting planes instead of spheres. For example, a cube may be represented using one sample per side, with normals pointing outwards. Furthermore, the MAT produces poor reconstructions when few sample points are available, or when normals are not consistently oriented. As we will see, using the Symmetric NCH gives much better quality results in these cases.

Overall, a NCH representation can be more accurate, robust, compact, and require less samples than the MAT.

4. Shrinking Planes

In this section, we introduce Shrinking Planes, our algorithm to approximate the NCH; its pseudocode is shown in algorithm 1. In order to understand it, we must first explain Shrinking Ball, an algorithm to approximate the Medial Axis Transform proposed by [20], with follow-up work by [15] and [24]. The core idea is similar to the Naïve Non-Convex Hull (NNCH) algorithm [31], and has been spotted in unrelated works [28] [4].

Take an oriented input point cloud $(\mathcal{P}, \mathcal{N})$ sampled from $\partial\mathcal{S}$, and imagine you want to find a finite approximation to $\mathbf{MAT}(\mathcal{S})$. One idea is to fit one ball per point $\mathbf{p_j} \in \mathcal{P}$: first, begin with a ball B such that $\mathbf{p_j}$ is at its border and the normal matches accordingly; then pick any other point $\mathbf{p_i} \in (\mathcal{P} \cap \text{int}(B))^1$, and compute a new maximal ball with $\mathbf{p_i}$ in the border. Since the ball must be maximally contained, we can repeat the process until it cannot shrink any more (i.e, the intersection is empty), defining a sequence of progressively smaller balls $\mathcal{B}_j = \{B_{r_{ji}}(\mathbf{c_{ji}})\}_{i\in\mathbb{N}}$ that converges into a ball $B_{r_j}(\mathbf{c_j})$. If the sample is noise-free and sufficiently dense, doing this will result in a set of maximal balls $\mathcal{B} = \{B_{r_1}(\mathbf{c_1}), \dots, B_{r_N}(\mathbf{c_N})\} \subset \mathbf{MAT}(\mathcal{S})$. This is the core idea behind the SB algorithm; its pseudo-code, in a format close to the original paper but including our contributions, can be seen in algorithm 1. Without our changes, the algorithm is the Shrinking Ball variant later used for comparison.

[1] $\text{int}(B)$ is the interior of the set, using the standard definition

The NNCH algorithm runs a similar procedure, except that its initial "ball" is actually of infinite radius (i.e. a plane), and every point is considered instead of iteratively picking points. Seen this way, the SB algorithm is an clear optimization, since it can choose strategic points. However, for the purpose of reconstruction a few hurdles need to be overcome, as we explain below.

4.1. Radius initialization

Once we pick a point $\mathbf{p_j}$, the first step is to determine an initial radius (InitialRadius auxiliary method, line 2 algorithm 1). Ideally, this initial radius would be exactly the distance to the Medial Atom center, tangent to the point:

Definition 4.1 (Local Feature Size)**.** Given $\mathbf{w} \in \mathcal{S}$,

$$\mathbf{LFS}(\mathbf{w}) = d(\mathbf{w}, \mathbf{MA}(\mathcal{S})) = \inf_{\mathbf{c}\in\mathbf{MA}(\mathcal{S})} \{d(\mathbf{w}, \mathbf{c})\}$$

That is, the distance to the nearest point in $\mathbf{MA}(\mathcal{S})$.

Since this value is unknown, heuristics are used, and it is adjusted after every point is processed (AdjustRadiusHeuristic method, line 15 algorithm 1) to improve the estimation.

Thus, the first issue is that it unclear what the initial radius should be, as it can not be infinite (i.e. a plane): if it is too small, then the ball is not in the MAT, and will not cover the entire surface; if it is too large, it will protrude from the surface and convergence will be slow.

Ma et al. [20] proposed an heuristic that randomly chooses another point and uses it as border to compute the initial radius. For all following points, adjust it to the radius of a ball, in the proper direction of the normal for the point. From a theoretical standpoint, it exploits the fact that the next point should always keep the current point outside of its MA, which follows from the maximality requirement. However, this has a crucial problem, which is that such an atom does not necessarily exist: the previous point could be in the opposite direction of the current point's normal, which would require a negative radius. The authors did not elaborate on this problem in their work, and it is a significant one, especially when dealing with noisy point clouds.

Jalba et al.[15] presented a parallelization-friendly approach to initial radius estimation, which consists of dividing the point cloud into equal-sized chunks and processing one chunk per thread. Concretely, they estimate the initial radius to a number similar to the average LFS for the sample. A priori, the average LFS can be much smaller than the actual LFS of any given point, which translates into smaller balls than required (i.e. holes in the surface). They do not comment on this, and we could not find their source code for further experimentation or inspection.

The optimal initial value for a given $\mathbf{p_j}$ is $\mathbf{LFS}(\mathbf{p_j})$, meaning $\sup_{\mathbf{p}\in\mathcal{S}} \mathbf{LFS}(\mathbf{p})$ would be a reasonable choice for every point. Peters et al.[25, 24] use an user-supplied maximum radius for all points. It is a reasonable way to avoid the problem using this strategy, as we usually have an estimate of the maximum two points that can be apart, which is an upper bound to the LFS.

We estimate an upper bound to the maximum distance between any two points in linear time as the diagonal of the point cloud bounding box. This is guaranteed to be larger than the local feature size at any given sample, making it correct to use as a constant initial radius without incurring in a severe performance loss.

4.2. Shrinking the ball

Given a point $\mathbf{p_i}$ and its normal vector $\mathbf{n_i}$, we fit either a plane or a sphere. Spheres can be determined by either setting their radius or giving another point in its border.

If we fix another point $\mathbf{p_j}$, the radius of the sphere that puts both $\mathbf{p_j}$ and $\mathbf{p_i}$ in its border and guarantees that the normal at $\mathbf{p_i}$ matches is $r'_i = \frac{||\mathbf{p_j}-\mathbf{p_i}||^2}{2\langle \mathbf{n_i}, \mathbf{p_j}-\mathbf{p_i}\rangle}$ (TangentSphereRadius, line 5 of algorithm 1); r'_i will be negative when $\mathbf{p_j}$ is in the direction opposite to the normal with respect to the point, and infinite if both points lie in the same plane. Both NNCH and SB attempt to find a maximal ball that touches only two points of the surface (another characteristic of the MAT). For a given point, the former iterates through all remaining points trying to find the smallest maximal ball that fits, while the latter attempts to pick a subset of points and reduce the search space. If NNCH does not find a ball in the direction of the normal, it fits a plane.

The heuristic that SB uses to pick a subset of points is to exploit that these balls will be progressively smaller and closer to what the real value is, and hence a nearest neighbor search will yield candidates likely to be the real tangent point as iterations progress.

There is one significant issue with Ma's idea [20]: since the Kd-Trees used in practice are usually approximate, the "nearest neighbor" found may not be in the appropriate side of the normal of the point. Thus, the radius computation could return a negative value when it should found a better candidate point, and hence crippling the reconstruction process. A corrected version is already shown in algorithm 1 by only checking the sign of r'_i.

The SB algorithm as presented finds an approximation to the MAT. However, we are interested in approximating the NCH. This can be done by mimicking what the NNCH algorithm does: begin with a plane, and shrink from there on. Since it is not possible to use the SB algorithm with a plane as an initial guess (because there is no center to the Medial Atom being fit), we need to find a different way to choose to fit a plane.

If we can guarantee that the initial radius r_i is strictly greater than $\mathbf{LFS}(\mathbf{p_i})$, then we can also guarantee that any sufficiently dense sampling shrinks r_i at least once. However, points that ought to be planes instead are different: no point can be used to compute a new radius, causing a break from the iterative process in the very first iteration. Hence, we can check if the current radius is still the same as the initial one, and if so fit a plane (line 12 of algorithm 1).

We have taken $\delta = 1 \times 10^{-5}$ and $t_{\max} = 10$ (maximum number of iterations). For all models we have run, convergence take no more than 5 iterations. Smaller values of δ showed no significant improvements in the quality of the estimation.

Algorithm 1: Shrinking Planes algorithm for computing the NCH. Main differences with SB are highlighted.

```
 1: for $i = 1 \dots N$ do
 2:     $r_i \leftarrow$ InitialRadius($i$)
 3:     for $1 \dots t_{\max}$ do
 4:         $j \leftarrow$ NearestNeighbor($\mathbf{p_i} + r_i\mathbf{n_i}, \mathbf{p_i}$) {Excludes $\mathbf{p_i}$}
 5:         $r'_i \leftarrow$ TangentSphereRadius($\mathbf{p_i}, \mathbf{p_j}$) {$\frac{||\mathbf{p_j}-\mathbf{p_i}||^2}{2\langle \mathbf{n_i}, \mathbf{p_j}-\mathbf{p_i}\rangle}$}
 6:         if $r'_i < 0$ then
 7:             break {$\mathbf{p_j}$ is in the wrong direction of the normal}
 8:         swap($r_i, r'_i$)
 9:         if $|r_i - r'_i| < \delta$ then
10:             break {Algorithm has converged}
11:     RadiusRefinementStage($i$) {See algorithm 2}
12:     if $r_i \geq$ InitialRadius($i$) then
13:         $r_i \leftarrow \infty$ {Fit a plane with normal $\mathbf{n_i}$}
14:     $\rho_i \leftarrow \frac{1}{2r_i}$ {Convert into the NCH}
15:     AdjustRadiusHeuristic($i$)
16: return {$\rho_1, \dots, \rho_N$}
```

4.3. Radius refinement

The algorithm as presented up to here works for most shapes, but fails subtly in others. The causes were elusive to find, but easy to understand: it is likely to break the loop due to false convergence issues (line 10 of algorithm 1), since the radiuses often do not change critically from one iteration to the next; it also rarely happens that the iteration gets stuck between two points due to the usage of an approximate Kd-Tree. Our fix for these issues is to enforce the maximality of the ball; after the inner loop has finished (line 11 of algorithm 1), we are in one of three cases: (i) we have a plane -in which case it is impossible to shrink at all-, (ii) the ball has converged -i.e. it is maximal-, or

(iii) the ball has shrunk partially -thus the estimate can be improved-.

In any case, we can run a radius search using a KD-Tree (line 2 algorithm 2), which we already have for the nearest neighbor search in algorithm 1, and shrink it so that all points found are outside the ball. In the first case, only $\mathbf{p_i}$ will be returned; in the second case, only border points will be found, and thus no shrinking will happen; in the third case, points will be found inside the ball, which will shrink it further into convergence. See algorithm 2 for the pseudo-code.

Algorithm 2: RadiusRefinementStage

```
1: for $1 \dots y_{\max}$ do
2:   $N \leftarrow$ RadiusSearch$(\mathbf{p_i} + r_i\mathbf{n_i}, r_i, k)$
3:   changed $\leftarrow$ false
4:   for $j \in N$ do
5:     if $j \neq i$ then
6:       $r'_i \leftarrow$ ComputeTangentSphereRadius$(\mathbf{p_i}, \mathbf{p_j})$
7:       if $r'_i < r_i$ then
8:         $r_i \leftarrow r'_i$
9:         changed $\leftarrow$ true
10:  if not changed then
11:    break {Early stop refinement}
12: return $r_i$
```

It is important to point out that the radius search stage is not viable on its own as a way to compute the MAT, although it would be a correct algorithm if the number of neighbor points $k = \infty$, basically by being equivalent to NNCH. If the radius search is constrained to be returned in descending sorted order from the center point, the search becomes very costly and the algorithm runs more slowly. These problems imply that a good radius estimate is required before actually running it, and that is precisely what lines 3 through 10 of algorithm 1 do.

One possible problem with this algorithm, as provided, is the presence of noise, which may lead to over-shrinking. However, it should be straightforward to incorporate denoising heuristic based on [11]. Such modifications will not be explored in this work and will be left as future work.

In terms of correctness, on one hand, [20] proved that given an ϵ-sample, the SB algorithm converges to a subset of the MAT, and furthermore it converges to the true MAT as $\epsilon \to 0$. This indeed proves that error-free reconstruction is possible with SB. On the other hand, [31] established the same for the NCH algorithm.

In conclusion, algorithms 1 and 2 are basically equivalent, so the radius approximation derives directly from this property: a plane is fit only when there are no points in the direction of the half-space (which is exactly what NCH does), and otherwise we are guaranteed to find the maximal ball because of the call to algorithm 2 in line 11 of algorithm 1. Of course, this is a priori only true when $k = \infty$; otherwise, we may miss points.

4.4. Complexity

In terms of time complexity, it is important to point out that both SB and SP are $\mathcal{O}(N^2)$, as we have to process each point, and the nearest neighbor search may take $\mathcal{O}(N)$ in the worst case. Under the assumption that the data structure used for lookup is a KD-Tree, the expected complexity of the search is $\mathcal{O}(\log N)$, which implies that the expected worst case complexity is $\mathcal{O}(N \log N)$, a significant improvement from NNCH's $\Theta(N^2)$. Notice that $t_{\max}$, the number of iterations for shrinking (line 3 of algorithm 1), have not entered in our analysis, because this is a constant in practice and can be systematically reduced when using a good radius initialization heuristic. In the same way, $y_{\max}$ iterations of the radius refinement loop are allowed (line 1 of algorithm 2), but none of the shapes presented in this work took more than 2 iterations per point, and most of them only required one.

It is important to point out that the need of a radius search over the KD-Tree increases the time complexity of SP. The worst case of such a search is $\mathcal{O}(k \log N)$, where k is the number of neighbor points to be found, so that indeed turns the complexity into $\mathcal{O}(kN \log N)$. However, k was set to 10 because we did not notice any significant contribution to the convergence when setting it higher, leading to a total complexity of expected linearithmic time. As will be shown later in section 5, the difference in performance between SB and SP is negligible.

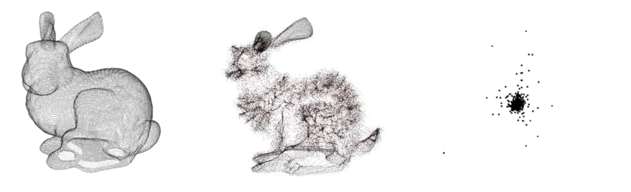
Finally, the algorithm has a spatial complexity of $\mathcal{O}(N)$, since that is how much the KD-Tree, points, normals and corresponding radiuses take to store.

5. Results

For all experiments run, the default parameters (named as they appear in the pseudo-code) will be used. All point clouds and meshes are inside the unit cube centered at 0 (so the choice of initial radius is good), and no re-scaling or transform is used. All normal vectors are always pointing inwards when input to the program.

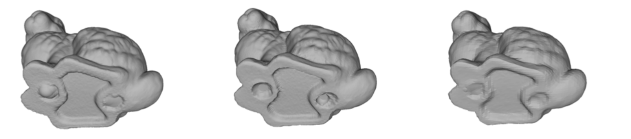
As a first example, notice in fig. 1 that the Outer MAs does not look like a fit to the shape. This is because there are more plane fits than spheres, and a point is drawn only if the ball's radius is not infinite. This renders the visualization for the Outer MAT useless.

In terms of the missing data, the results are more interesting than before, as shown in fig. 2. Notice the difference in NCH fit on the under side where there is missing data: the Inner (fig. 2 a) has protruding balls, while the Outer (fig. 2 b) goes in the other direction, as expected. Also, notice how the Symmetric NCH SDF makes a compromise between the two solutions (fig. 2 c).



(a) Oriented point cloud with 34k points. (b) Inner Medial Atoms. (c) Outer Medial Atoms.

Figure 1: MAT computed with the SP algorithm



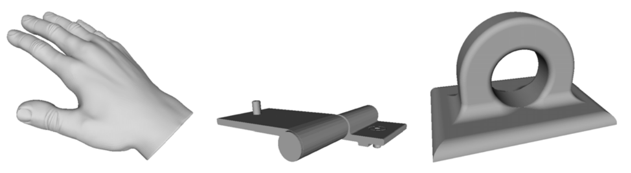
(a) Underside of the Inner NCH reconstruction (b) Underside of the Outer NCH reconstruction (c) Underside of the Symmetric NCH reconstruction

Figure 2: Detail of a reconstruction by the SP algorithm. The three meshes were extracted using the NCH SDF for the inner, outer and symmetric MAT.

5.1. Radius approximation

In order to evaluate the NCH fit, we compare the ρ^+ and ρ^- sets by computing the absolute value of the difference and looking at its distribution. We have the NNCH method which is what we want to approximate, and we have a two algorithms for approximating it, the Shrinking Planes (SP) and Shrinking Ball (SB) algorithms. Since the output of both methods must be of the same size, we can directly compare $\rho_i^{+\text{NNCH}}$ and $\rho_i^{-\text{NNCH}}$, the Inner and Outer NCH fit for ρ_i, respectively, against $\rho_i^{+\text{SP}}$ and $\rho_i^{-\text{SP}}$, the fit produced by SP, and analogously for the SB.

We take three models, shown in fig. 3: one with smooth features but large planar regions, one with sharp regions, and one with just smooth features. Then, we sample point clouds for each one, perform reconstruction using SB, SP and NNCH, and compare the results as explained above.



(a) Hand (b) Hinge (c) Key-pad

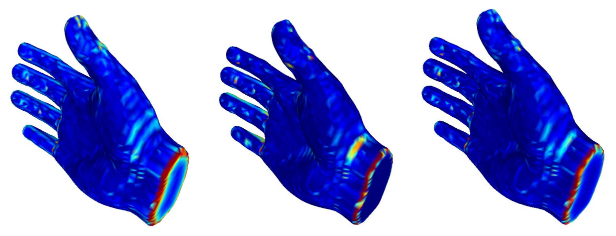
Figure 3: The reference models used for experimentation. In all cases, a 40k point cloud sample will be used throughout the experiments, since we found this size to visually preserve all features in the models.

The results of these comparisons are shown in table 1. As we can see, both SB and SP successfully approximate the radiuses computed by NNCH, with a better performance for SP in the Outer NCH case, as expected.

		Outer NCH		Inner NCH	
model / method		max	mean	max	mean
Key-pad	SP	5.15E-05	6.09E-07	9.44E-01	2.72E-04
	SB	4.35E-01	4.82E-05	9.44E-01	3.16E-04
Hinge	SP	1.53E-05	7.36E-07	1.52E-01	5.15E-05
	SB	1.84E-01	7.95E-05	1.52E-01	6.87E-05
Hand	SP	4.01E-05	2.79E-07	9.50E-01	2.02E-04
	SB	7.29E-01	8.91E-05	9.50E-01	2.28E-04

Table 1: Error Metrics for the NCH representations computed by Shrinking Ball (SB) and Shrinking Planes (SP), both compared against Naïve Non-Convex Hull (NNCH).

After this, we use the estimated radiuses to extract the zero-level isosurface of the NCH SDF using Marching Cubes with a resolution of 50x50x50, and compare results against the original source mesh using the Hausdorff distance. The three methods produce almost the same mesh, according to the Hausdorff distance for the Inner, Outer and Symmetric NCH reconstructions in all the models. From the table 1 we observe that SB and SP behave very similarly to NNCH for radius estimation, so it makes sense that the NCH SDF for each method also behaves in a similar way. However, there are small differences between SDFs as pointed out in the bunny example. For reference, a reconstruction of the Hand model using SP is shown in fig. 4.



(a) Inner NCH (b) Outer NCH (c) Symmetric NCH

Figure 4: Reconstructions of Hand point cloud sampled from the reference model. The process was performed using SP and MC with a resolution of 50 in all axes. Minimum value (blue) is 0, maximum value (red) is 0.003.

Finally, we run Screened Poisson (SPR) Reconstruction for each sampled model and compare the resulting mesh with the original one. For SPR, algorithm parameters were fixed to the defaults following [18] and implemented by the its authors[2]. Table 2 shows that SPR performs better for the Hinge and Hand models, however we can observe that the SDF's computed by Shrinking Planes present competitive results in all cases, being the best for the Key-pad model.

[2]https://github.com/mkazhdan/PoissonRecon

In general, SP generates approximations of the NCH that are very close to the NNCH. Thus, the behavior of both algorithms tends to be very similar; for example, neither are resilient to noise -especially in the normal vectors-, when compared to more complex algorithms such as SPR.

Model	Method / SDF	Error
Key-pad	SPR	2.34E-02
	INCH	**9.35E-03**
	ONCH	1.05E-02
	SNCH	9.95E-03
Hinge	SPR	**8.42E-03**
	INCH	4.53E-02
	ONCH	1.19E-02
	SNCH	3.24E-02
Hand	SPR	**5.72E-03**
	INCH	9.65E-03
	ONCH	1.01E-02
	SNCH	9.85E-03

Table 2: Errors measured by the Hausdorff distance for the reconstructed meshes. SP was used to approximate each SDF: Inner (INCH), Outer (ONCH) and Symmetric (SNCH). SPR stand for Screened Poisson Reconstruction.

5.2. Performance

Regarding timing measurements, we decided to take the Hinge model's reference mesh and generate multiple point samples, each increasing in size, and run the full reconstruction pipeline for NNCH, SP, and SB. All measurements have been done on a Intel Core i7-6500U running at 2.50GHz with 16 GB of memory.

The results from the measurements of the reconstruction stage (i.e. radius estimation) are shown in fig. 5. Notice that, as expected, for small point clouds we do not achieve a significant computation gain over NNCH (although there is still a 4 seconds difference on average). Also expected is the growth shown, in which our gains progressively become higher: notice how by 70k points we manage to perform the fit in roughly 24 seconds less on average. The SP algorithm is clearly slightly slower than SB. As expected, the memory requirements of SP are very low by today's standards, both for obtaining the NCH and for reconstruction keeping as high as 250 MB for all the tested models.

Performance was not compared with SPR because our isoextraction routine is very slow, mainly due to the NCH SDF evaluation and improving it is not the focus of this work.

6. Conclusions

We have developed the Shrinking Planes algorithm, which computes an approximation to the NCH in expected

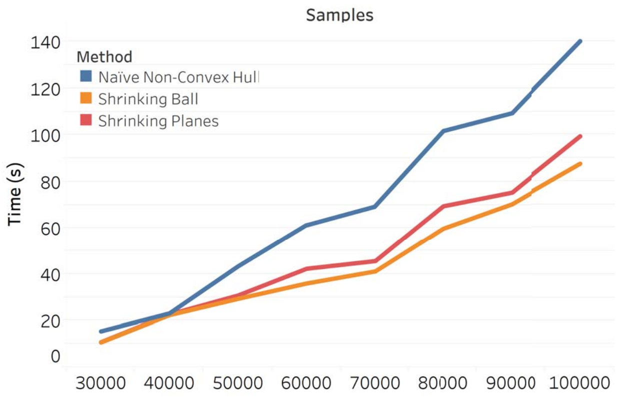


Figure 5: Performance measurements for radius estimation as the number of samples grow; three (averaged) runs are shown per sample size and algorithm

linearithmic time, a significant improvement over the original quadratic algorithm. We have tested its bounded-error reconstruction in densely sampled point clouds, and have run several experiments to test both its reconstruction quality and performance across different models. We have shown that the SP algorithm provides an excellent approximation to the NCH as computed by the NNCH algorithm, and in particular superior to that of the SB algorithm that we adapted for the NCH.

We also performed comparisons with Screened Poisson Surface Reconstruction, a state-of-the-art algorithm in the field, presenting very competitive results in terms of reconstructed meshes from point samples. It is quite clear that the algorithms shown in this work can appropriately reconstruct surfaces, albeit with difficulty in the presence of noise and extensive missing data. At last, our performance experiments showed our very small memory footprint, as well the much improved computation times of SP and SB in comparison to the Naïve Non-Convex Hull.

As to future work, all algorithms presented in this work are embarrassingly parallel, and thus very straightforward to parallelize both into multiple cores and into graphic processing units. [20] and [15] have done this previously as part of their work, and managed to get running times in the order of milliseconds for very large point clouds. The most time-consuming aspect of our implementation is the surface isoextraction. The NCH has a lot of structure, which could be exploited for faster isoextraction. We are currently working along these lines and hope to produce results in the near future.

Acknowledgments

This work was partially supported by the National Science Foundation grant N^o 1717355 and the Cognex Research Award Program.

References

[1] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva. Point set surfaces. In *Proceedings of the Conference on Visualization '01*, VIS '01, pages 21–28, Washington, DC, USA, 2001. IEEE Computer Society. 2

[2] N. Amenta and M. Bern. Surface reconstruction by voronoi filtering. In *Proceedings of the Fourteenth Annual Symposium on Computational Geometry*, SCG '98, pages 39–48, New York, NY, USA, 1998. ACM. 2

[3] N. Amenta, S. Choi, and R. K. Kolluri. The power crust, unions of balls, and the medial axis transform. *Comput. Geom. Theory Appl.*, 19(2-3):127–153, July 2001. 1, 2

[4] M. Berger and C. T. Silva. Medial kernels. *Comput. Graph. Forum*, 31(2pt4):795–804, May 2012. 1, 4

[5] M. Berger, A. Tagliasacchi, L. M. Seversky, P. Alliez, G. Guennebaud, J. A. Levine, A. Sharf, and C. T. Silva. A survey of surface reconstruction from point clouds. *Comput. Graph. Forum*, 36(1):301–329, Jan. 2017. 1

[6] H. Blum. A Transformation for Extracting New Descriptors of Shape. *Models for the Perception of Speech and Visual Form*, pages 362–380, 1967. 1

[7] F. Calakli and G. Taubin. *SSD-C: Smooth Signed Distance Colored Surface Reconstruction*, pages 323–338. Springer London, London, 2012. 2

[8] H. I. Choi, S. W. Choi, and H. P. Moon. Mathematical theory of medial axis transform. *Pacific J. Math*, 181:57–88, 1997. 2

[9] T. Delamé, C. Roudet, and D. Faudot. From a medial surface to a mesh. *Computer Graphics Forum*, 31(5):1637–1646, 2012. 2

[10] T. K. Dey. *Curve and Surface Reconstruction: Algorithms with Mathematical Analysis (Cambridge Monographs on Applied and Computational Mathematics)*. Cambridge University Press, New York, NY, USA, 2006. 1

[11] M. Foskey, M. C. Lin, and D. Manocha. Efficient computation of a simplified medial axis. In *Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications*, SM '03, pages 96–107, New York, NY, USA, 2003. ACM. 2, 6

[12] P. Giblin and B. B. Kimia. A formal classification of 3d medial axis points and their local geometry. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(2):238–251, Jan. 2004. 4

[13] G. Guennebaud and M. Gross. Algebraic point set surfaces. *ACM Trans. Graph.*, 26(3), July 2007. 2

[14] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. *SIGGRAPH Comput. Graph.*, 26(2):71–78, 07 1992. 1, 2

[15] A. C. Jalba, J. Kustra, and A. C. Telea. Surface and curve skeletonization of large 3d models on the gpu. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(6):1495–1508, 06 2013. 1, 2, 4, 8

[16] T. Ju, F. Losasso, S. Schaefer, and J. Warren. Dual contouring of hermite data. *ACM Trans. Graph.*, 21(3):339–346, July 2002. 2

[17] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP '06, pages 61–70, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association. 1, 2

[18] M. Kazhdan and H. Hoppe. Screened poisson surface reconstruction. *ACM Trans. Graph.*, 32(3):29:1–29:13, July 2013. 7

[19] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21(4):163–169, Aug. 1987. 2

[20] J. Ma, S. W. Bae, and S. Choi. 3d medial axis point approximation using nearest neighbors and the normal field. *The Visual Computer*, 28(1):7–19, 01 2012. 1, 2, 4, 5, 6, 8

[21] G. M. Nielson. Dual marching cubes. In *IEEE Visualization 2004*, pages 489–496, Oct 2004. 2

[22] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, and H.-P. Seidel. Multi-level partition of unity implicits. *ACM Trans. Graph.*, 22(3):463–470, July 2003. 2

[23] R. Peters. *Geographical point cloud modelling with the 3D medial axis transform*. PhD thesis, Delft University of Technology, 03 2018. ISBN: 978-94-6186-899-2. 2

[24] R. Peters and H. Ledoux. Robust approximation of the Medial Axis Transform of LiDAR point clouds as a tool for visualisation. *Computers & Geosciences*, 90(A):123–133, mar 2016. 1, 2, 4, 5

[25] R. Peters, H. Ledoux, and F. Biljecki. Visibility Analysis in a Point Cloud Based on the Medial Axis Transform. In *Eurographics Workshop on Urban Data Modelling and Visualisation 2015*, pages 7–12, Delft, Netherlands, Nov. 2015. 5

[26] E. C. Sherbrooke, N. M. Patrikalakis, and F.-E. Wolter. Differential and topological properties of medial axis transforms. *Graphical Models and Image Processing*, 58(6):574 – 592, 1996. 2

[27] K. Siddiqi and S. Pizer. *Medial Representations: Mathematics, Algorithms and Applications*. Springer Publishing Company, Incorporated, 1st edition, 2008. 2

[28] S. Stolpner, P. Kry, and K. Siddiqi. Medial spheres for shape approximation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(6):1234–1240, June 2012. 4

[29] R. Tam and W. Heidrich. Shape simplification based on the medial axis transform. In *IEEE Visualization, 2003. VIS 2003.*, pages 481–488, 10 2003. 1

[30] G. Taubin. Smooth signed distance surface reconstruction and applications. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 38–45, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. 2

[31] G. Taubin. Non-convex hull surfaces. In *SIGGRAPH Asia 2013 Technical Briefs*, SA '13, pages 2:1–2:4, New York, NY, USA, 2013. ACM. 1, 3, 4, 6