Progressive Discrete Domains for Implicit Surface Reconstruction

Tong Zhao, Pierre Alliez, Tamy Boubekeur, Laurent Busé, Jean-Marc Thiery

Eurographics Symposium on Geometry Processing 2021
K. Crane and J. Digne
(Guest Editors)

Volume 40 (2021), Number 5

Progressive Discrete Domains for Implicit Surface Reconstruction

Tong Zhao[1,2] , Pierre Alliez[1] , Tamy Boubekeur[3] , Laurent Busé[1] , Jean-Marc Thiery[2]

[1]Université Côte d'Azur, Inria, France
[2]LTCI, Télécom Paris, Institut Polytechnique de Paris, France
[3]Adobe, France

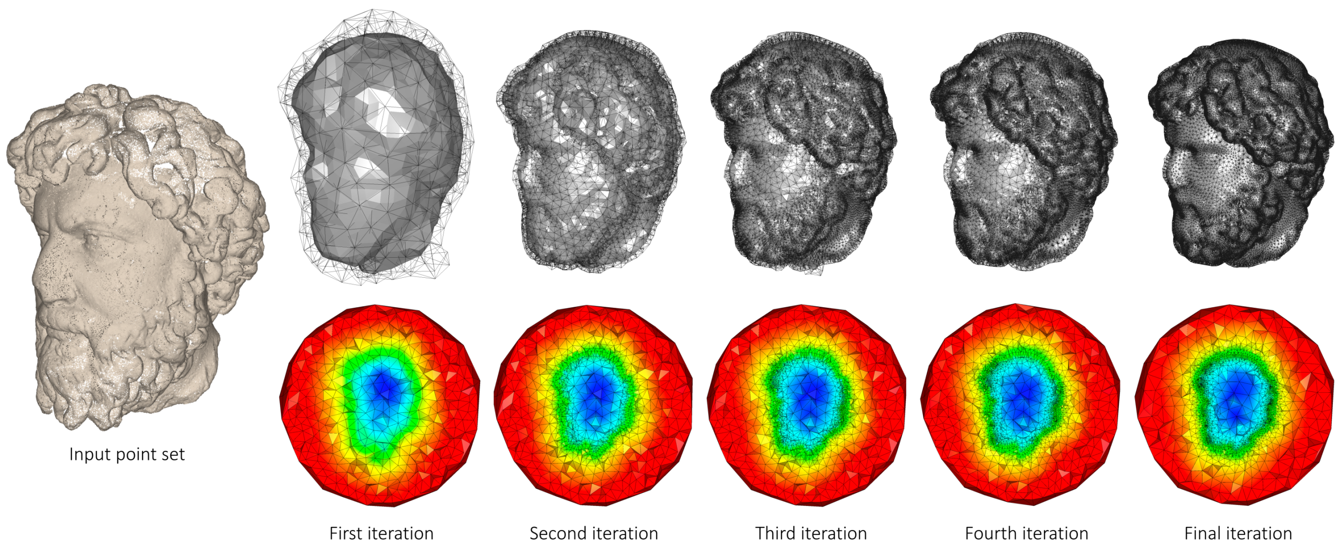


Figure 1: *Progressive discrete domain for surface reconstruction. Left: input point set. Right: our approach jointly refines and optimizes the implicit function (bottom) and its discretized domain (a 3D Delaunay triangulation) around the refined isosurface (top). In such a progressive approach, the implicit solver is used iteratively, as a means of consolidating hypotheses emitted in previous iterations. Top: the isosurface and only the set of tetrahedra intersected by the isosurface are shown. Bottom: the implicit function (piecewise-linear over the 3D triangulation) is depicted on the facets intersected by a clipping plane.*

Abstract

Many global implicit surface reconstruction algorithms formulate the problem as a volumetric energy minimization, trading data fitting for geometric regularization. As a result, the output surfaces may be located arbitrarily far away from the input samples. This is amplified when considering i) strong regularization terms, ii) sparsely distributed samples or iii) missing data. This breaks the strong assumption commonly used by popular octree-based and triangulation-based approaches that the output surface should be located near the input samples. As these approaches refine during a pre-process, their cells near the input samples, the implicit solver deals with a domain discretization not fully adapted to the final isosurface. We relax this assumption and propose a progressive coarse-to-fine approach that jointly refines the implicit function and its representation domain, through iterating solver, optimization and refinement steps applied to a 3D Delaunay triangulation. There are several advantages to this approach: the discretized domain is adapted near the isosurface and optimized to improve both the solver conditioning and the quality of the output surface mesh contoured via marching tetrahedra.

CCS Concepts

• ***Computing methodologies*** → ***Mesh geometry models;*** *Point-based models;*

1. Introduction

Assuming input measurement data provided as an unorganized point set, surface reconstruction is the process of recovering shapes or entire scenes that fit these data, while dealing with defect-laden or missing data. The reconstruction problem is inherently ill-posed as an infinite number of shapes may fit the data. The common wisdom consists of reducing the search space, i.e. regularizing the problem via adding predetermined priors deriving from assumptions about geometry, semantics, acquisition, or structure. A geometric prior may relate to trivial topology, absence of boundaries, canonical shape primitives or smoothness. Furthermore, data fitting is only half the problem, as satisfactory complexity-distortion tradeoffs are also sought after. In addition, other properties are desirable for downstream use of the reconstructed discretized surfaces (often triangle meshes), such as well-shaped elements and adaptive density.

Global implicit surface reconstruction methods commonly hinge upon global solvers (possibly multi-scale). These solvers, commonly tailored to discrete differential operators, yield implicit functions that trade data fidelity for geometric prior matching, where the prior often favors closed smooth surfaces. Such global solvers require discretizing both the 3D domains where the implicit function is defined and the aforementioned operators, and contouring the implicit function to extract the final meshes. Ideally, the above discretization should provide (1) just-enough degrees of freedom near the reconstructed surface for the solver, (2) well-shaped volumetric elements everywhere to ensure good numerical conditioning for the solver, (3) geometric regularity when inferring smooth surfaces, and (4) well-shaped elements of the output surface mesh.

1.1. Previous Work

Surface reconstruction has been explored extensively during the past decades and a wide range of approaches tackle different settings [Dey06, BTS*17, MWA*13]. We restrict our review to implicit global reconstruction methods with a focus on discretization issues and a mention to coarse-to-fine reconstruction approaches. As our progressive approach utilizes tetrahedron mesh optimization and refinement principles, we also review the related work.

Implicit Surface Reconstruction. A general implicit surface is defined as the level-set of a scalar function $\{f(x), x \in \mathbb{R}^3\}$. The existing approaches mostly differ in the way they (1) define the scalar function with respect to the inferred surface (e.g., smoothed indicator [KBH06], signed distance [CT11], signed robust Wasserstein distance [MdGD*10], or occupancy function [MON*19]), (2) define the objective function (often trading data fidelity for regularity) and (3) solve for it (Poisson equation, generalized eigenvalue solver or machine learning). Other main differences include the choice of level-set and the method used for contouring it (marching cubes, marching tetrahedra, dual contouring, Delaunay refinement).

In the aforementioned methods, an important aspect is the type of domain discretization used for representing the implicit function. The popular Poisson reconstruction method and its variants [KBH06, KH13, KCRH20] refine an octree before solving. It utilizes smooth basis functions defined on the octree elements, and requires diffusing the input normals in order to compute the divergence operator. As smooth functions and normal diffusion tend to oversmooth, the screened variant [KH13] adds a data fitting term to the objective function, to favor that the isosurface passes through the data points. On areas with missing data, such approaches can generate spurious surfaces and coarse output meshes where the cells of the octree are not refined. A recent variant [KCRH20] adds a close envelope and constraints the solver to generate level sets only inside this envelope, which significantly reduces the undesired spurious surfaces. The smooth signed distance (SSD) reconstruction approach proposed by Calakli and Taubin [CT11] also utilizes an octree before solving, but does not require diffusing the normals or discretizing the divergence operator. A spectral approach [ACTD07] removes the need for oriented point sets as input points, at the cost of solving for a generalized eigenvalue problem instead of a simpler linear system. It uses a 3D Delaunay triangulation as domain discretization, refined before solving as in previous work. The signing-the-unsigned approach [MdGD*10] signs an outlier-robust distance function. It performs an adaptive domain discretization by using an octree only for initialization, combined with Delaunay refinement before solving.

In summary, most common implicit-based approaches rely upon a predetermined domain discretization: they determine a data structure (often an axis-aligned bounding volume hierarchy such as an octree) from the local density of input samples, a resolution for the reconstructed surfaces, and then utilize a global solver to infer an implicit surface. Refining the data structure *a priori*, and only near the input samples, can either overlook areas where the solver completes missing data or over-refine where the reconstructed surface is flat. In addition, the axis-aligned nature of the above data structure is too rigid: it produces output meshes that may contain badly shaped elements and the overall approach is not intrinsically invariant to rotations. We are thus left with a chicken-and-egg problem, as knowing the reconstructed surface requires solving, and ideal solving conditions require knowing the final reconstructed surface. The difficulty of sequencing actions where each seems to depend on others calls for a progressive approach in which the solver is used iteratively, as a means of consolidating hypotheses emitted in previous iterations, and interleaved with (isosurface-driven) optimization and refinement of the discretization. Several approaches already proceed coarse-to-fine during reconstruction (e.g., [OBA*03, SLS*06] to cite a few), but do not jointly refine and optimize both the implicit function and its representation domain, where optimize should be understood by adapting the domain geometry around the isosurface.

Tetrahedron Mesh Optimization. Quality meshes commonly refer to tetrahedron elements with controlled size and shape, which offer both accuracy and conditioning for discrete operators such as Laplacian, Hessian or divergence [She02]. Some approaches optimize via local topological transformations [LCS09, CZZ*17]. Other approaches relocate the vertices, either locally [JWZ11] or globally [VWP13]. Klingner and Shewchuk use an even broader repertoire of mesh transformations [Kli08]. These approaches are insufficient for our specific context where both the discretized domain (shape of tetrahedra) and the discretized isosurface (shape of triangles extracted via marching tetrahedra) must be optimized jointly.

Tetrahedron Mesh Refinement. Mesh refinement is a relevant paradigm for improving the complexity-distortion tradeoff of finite element simulations [BKK20]. Mesh adaptation requires estimating the simulation error [GBA*17]. Delaunay refinement is a popular greedy refinement method for isotropic mesh generation and shape approximation [CDS13, Si15]. The Delaunay refinement paradigm is also relevant for discretizing the 3D domain in our context, by inserting circumsphere centers of tetrahedra. However, we also need a specialized refinement scheme around the inferred surface (discretized by marching tetrahedra), tailored to preserve the shape of isosurface triangles.

1.2. Positioning and Contributions

Departing from most implicit surface reconstruction methods that discretize the domain *a priori* based on the input data, we propose an output-sensitive progressive coarse-to-fine approach that jointly refines the implicit function and its representation domain, while discovering its isosurface (see Figure 1). The motivation for such an approach stems from the observation that the global surface reconstruction problem is two-fold: the domain should be adapted both for the solver to capture adequately the variations of the solution as a scalar field ($\{f(x), x \in \mathbb{R}^3\}$) and then specifically around an isosurface to be extracted (without loss of generality, $S = f^{-1}(0)$). Dedicated refinement and optimization approaches must be developed for these two entangled sub-problems. Our method is intended to deal with implicit reconstruction methods that can be discretized on tetrahedron meshes.

Our key insights and technical contributions are:

- *The domain discretization should be of high quality*, so that the differential operators used to solve for the implicit function can perform reliably. Meanwhile, the discretization should be economical, to allow for fast updates of the solution between two iterations. We contribute a sparse refinement scheme that allocates degrees of freedom where most needed.
- *The local discretization density should be adapted to the target surface*: denser near the isosurface, and even denser where it exhibits small local feature size. Without prior knowledge about the output surface (i.e., the locus where $\{f(x) = 0\}$), rendering pre-allocation of an adaptive structure around input samples inadequate, we adopt a progressive approach that allocates additional degrees of freedom that are necessary to improve the accuracy and quality of the isosurface.
- *The isosurface should be of high-quality*. Given a current isosurface contoured by marching tetrahedra, we optimize the tetrahedron mesh so that the isosurface intersects the tetrahedra preferably through their edge midpoints. Our approach differs from the recent contribution from Hass and Trnkova [HT20] in that we also optimize for the shape of tetrahedra.
- Our method is intended to be generic, making little assumptions about the implicit function, solver and surface extraction method, so that existing solvers can be used as black boxes. We demonstrate the relevance of our method by instantiating it on the Poisson, smooth signed distance (SSD) and spectral reconstruction methods.

2. Background

2.1. Implicit Surfaces

An implicit surface, referred to as isosurface in the sequel, is defined as the level-set of a function $f : \mathbb{R}^3 \to \mathbb{R}$. Some conditions such as non-vanishing gradients are required to ensure that the isosurface is a surface. In our context where we perform a progressive implicit reconstruction of surfaces from point samples, we distinguish between three main cases, locally: (1) The isosurface passes near the input points: the original objective of faithfully approximating the input data is met; (2) The isosurface locally passes far away from the input points: either it fills a hole or the domain discretization is too coarse, or a high regularization term creates a high tension of the isosurface; (3) Some points are isolated, i.e., not locally approximated by the isosurface: these points are outliers or the domain discretization is too coarse.

Once the implicit function is computed, the isosurface can be contoured by the *marching tetrahedra* approach. Since the function is linearly interpolated inside each tetrahedron, in general (omitting degenerate cases), it extracts linear surface elements (quadrangles or triangles) inside the tetrahedra whose vertices have function values with opposite signs. The isosurface is thus a hybrid quad-triangle surface mesh.

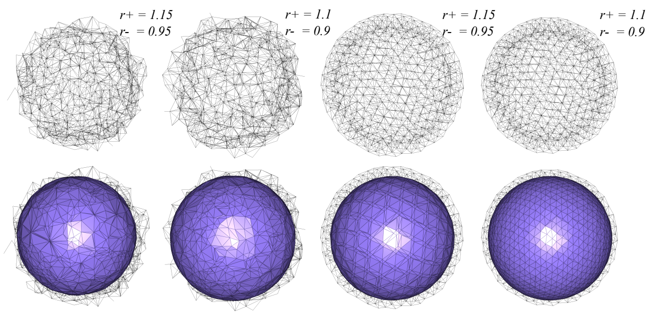
2.2. Discretization

What is a good tetrahedron discretization for the global implicit reconstruction problem is a central question in our context. The tetrahedron elements should be well-shaped to ensure a good conditioning of the solver. In addition, the 3D triangulation should exhibit denser elements only near the isosurface, where the inferred surface has a small local feature size (equivalently: large curvature, small thickness or separation distance). Furthermore, the quality of the isosurface mesh elements has a close relationship with the discretized domain used to represent the implicit function, for the tetrahedra intersecting the isosurface. Figure 2 depicts four different discretizations of an implicit function approximating a unit sphere, and the resulting isosurface meshes. The implicit function is defined analytically in order to eliminate the solver's influence, and all quadrangles are triangulated so as to maximize the smallest triangle angle. This illustrates that contouring well shaped tetrahedron elements through their edge midpoints lend to isosurface meshes containing mostly high-quality triangles, with controlled size. To summarize, we seek a discretized domain whose elements are well shaped, denser near detailed isosurfaces and sparser far away, with one layer of well shaped tetrahedra intersecting the isosurface in their edge midpoints.

3. Method

3.1. Overview

Our algorithm takes as input a 3D point set with oriented or unoriented normals (depending on the implicit solver) and generates as



(a) Biased, Noisy (b) Unbiased, Noisy (c) Biased (d) Unbiased

Figure 2: *Contouring a discretized domain. (a) A "biased" triangulation is generated by two layers of randomly-generated vertices located on two concentric spheres with inner radius 0.95 and outer radius 1.15. (b) A noisy yet unbiased triangulation is generated by two layers of random vertices sampled on two concentric spheres (outer radius 1.1, inner radius 0.9). The resulting isosurface contains badly shaped triangles as well. (c) A biased triangulation is generated by two layers of evenly-placed vertices sampled on two concentric spheres. Although the tetrahedra are well shaped (isotropic, i.e., nearly-equilateral), the isosurface mesh contains many skinny triangles which correspond to triangulated quadrangle elements connecting well shaped triangles of different size due to the biased triangulation. (d) A unbiased triangulation is generated by two layers of evenly-placed vertices sampled on two concentric spheres. Most triangles of the isosurface are well shaped and uniformly sized.*

output a discretized reconstruction domain following the pipeline shown in Figure 3. We assume that the point set is sampled around a closed 2-manifold smooth surface, i.e., the boundary of a solid. Some measurement noise as well as missing data are tolerated.

We aim at generating and adapting a tetrahedron mesh discretizing the domain, serving as a support of an implicit function computed via a user-defined implicit surface reconstruction solver, from which we extract an isosurface in the form of a surface triangle mesh. Possible choices for the solver include Poisson (screened or not) [KBH06, KH13], smooth signed distance [CT11] and spectral [ACTD07].

Our algorithm outputs an isosurface that fits well the input data set while exhibiting the following properties: (1) It represents an approximation of the smooth inferred surface, with a uniform or adaptive sizing; (2) It completes missing data (holes) with piecewise linear approximations of smooth surface patches, in accordance to the regularity requested for the solver; and (3) Its triangles are well shaped, i.e. isotropic.

A main user parameter of our algorithm is the aforementioned sizing field, which provides indirect control over the final mesh complexity. In the uniform case, a target triangle area is required. In the adaptive case, a variable sizing field is required. We propose such a sizing field based on a local curvature estimate.

Notations. The input point set is denoted by $\mathcal{X} = \{(p_1, n_{p_1}), \ldots, (p_N, n_{p_N})\}$. The loose bounding 3D volume mesh is denoted by $\mathcal{T} = \{V, E, T\}$ where $V = \{v_1, \ldots, v_M\}$ denotes the vertex set, $E = \{e_1, \ldots, e_P\}$ denotes the edge set and $T = \{t_1, \ldots, t_Q\}$ denotes the tetrahedron set. The implicit function is denoted by a piecewise linear scalar function $f : \mathbb{R}^3 \to \mathbb{R}$ defined onto $\mathcal{T}$ and the isosurface is denoted by a set of oriented triangles $\mathcal{S} = f^{-1}(0) = \{(a_1, n_{a_1}), \ldots, (a_R, n_{a_R})\}$. A uniform (constant) sizing field is denoted by s, and an adaptive sizing field is defined as a scalar function $s(x)$ from $\mathbb{R}^3 \to \mathbb{R}$.

Assumptions Made on the Solver. Our method requires the solver to be discretizable on tetrahedron meshes. This is the case for all solvers requiring discrete differential operators such as the Laplacian, gradient, divergence and Hessian. For solvers that take as input parameter a kernel for diffusing attributes from the input samples (e.g., normals for Poisson reconstruction), we require as parameter the kernel size in order to ensure a minimal vertex sampling around the input point samples. SSD and Spectral reconstruction do not suffer from this constraint. We assume that the solver provides as output a piecewise-linear scalar implicit function defined on the tetrahedron mesh vertices, and that the output surface is the zero level set of this function.

3.2. Initialization

Given the input point set $\mathcal{X}$, we compute its bounding sphere whose center is denoted as $c_{\mathcal{X}}$ and radius denoted as $r_{\mathcal{X}}$. The geometric domain is initialized by inserting into a 3D Delaunay triangulation 100 points uniformly sampled on the surface of a sphere centered at $c_{\mathcal{X}}$ with radius $1.4r_{\mathcal{X}}$. We then perform 3D Delaunay refinement according to the circumradius-to-shortest edge ratio (threshold set to 1.3 by default), which ensures an initial quality 3D triangulation. The initialization has little impact on the final reconstructed surface when the enlarge ratio (1.4 by default) is large enough. Its role is to bootstrap the refinement algorithm after obtaining an initial implicit function from the solver. The domain is then further discretized gradually in the following process.

3.3. Optimization

Given the current triangulation $\mathcal{T}$, implicit function and relating isosurface $\mathcal{S}$, the optimization step is designed to relocate the triangulation vertices so as to find a balance between improving the shape of tetrahedra (making them as isotropic as possible), favoring that tetrahedra intersected by $\mathcal{S}$ are intersected through their edge midpoints (creating a layer of tetrahedra "sandwiching" the isosurface), while preserving the boundary of the initial bounding sphere.

We formulate the optimization as a function-minimization process for the variable vertex positions $\{v_i\}$, in which the objective function comprises three terms: (1) The As-Similar-As-Possible (ASAP) term E_a rewards isotropic tetrahedra, weighted by a coefficient λ_a, (2) The mid-edge term E_m rewards midpoint isosurface intersection, weighted by a coefficient λ_m and (3) The damping term E_d penalizes large-scale vertex relocation far away from the isosurface.

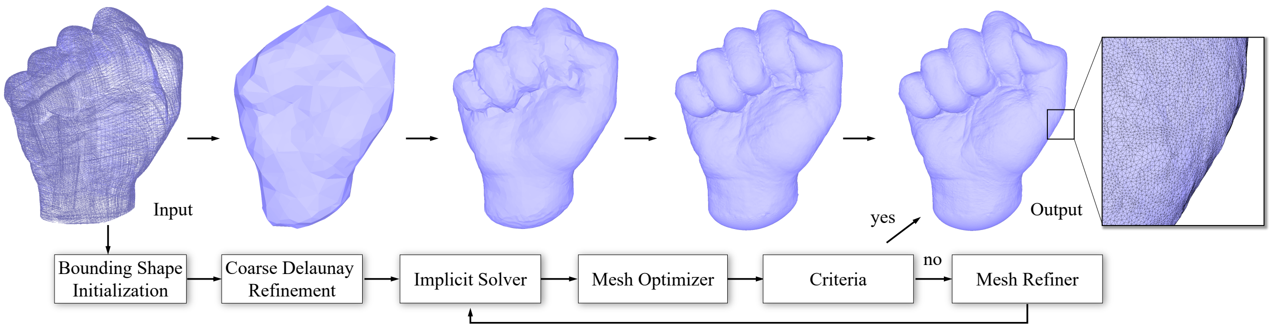


Figure 3: *Overview. The domain boundary is initialized by a loose sphere bounding the input point samples, refined by coarse Delaunay refinement. The algorithm then iterates through the solver, optimizer and refiner. The isosurface, contoured by marching tetrahedra, is generated as output once user-specified criteria are met.*

Denote by $T_{\mathcal{S}}$ the set of tetrahedra intersecting $\mathcal{S}$, the objective function $E(\mathcal{T})$ is defined as:

$$E(\mathcal{T}) = \lambda_a \sum_{t\in\mathcal{T}} w_a(t)E_a(t) + \lambda_m \sum_{t\in\mathcal{T}_{\mathcal{S}}} w_m(t)E_m(t) + \sum_{v\in V} w_d E_d(v). \tag{1}$$

Figure 4 depicts the two principal terms of the optimized objective function.

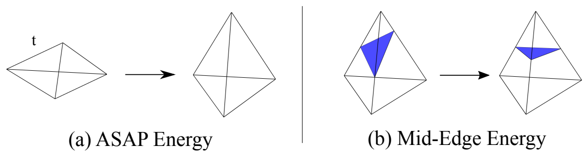


Figure 4: *Objective function. (a) The as-similar-as-possible (ASAP) term favors the tetrahedron t to become equilateral under volume constraint. (b) The mid-edge term favors the midpoint of edges with opposite function value signs to pass through the isosurface (blue).*

ASAP Objective. The goal is to deform each tetrahedron t to make it as congruent as possible to an equilateral tetrahedron, referred to as reference tetrahedron, under a scaling operator that preserves a given volume, while preserving its center. We define the reference tetrahedron t' as the centered unit regular tetrahedron and find the optimal transformation by minimizing the following function:

$$E_a(t) = \min_{\mathbf{S}} \sum_{i=0}^{3} \|(c(t) + \mathbf{S}v_{t'_i}) - v_{t_i}\|^2 \tag{2}$$

where $\mathbf{S}$ denotes an arbitrary isotropically stretched rotation matrix (a similarity matrix), $c(t)$ denotes the center of t, and t_i (resp. t'_i) denotes the i^{th} corner of t (resp. t'). To find the minimizer matrix $\mathbf{S}^*$, we borrow our idea from the local/global optimization strategy introduced in [SA07]. We start from translating the centroid of the tetrahedron $c(t)$ to the origin, and then compute the 3×3 covariance matrix as

$$\mathbf{C}(t', t - c(t)) = t'(t - c(t))^T. \tag{3}$$

We then perform a singular value decomposition (SVD) onto the covariance matrix:

$$\mathbf{C} = \mathbf{U}\Sigma\mathbf{V}^T. \tag{4}$$

To determine the scaling factor of $\mathbf{C}$, we first compute the volume of all tetrahedra T and then smooth it by averaging the volume values of adjacent tetrahedra, which decreases the sizing gradation among adjacent tetrahedra, thus better conditions the system. Given $\overline{vol}(t)$, the smoothed volume of the tetrahedron t, we set the modified singular value matrix Σ' to be:

$$\Sigma' = \sqrt[3]{\frac{\overline{vol}(t)}{vol(t')}} \cdot \mathbf{I}_3. \tag{5}$$

The minimizer matrix $\mathbf{S}^\star$ is set to:

$$\mathbf{S}^\star = \mathbf{V}\Sigma'\mathbf{U}^T. \tag{6}$$

If the determinant of $\mathbf{S}^\star$ is negative, we invert the sign of the last singular vector of $\mathbf{U}$ and then recompute $\mathbf{S}^\star$. The objective function can then be written as:

$$E_a(t) = \sum_{i=0}^{3} \|(c(t) + \mathbf{S}^* v_{t'_i}) - v_{t_i}\|^2 \tag{7}$$

where the vertex locations v_{t_i} denote the variables under optimization.

Mid-edge Objective. This term favors that the isosurface intersects tetrahedra through their edge midpoints. Combined with the ASAP objective, contouring the intersected tetrahedra yields well-shaped triangles with locally uniform sizing, where the length should be expressed within the norm of the sizing field.

Given a tetrahedron t intersected by isosurface $\mathcal{S}$, we consider only the edges intersecting $\mathcal{S}$, i.e. the edges with alternating signs of the implicit function value. In essence, this term encourages the midpoint of all edges crossing $\mathcal{S}$ to be mapped to it, which in turn encourages the isosurface $\mathcal{S}$ to be located in the middle of two offset surfaces composed of the faces of $T_{\mathcal{S}}$ that do not cross $\mathcal{S}$.

$$E_m(t) = \sum_{e \in t, f(v_{e_1}) \cdot f(v_{e_2}) < 0} \left(\left(\frac{v_{e_1} + v_{e_2}}{2} - p_e \right) \cdot n_{a_t} \right)^2, \qquad (8)$$

where a_t denotes the isosurface inside tetrahedron t (a quadrangle, resp. a triangle, if the number η_t of edges of t crossing $\mathcal{S}$ is 4, resp. 3), p_e is the intersection of edge e and a_t, and n_{a_t} is the normal of a_t. The variables v_{e_1} and v_{e_2} refer to the vertex locations of the vertices of edge e.

Damping Objective. To prevent the boundary of the triangulation $\mathcal{T}$ from expanding or shrinking during optimization, we add a damping term to restrict the movement of the vertices near $\partial\mathcal{T}$. More specifically, we first iterate over each vertex v and compute the minimum / maximum function values $f_{\min}$, $f_{\max}$, and the minimum / maximum function values at each vertex adjacent to a tetrahedron crossing the isosurface $f^{\mathcal{S}}_{\min}$, $f^{\mathcal{S}}_{\max}$.

The weight of vertex $g_d(v)$ is defined as:

$$g_d(v) = \begin{cases} \dfrac{f(v) - f^{\mathcal{S}}_{\min}}{f_{\min} - f^{\mathcal{S}}_{\min}}, & \text{if } f(v) < f^{\mathcal{S}}_{\min} \\ \dfrac{f(v) - f^{\mathcal{S}}_{\max}}{f_{\max} - f^{\mathcal{S}}_{\max}}, & \text{if } f(v) > f^{\mathcal{S}}_{\max} \\ 0, & \text{otherwise.} \end{cases} \qquad (9)$$

Large holes may generate open isosurfaces with boundary on the domain boundary $\partial\mathcal{T}$. In order to avoid shrinking of the isosurface's boundary, we compute the distance from each vertex v to $\partial\mathcal{T}$ and set $g_d(v) = 1$ for vertices located near $\partial\mathcal{T}$ to ensure that they remain unchanged. Once the weight $g_d(v)$ is determined, the objective is computed as:

$$E_d(v) = g_d(v) \cdot \|\tilde{v} - v\|^2, \qquad (10)$$

where $\tilde{v}$ denotes the vertex position at the previous iteration.

There are several reasons motivating this design choice: **i)** We observed that large sliver tetrahedra are generally located at the boundary of $\mathcal{T}$, and while they contribute in practice very little to the accuracy of the solution around the target isosurface $\mathcal{S}$, penalizing their shape's distortion comes at the much higher price of preventing optimization of the tetrahedra around $\mathcal{S}$. **ii)** Formulating the damping weights as a function of the scalar field f itself allows us to favor shape improvement of the tetrahedra around $\mathcal{S}$ without having to walk explicitly on the triangulation (making it possible to set them in linear time), or updating and querying a nearest neighbor structure, which would be necessary if the weights were defined in terms of distance to $\mathcal{S}$. **iii)** For all solvers we consider, the *uncertainty* of $\mathcal{S}$ is better expressed in terms of variation of the scalar field around 0 than in terms of distance to $\mathcal{S}$ (consider the Poisson solver for example, for which large holes in the data are filled with an extremely slowly-varying scalar field f).

Weighting Functions. To make the optimization invariant to rigid motion and scaling of the input point set $\mathcal{X}$ and triangulation $\mathcal{T}$, we define weighting functions for the three terms respectively. Denote by $r(\mathcal{T})$ the diameter of the triangulation (furthest distance between pairs of vertices). For the ASAP term we define $w_a(t) = vol(t)/r(\mathcal{T})^3$; for the mid-edge term we define $w_m(t) = area(a_t)/r(\mathcal{T})^2$; for the damping term we define a high coefficient, by default $w_d = 100/r(\mathcal{T})$, which makes it a hard constraint to $E(\mathcal{T})$.

Linear Solver. Since the three above terms are quadratic, the total objective function $E(\mathcal{T})$ is quadratic and can be minimized by solving a linear system with $3|V|$ variables. We initialize the solution by the unoptimized vertex coordinates and utilize an iterative linear solver. As all vertices are relocated, we update the implicit function values at the vertices by linear interpolation over the triangulation computed by the previous solver, since the cost of running a solver on a large triangulation is substantially more expensive than the one of the optimizer.

3.4. Adaptive Refinement

We now describe a progressive and parsimonious refinement process for the 3D triangulation of the domain, in order to provide "just enough" degrees of freedom around the isosurface $\mathcal{S}$ for further computation. More specifically, we propose two refinement schemes tailored to induce a local subdivision of the triangle facets of the isosurface with variable granularity, that preserves the shapes of both the facets of the isosurface and tetrahedra of the 3D triangulation.

First observe the two possible configurations of a tetrahedron intersecting the isosurface, i.e. with both positive and negative function values at its vertices (see Figure 5). For case 1, we have either 3 positive values and 1 negative value, or vice-versa: the isofacet a_t is a triangle. For case 2, we have 2 positive and 2 negative values: the isofacet a_t is a quadrangle. In order to triangulate a_t, we split the quadrangle by its shortest diagonal.

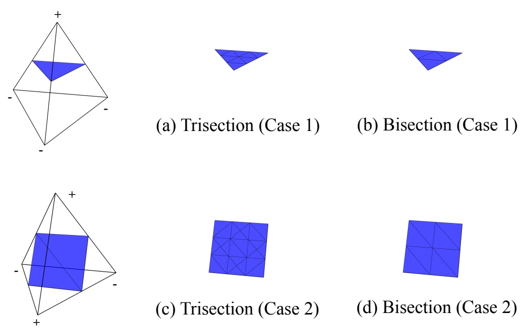


Figure 5: *Subdivided facets of the isosurface. Left: two configurations of a tetrahedron containing the isosurface. Middle: trisection refinement. Right: bisection refinement. For case 1 (1/3 vertices), the number of triangles is multiplied by a factor 9 after trisection refinement (a) and by 4 after bisection refinement (b). For case 2 (2/2 vertices), the number of triangles is multiplied by a factor between 9 and 15 after trisection refinement (c) and by 4 after bisection refinement (d).*

Trisection Refinement. Given a tetrahedron t to be refined, we insert the trisection points of each edge and the centroid of each face into the domain triangulation (see Figure 6(b)). While for case 1 the isofacet is always divided into 9 sub-triangles (Figure 5(a) and 6(e)), it is not true for case 2, in which the number of sub-triangles depends on the shape of t. Empirically, the number is between 9 and 17, with an average 12, which we use to define our refinement algorithm.

Bisection Refinement. We insert the midpoint of each edge with same value signs (Figure 6(c)(d)). In both cases, the isofacet is divided into 4 sub-triangles (Figure 5(g-h)).

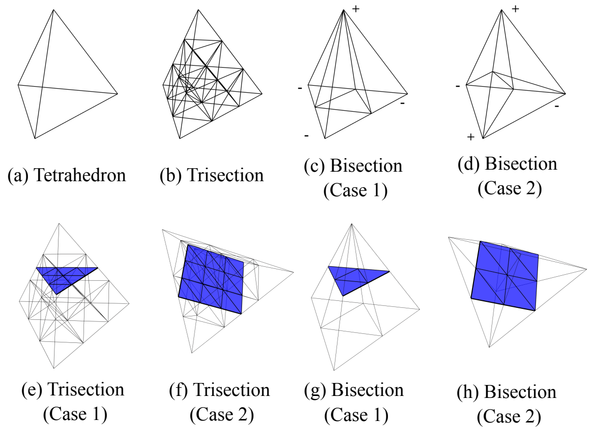


Figure 6: *Refinement schemes. (a) Tetrahedron before refinement. (b) Refinement by crossing edge trisection: we insert three points uniformly sampled on each edge and the centroid of each face. (c-d) Refinement by bisection: we insert the midpoint of each edge with similar value signs (two cases). (e-f) Refined isosurfaces for both cases after trisection refinement. (g-h) Refined isosurfaces for both cases after bisection refinement.*

Refinement Strategy. The adaptive mesh refinement step utilizes the above refinement schemes as follows. Consider a tetrahedron t intersecting the isosurface into a facet f_t. We evaluate the target area at the centroid of f_t from the input sizing field, denoted as $s(f_t)$. The area ratio r_t is defined as the ratio between the current area of f_t and the target area $s(f_t)$. If r_t is smaller than 1, we deduce that the local sizing criterion is already satisfied, and we do not refine t. Otherwise, we check the following conditions and adopt the following schemes:

1. If t belongs to case 1 and $r_t > 9$, we trisect t
2. If t belongs to case 2 and $r_t > 12$, we trisect t
3. If t belongs to case 1 and $4 \leq r_t < 9$, we bisect t
4. If t belongs to case 2 and $4 \leq r_t < 12$, we bisect t
5. If $1 < r_t < 4$, we bisect t with a certain probability

When $1 < r_t < 4$, we expect r_t to approach 1 while avoiding over-refining t. Suppose we have N tetrahedra intersecting isosurface with average area $\overline{a}$ before refinement. We bisect u percent of tetrahedra and the average area after the refinement is $\overline{a}'$. In order to preserve the total area of the isosurface, we have:

$$\overline{a}' = \frac{N\overline{a}}{4uN + (1-u)N} \quad \rightarrow \quad u = \frac{\overline{a} - \overline{a}'}{3\overline{a}'}$$

For a specific tetrahedron t, we evaluate a uniform random variable $c \in [0,1]$ and bisect the tetrahedron only if $c < \frac{1}{3}(r_t - 1)$. Otherwise we do not refine it.

Adaptive Sizing Field. While the target area is intuitive to deduce for a uniform sizing field, the adaptive sizing field remains a tricky problem. Users can come with their own sizing fields, but we provide a reasonable choice guided by the curvature of the surface. Our sizing field is dense near the region with high curvatures and sparse otherwise.

We start by evaluating the minimum curvature $c_{\min}$ and the maximum curvature $c_{\max}$ onto the input point cloud $\mathcal{X}$ by monge form. Then each time before beginning the refinement, we construct a smoothed curvature map for tetrahedra intersecting isosurface. Given a tetrahedron t intersecting isosurface, we consider the k-nearest neighbors of the centroid of a_t in $\mathcal{X}$ and estimate the local curvature by computing the maximum absolute value of the two principal curvatures of the neighbors. This curvature map is smoothed following the same process of the volume field in ASAP energy and then clipped between $[c_{\min}, c_{\max}]$.

When refining a tetrahedron t, we deduce the target sizing from its smoothed curvature c. A mapping function is applied to c in order that users can control the gradation of the sizing field.

$$c' = \left(\frac{c - c_{\min}}{c_{\max} - c_{\min}}\right)^{\lambda} \cdot (c_{\max} - c_{\min}) + c_{\min}$$

Given d a user-controlled parameter which defines the expected distance tolerance from the isosurface to the point cloud, the target sizing s is computed by:

$$s = 4\sqrt{3} \cdot \left(\frac{2d}{c} - d^2\right)$$

Last, s is clipped between $[s_{\min}, s_{\max}]$, which guides the refinement described above.

3.5. Solvers

We now detail our framework at work using global implicit reconstruction solvers applied to tetrahedron meshes.

Discretization Elements. We use piecewise-linear basis functions $\{\phi_i\}_i$, $\phi_i(v_j) = \delta_i^j$ for all vertices v_j of $\mathcal{T}$). For a point x inside a tetrahedron $t = (t^0, t^1, t^2, t^3)$, $\{\phi_{t^k}\}_{k=0}^3(x)$ are its barycentric coordinates, and given scalar values $\{f_i\}$ associated with the vertices $\{v_i\}$ of $\mathcal{T}$, $f(x)$ is defined as $f(x) = \sum_{k=0}^3 \phi_{t^k}(x) f_{t^k} = \sum_i \phi_i(x) f_i$. Similarly, the gradient of f at x is defined as $\bigtriangledown f(x) = \sum_{k=0}^3 \bigtriangledown\phi_{t^k}(x) f_{t^k}$. We note $G \in \mathbb{R}^{3|\mathcal{T}|\times|\mathcal{V}|}$ the gradient matrix whose $(3t, 3t+1, 3t+2)$ rows contain the four gradients of the barycentric coordinate functions of the tetrahedron t, Div the (integrated) Divergence matrix, $L \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ the (integrated) Laplacian matrix, $B := LM_{\mathcal{V}}^{-1}L \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ the (integrated) bi-Laplacian matrix, and $H \in \mathbb{R}^{9|\mathcal{V}|\times|\mathcal{V}|}$ the (integrated) Hessian matrix. We use the construction presented by Stein et al. [SGWJ]: Noting

$G_x, G_y, G_z \in \mathbb{R}^{|\mathcal{T}|\times|\mathcal{V}|}$ the gradient matrices of the (x,y,z) coordinates, H is defined as

$$H := \tilde{D}^T \tilde{M}_{\mathcal{T}} \tilde{G}, \text{ with} \tag{11}$$
$$\tilde{G} := \begin{bmatrix} G_x \\ G_y \\ G_z \end{bmatrix}, \tilde{D} := \begin{bmatrix} G_x & G_y & G_z & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & G_x & G_y & G_z & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & G_x & G_y & G_z \end{bmatrix}$$

L, denoting the integrated Laplacian matrix, is defined using the standard cotangent formula as:

$$\begin{cases} L_{ij} & = \sum_{t\ni(i,j)} l_{ij} \cot(\gamma^t_{ij})/6 \qquad \forall j \in N_1(i) \\ L_{ii} & = -\sum_{j\neq i} L_{ij} \end{cases} \tag{12}$$

where l_{ij} denotes the length of edge (i,j), and γ^t_{ij} denotes the dihedral angle opposite to edge (i,j) in the tetrahedron t.

Screened Poisson Solver. We minimize the objective function:

$$\mathcal{E}_{SP} := \int_{\mathcal{T}} \| \bigtriangledown f(x) - \tilde{n}(x) \|^2 dx + \frac{\alpha}{|\mathcal{X}|} \sum_p f(p)^2 \rightarrow \text{min}, \tag{13}$$

where $\tilde{n}(x)$ denotes a smooth approximation of the normal field n defined on $\mathcal{X}$. Note that it is common to shift the solution after minimization, in order to find the isosurface best approximating the input samples $\mathcal{X}$: $f \leftarrow f - \text{median}(\{f(p)\}_{p\in\mathcal{X}})$. If α is set to 0, one obtains the original Poisson solver, in which case it is necessary to add at least one constraint as otherwise the system is underconstrained. Minimizing this objective function on tetrahedron meshes amounts to solving the following linear system:

$$\left[L + \frac{\alpha}{|\mathcal{X}|} \Phi^T_{\mathcal{X}} \Phi_{\mathcal{X}} \right] F = \text{Div}\tilde{N}, \tag{14}$$

where $\Phi_{\mathcal{X}}$ denotes the matrix stacking the barycentric coordinates of $\mathcal{X}$.

Smooth Signed Distance Solver. We minimize the objective function:

$$\mathcal{E}_{SSD} := \frac{\alpha}{|\mathcal{X}|} \sum_p f(p)^2 + \frac{\beta}{|\mathcal{X}|} \sum_p \| \bigtriangledown f(p) - n_p \|^2 + \frac{\gamma}{|\mathcal{T}|} \int_{\mathcal{T}} \|Hf(x)\|^2 dx \rightarrow \text{min}. \tag{15}$$

Minimizing this objective function on tetrahedron meshes amounts to solving the following linear system:

$$\left[\frac{\alpha}{|\mathcal{X}|} \Phi^T_{\mathcal{X}} \Phi_{\mathcal{X}} + \frac{\beta}{|\mathcal{X}|} G^T S^T_{\mathcal{X}} S_{\mathcal{X}} G + \frac{\gamma}{|\mathcal{T}|} H^T \tilde{M}^{-1}_{\mathcal{V}} H \right] F = \frac{\beta}{|\mathcal{X}|} G^T S^T_{\mathcal{X}} N_{\mathcal{X}},$$

where $N_{\mathcal{X}} \in \mathbb{R}^{3|\mathcal{X}|}$ stacks the normals of $\mathcal{X}$, $\tilde{M}_{\mathcal{V}} \in \mathbb{R}^{9|\mathcal{V}|\times 9|\mathcal{V}|}$ denotes the mass matrix of the vertices repeated 9 times along the diagonal, and $S_{\mathcal{X}}$ denotes the selection matrix whose $(3i, 3i+1, 3i+2)$ rows contain the Identity matrix at columns $(3t, 3t+1, 3t+2)$ if vertex i lies inside the tetrahedron t.

Spectral Solver. We minimize the objective function:

$$\mathcal{E}_{Spec} := \sum_p \bigtriangledown f(p)^T \cdot C_p \cdot \bigtriangledown f(p) \rightarrow \text{max, s.t.} \tag{16}$$
$$\frac{\alpha}{|\mathcal{X}|} \sum_p f(p)^2 + \beta \int_{\mathcal{T}} \| \bigtriangledown f(x) \|^2 dx + \gamma \int_{\mathcal{T}} \triangle f(x)^2 dx = 1$$

Maximizing this objective function on tetrahedron meshes amounts to finding the largest eigenvalue and related eigenvector of the following generalized eigenvalue problem:

$$\left[G^T S^T_{\mathcal{X}} C_{\mathcal{X}} S_{\mathcal{X}} G \right] F = \lambda \left[\frac{\alpha}{|\mathcal{X}|} \Phi^T_{\mathcal{X}} \Phi_{\mathcal{X}} + \beta L + \gamma B \right] F, \tag{17}$$

where $C_{\mathcal{X}}$ denotes the block diagonal matrix whose i^{th} block is the anisotropy matrix $C_i := I + \mu n_i n_i^T$, n_i denotes the (unoriented) normal of input point $p_i \in \mathcal{X}$ and μ controlling the anisotropy favoring alignment between the gradient and the unoriented normal at p_i. Compared to the above solvers, this one is oblivious to the orientation of the input normals. This comes at the cost of solving a generalized eigenvalue solver to solve for signing the implicit function.

4. Experiments

Our framework is implemented in C++, using the CGAL library for 3D triangulations and geometric computations [The21], the Eigen library for linear algebra and solvers [GJ*10], the Spectra library for solving generalized eigenvalue problems [Qiu21] and OpenMP for multithreading acceleration [DM98]. The experiments are conducted on a MacBook Pro with 2,9 GHz Quad-Core Intel Core i7 CPU and 16GB memory.

4.1. Adaptivity

Figure 7 depicts the progressive reconstruction process on the *Massai* point set. The optimizer and the refiner jointly improve the quality of the discretized domain and isosurface mesh. The discretized domain is getting denser and denser while sandwiching the reconstructed isosurface. The ratio between the number of vertices near the isosurface and of the entire triangulation increases rapidly, showing that we allocate more degrees of freedom where needed. The implicit functions depicted in a cutting plane highlight that the triangulation is denser around the high curvature area, which helps reducing the interpolation error.

4.2. Progressive Refinement

We validate the relevance of the proposed progressive domain approach, by comparing four different approaches for generating the discretized geometric domain and reconstructing the output isosurface:

1. **Octree-based initialization**: we discretize the domain by inserting into the 3D triangulation all nodes of a dense octree refined from the local density of the input point set.

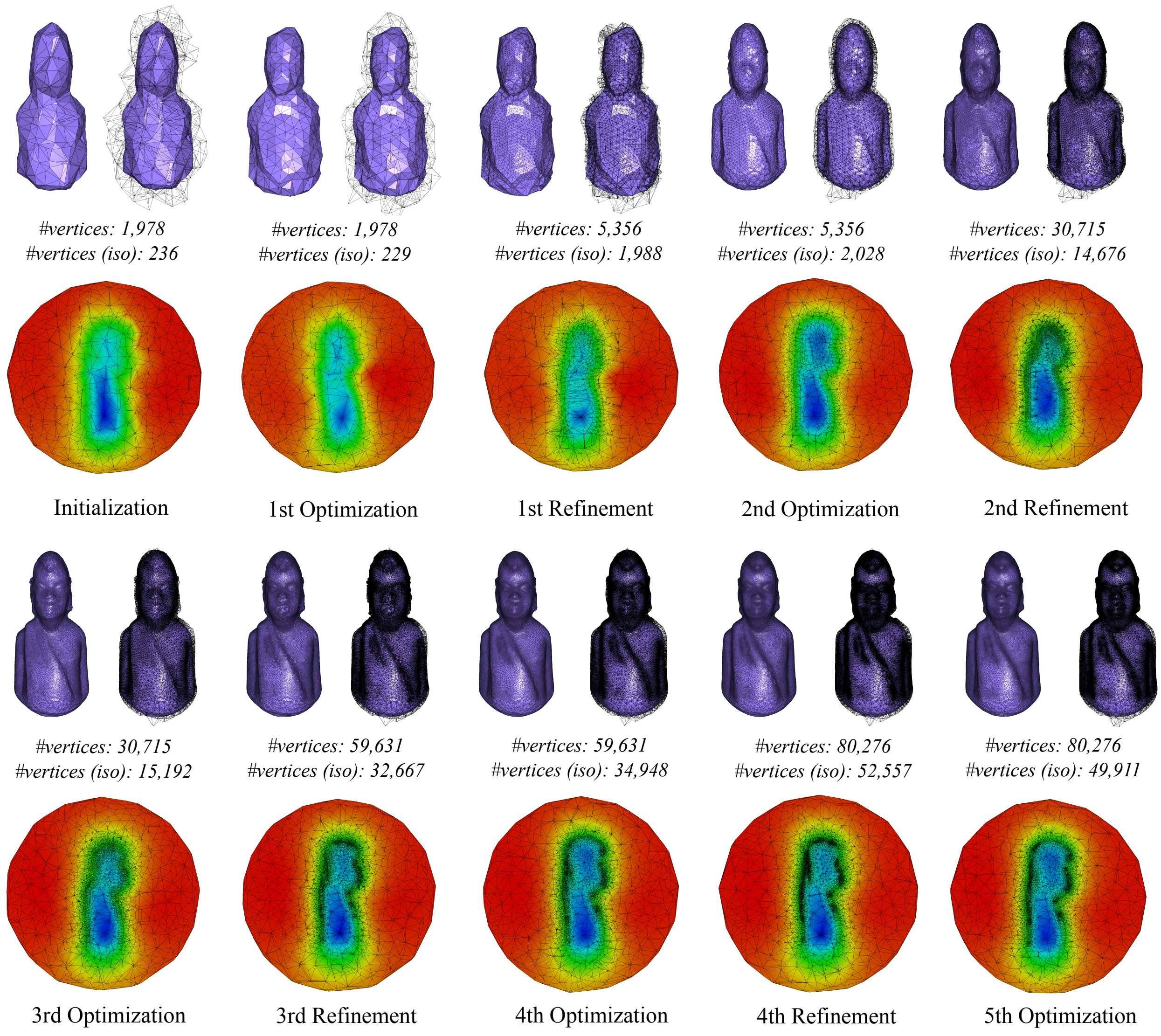


Figure 7: *Reconstruction process for the Massai Model. The first and third rows depict the isosurface and the tetrahedra intersected by the isosurface during the reconstruction process. The second and fourth rows depict the implicit function clipped by a plane.*

2. **Input-based initialization**: we first insert all input points into the 3D triangulation, then perform dense Delaunay refinement inside a loose bounding box of the input point set until all tetrahedra are well-shaped.
3. **Direct Refinement**: we initialize the 3D triangulation by a sparse point set sampled on the loose bounding box of the input point set, followed by Delaunay refinement until all tetrahedra are well-shaped. We then launch the solver to obtain an initial isosurface and refine the 3D triangulation by using our refinement process until the sizing criteria is satisfied, without using any optimizer or solver. We run a final solver to generate the final isosurface.
4. **Progressive Refinement**: we perform the proposed progressive algorithm through "solver-optimizer-refiner" iterations.

Our goal is not to evaluate the output result by common criteria such as e.g. the average distance from the points to the reconstructed surface, or deviation of normals, etc. Instead, we wish to verify the relevance of our domain discretization, given a solver and its regularization parameters. Once combined, they trade data fidelity for regularization, and can thus deviate largely from perfect data fidelity.

In order to evaluate and compare the results, we compute a "ground truth" reference isosurface from a point set, by running the given solver and regularization parameters on a densely discretized domain. We generate the domain as follows: (1) Compute the isosurface using a dense octree-based SSD (2) Sample very densely the isosurface, duplicate the resulting point set and offset the two point sets along the negative and positive local normal directions

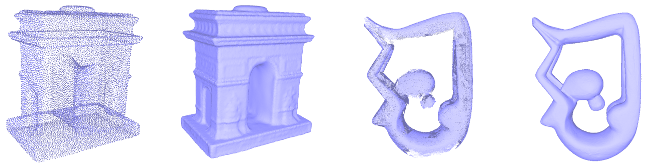

(a) Point Cloud Arc de Triomphe (b) Reference Arc de Triomphe (c) Point Cloud Indonesian Lady (d) Reference Indonesian Lady

Figure 8: *Two point clouds and their corresponding reference reconstructed surfaces computed on a very dense triangulation.*

Table 1: *Reconstructing the Arc de Triomphe and Indonesian Lady models*

Measures		Methods				
		Octree	Input	Direct	Progressive	Reference
Arc	Hausdorff	1.622006	2.463567	3.124351	1.504585	-
	Mean (Method - Ref)	0.054016	0.085333	0.053248	0.033301	-
	Mean (Ref - Method)	0.049102	0.080371	0.048665	0.031906	-
	RMS (Method - Ref)	0.093774	0.184979	0.111763	0.055931	-
	RMS (Ref - Method)	0.080871	0.162403	0.08594	0.051933	-
	Timing	78.27	805.42	758.92	2532.42	31493.71
	# Isovertices (1)	116232	53340	232447	331968	741211
	# Vertices around iso (2)	57643	24361	115574	153514	364041
	# Vertices (3)	117160	309478	238417	231673	1419823
	Parsimony (4)	0.492002	0.078716	0.484756	0.662632	0.256399
Indonesian Lady	Hausdorff	0.024654	0.036668	0.041566	0.022067	-
	Mean (Method - Ref)	0.000430	0.000838	0.002920	0.000596	-
	Mean (Ref - Method)	0.000612	0.001235	0.002925	0.000588	-
	RMS (Method - Ref)	0.001444	0.001785	0.005649	0.001324	-
	RMS (Ref - Method)	0.001801	0.002647	0.005501	0.001185	-
	Timing	615.17	1222.79	702.13	1502.67	27806.79
	# Isovertices (1)	249969	1576551	88207	312517	455312
	# Vertices around iso (2)	123723	754457	46928	144366	229912
	# Vertices (3)	280335	974507	272648	254091	1947880
	Parsimony (4)	0.441340	0.774194	0.172119	0.568167	0.118032

(1) Number of vertices on the remeshed isosurface. (2) Number of triangulation vertices around the isosurface. (3) Number of vertices of the entire triangulation. (4) Parsimony $= (2)/(3)$.

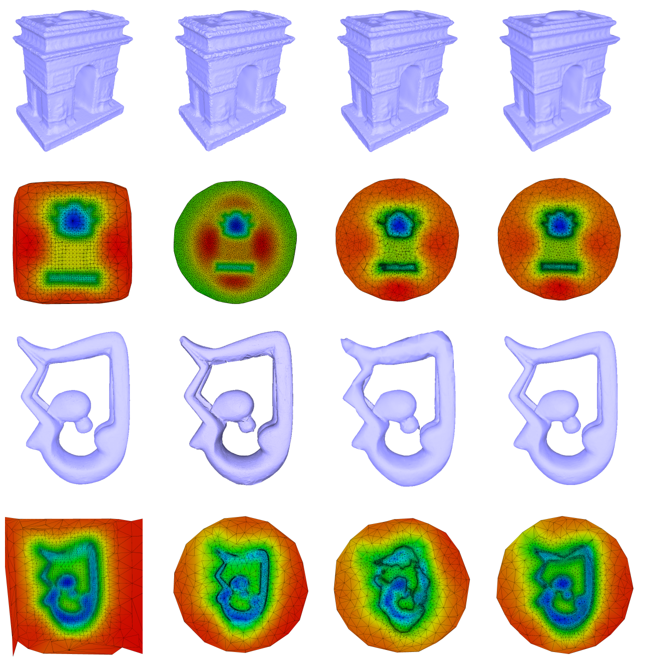


Figure 9: *Reconstructed surfaces and discretized domains of the four above approaches, clipped by a cutting plane. We utilize the SSD solver for both point clouds and use a uniform sizing field to guide the refinement. Our optimized domain adapts to the isosurface.*

(3) Insert all offset points into a 3D Delaunay triangulation (4) Optimize the 3D triangulation by minimizing the ASAP energy, and (5) Solve via SSD to generate the reference isosurface.

Figure 8 shows the selected point clouds and their corresponding ground truth isosurfaces. Figure 9 and Table 1 depict and record the four aforementioned reconstructions and relating statistics. For both input point sets, using our progressive approach we obtain more than 55% vertices of the triangulation near the final isosurface, which validates the parsimony of our approach. We compare the 4 above approaches in terms of (1) the output surface (2) the discretized domain (3) the distances from the reconstructed surface to the reference surface computed using Metro [CRS98], (4) the parsimony, defined as the number of vertices adjacent to tetrahedra intersecting the isosurface, divided by the total number of vertices of the triangulation.

4.3. Robustness

Different Densities. We sample the kitten model with different densities and compare the results of our algorithm using the SSD solver and uniform area criteria with the results of the original octree-based SSD algorithm, see Figure 10.

Variable Resolution. We generate a point set of the Kitten with two different resolutions: dense on the head and sparse elsewhere. Figure 10 compares our optimized one with the original SSD.

Noise. We compare our results with the original octree-based SSD on a model with increasing levels of noise ($\sigma \in \{0.005, 0.01, 0.05, 0.1\}$). The results are shown by Figure 11. When increasing σ, the original SSD becomes more and more sensitive to solver parameters and the isosurface becomes less and less smooth.

Holes. We verify the capability of our approach to fill holes on the two Indonesian models, see Figure 12, which are two laser scans with large holes and imperfect normals. Albeit filling holes is an ill-posed problem, our method seems to fill the holes more gracefully. The artifacts of the octree-based SSD is getting more evident when the depth of octree increases, while there are no such artifacts when we solve SSD on an optimized domain. Note that these artifacts (bumps with high curvature variations) are in contradiction with what is expected from an SSD solution on a smooth input with strong Hessian penalization, indicating that the allocated structure prevented the solver to obtain an accurate result, which an a posteriori remeshing approach would not help fixing.

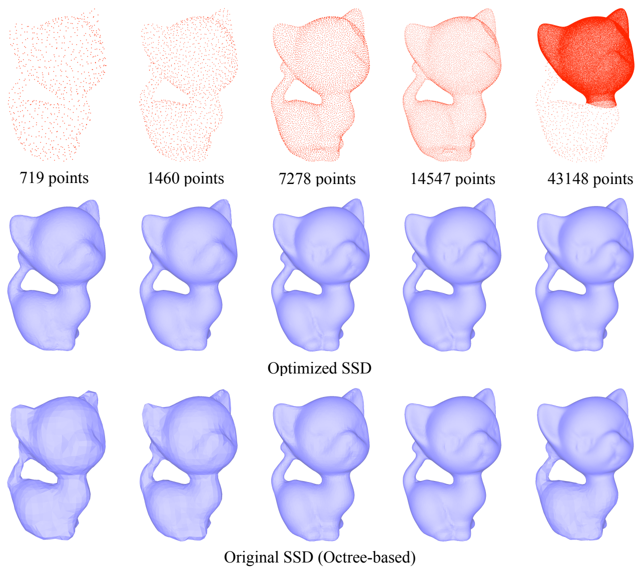


Figure 10: *Reconstruction results of Kitten model with different sampling densities. Top row: input point clouds. Middle row: SSD solved on our optimized geometric domain. Bottom row: SSD solved on an octree. It fails to discover more details when the point cloud is sparse.*

Table 2: *Reconstructing the Tiki model. Performing several iterations of ASAP optimizations results in faster solver convergence rates, indicating empirically that our optimization improves the conditioning of the solver for a fixed number of vertices.*

Num of ASAP Optimization	0	3	5	10
Num of Solver Iterations	13542	8403	2944	2444
Solver Error	8.67667e-11	9.88958e-11	9.96412e-11	9.94334e-11
Solver Time (s)	2.82103	1.59871	0.569161	0.46091

4.4. Solver Conditioning

The ASAP energy is the key component for improving the solver conditioning. It improves the quality of the tetrahedral elements in the triangulation to achieve this goal. We start by comparing the number of iterations to make the linear solver attain a fixed tolerance error (1e-10) before and after a pure ASAP optimization (without mid-edge energy). All the solvers begin with a zero vector as a guess solution to make it a fair comparison. From Table 2 and Figure 13, we observe that the number of iterations decreases and the quality of the triangulation tetrahedra is improved through the optimization step.

Combining with mid-edge energy, the optimizer makes a trade-off between the quality of the triangulation and the quality of the remeshed isosurface. In practice, we find that $\lambda_a = \lambda_m = 3$ is a good choice for most of the cases. For challenging cases, for example, when the point cloud has many salient features, λ_m can be increased to prevent the failure of the solver.

4.5. Ablation Study

We show the impact of the optimizer by removing one or several components from our approach and compare the produced isosurfaces on the *Horse* point cloud. The following options are tested:

1. **Non-optimized strategy**: Iterating over solver and refiner without any optimization. One step of ASAP optimization is applied before the solver, otherwise the solver fails to converge.
2. **Mid-edge strategy**: Iterating over solver, optimizer and refiner, while disabling ASAP energy. One step of ASAP optimization is applied before the solver, otherwise the solver fails to converge.
3. **ASAP strategy**: Iterating over solver, optimizer and refiner, while disabling mid-edge energy.

We notice that the ASAP energy is indispensable to the convergence of the solver. The reconstructed isosurfaces and the histogram of their qualities are shown in Figure 14. Together with mid-edge energy, they improve a lot the quality of the isosurface, eliminate the influence of outliers and fill the holes with a smooth surface.

4.6. Timings

Figure 15 records the execution times for each model shown. Each color represents one iteration and the final pink color indicates the time for the final solver, which produces the final isosurface on the optimized geometric domain. Compared to octree-based solvers,

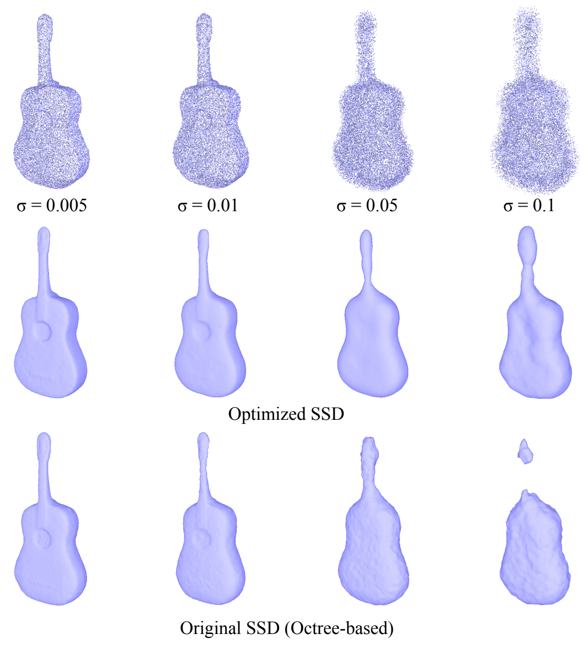


Figure 11: *Reconstruction results of the Guitar model with different noise levels. Top row: input point clouds. Middle row: SSD solved on our optimized geometric domain. Bottom row: SSD solved on an octree. The parameters of the original SSD are chosen to make the isosurface as smooth as possible.*

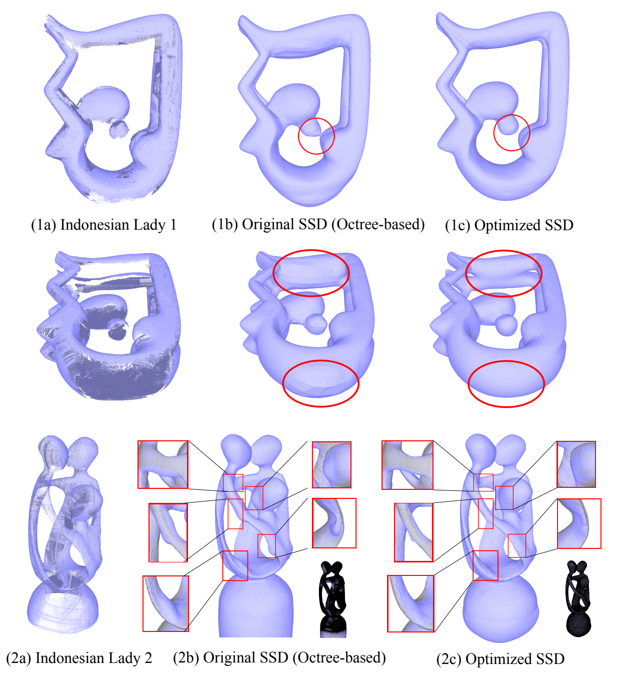


Figure 12: *Reconstructing the Indonesian Lady models, the first has large holes between the legs and near the stomach. The second has many small holes.*

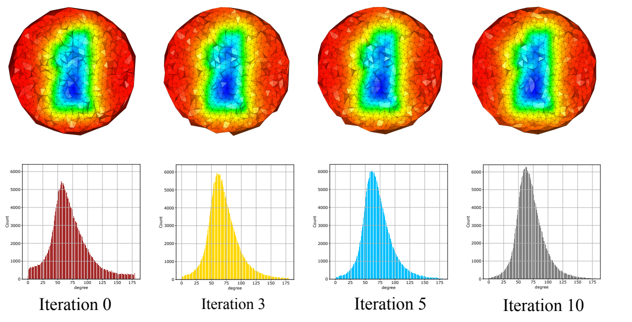


Figure 13: *Reconstructing the Tiki model. The first row shows the clipped domain at iteration {0, 3, 5, 10} and the second row plots the distribution of dihedral angles of the triangulation tetrahedra.*

our algorithm takes more time. However, it is scaling fairly well with the number of input points, and the execution time mostly depends on the target sizing field.

4.7. More Results

Figure 16 depicts a gallery of reconstructed surfaces. (1) is a very sparse point set with oriented normals (from [HCJ19]). Common octree-based methods cannot produce a smooth surface since the octree is refined only near the input point set, while our progressive approach yields smooth surfaces. (2)-(7) depict other scanned point sets with holes and noise. Our progressive approach reconstructs

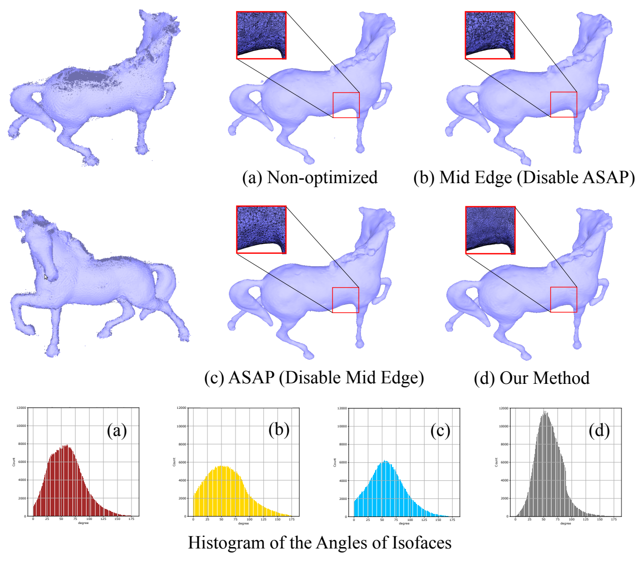

(a) Non-optimized
(b) Mid Edge (Disable ASAP)
(c) ASAP (Disable Mid Edge)
(d) Our Method
(a)
(b)
(c)
(d)
Histogram of the Angles of Isofaces

Figure 14: *Reconstruction results of Horse model by disabling one or several components of the proposed approach. The third row plots the distribution of the angles of all isofacets (triangles).*

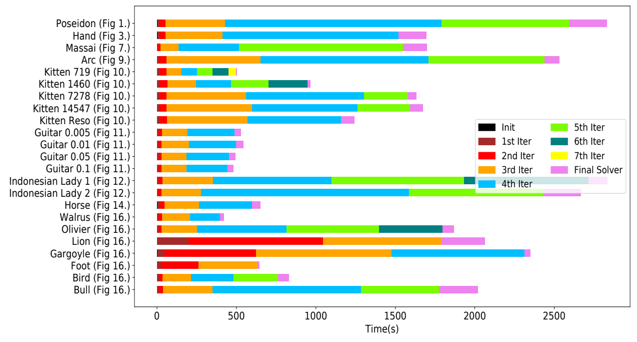


Figure 15: *Timeline of our reconstruction for all models shown. Each color corresponds to one iteration and pink corresponds to the final solver running on the optimized domain.*

smooth surfaces in accordance to the regularity parameter of the selected solver.

5. Discussion

5.1. Failure Cases

With no a priori knowledge about the curvature and local feature size (lfs) of the inferred isosurface, our approach may fail to reconstruct fine details due to insufficient discretization. Figure 17 depicts one failure case of our discretized domain. The book in the hand of *Ignatius* is a region with two layers of points with opposite normals. In order to reconstruct it correctly, this region should be densely refined to offer enough degrees of freedom. Given that the initial solution does not capture this region, it is not sufficiently

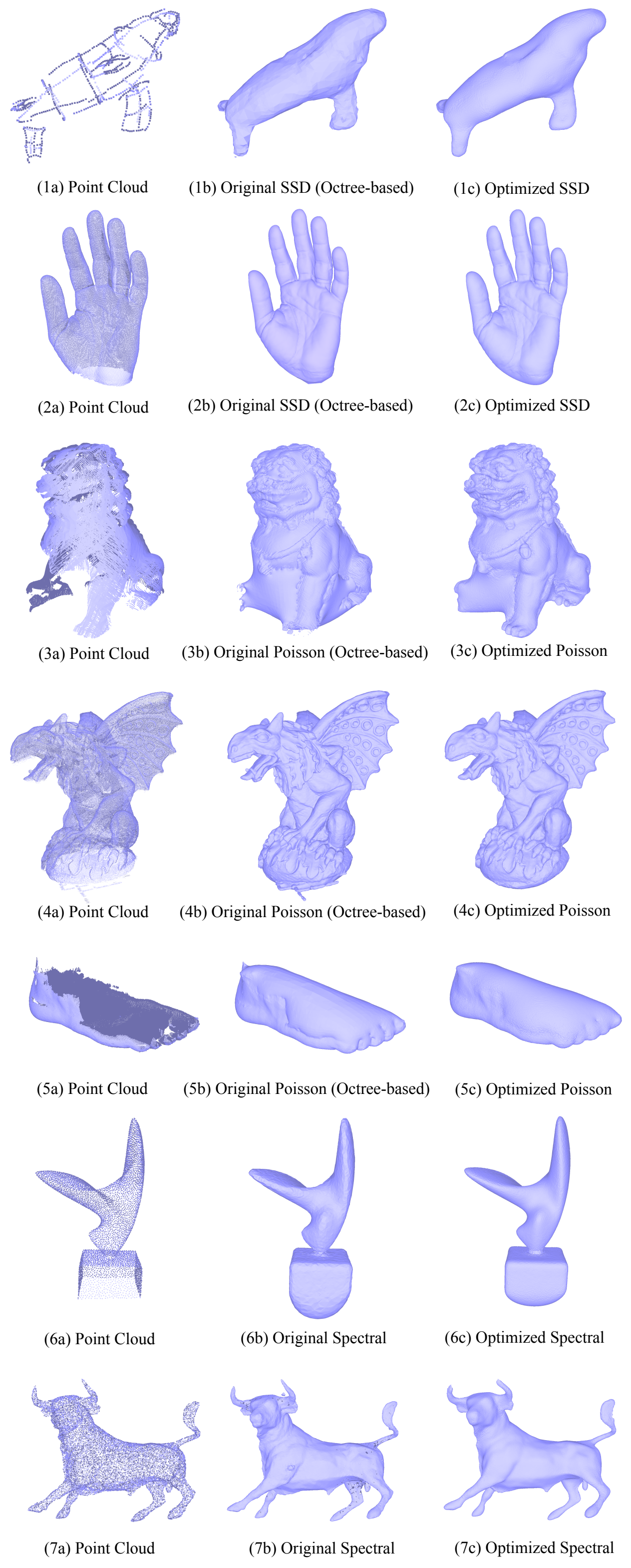


Figure 16: *Reconstruction gallery. We compare our reconstruction results (right) with the one solved using common input-sensitive approaches (middle).*

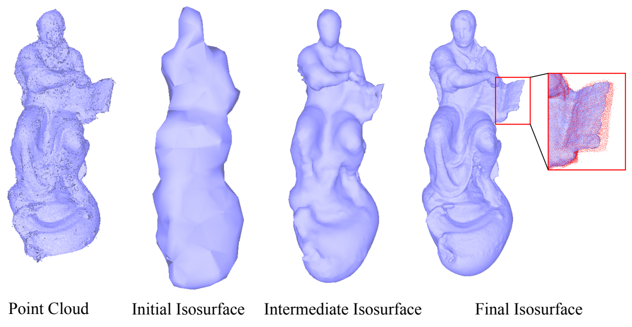


Figure 17: *Reconstruction process for the Ignatius Model. The final isosurface fails to completely reconstruct the book.*

refined. As the reconstruction goes on, the region gets denser, but this is insufficient to yield a good solution around this region.

5.2. Limitations and Future work

Our current approach presents several limitations. First, the ability of our method to discover salient geometric events - and adapt the reconstruction domain to them - remains bounded to the actual performance of the underlying reconstruction algorithm used at each iteration. Second, deriving sufficiency conditions for ensuring convergence remains to be done, and we believe that per-solver approaches could first be designed before addressing the more generic case. Third, our approach focuses on improving the quality of the solvers' outputs at the cost of longer execution timings, but we envision that simple strategies could be used to reduce those. For instance, we rely on Delaunay triangulations, whose structures can change unexpectedly when relocating vertices, requiring the solvers' algebraic structures to be updated accordingly even with a fixed vertex count. Adopting other tetrahedron mesh structures could help addressing this problem, while lowering the amount of slivers present in the triangulation, thus improving the conditioning of the solvers. Last, we discretized our solvers using piecewise linear elements. In future work we plan to explore a higher-order variant of this approach, in which the tetrahedron elements of the domain are the support of a non-linear implicit function. We also wish to address piecewise-smooth surfaces with boundaries and non-manifold features.

5.3. Conclusion

We proposed a progressive domain approach for global implicit surface reconstruction methods. Given an initial 3D Delaunay triangulation of the domain and an implicit solver, our approach iterates over three main steps (solve, optimization, adaptive refinement), all steps being designed to cooperate with each other and improve the conditioning of the solver, and the quality and complexity-distortion tradeoff of the output isosurface mesh. In such a progressive approach, the implicit solver is no longer used once, but iteratively as a means to discover more and more details for the isosurface. The benefit is to reconstruct and generate altogether output meshes with well-shaped triangles and adapted to the intrinsic geometric complexity of the reconstructed surface, instead of to the input point set density, as in previous work.

6. Acknowledgments

This work has been supported by the French government, through the 3IA Côte d'Azur Investments in the Future project managed by the National Research Agency (ANR) with the reference number ANR-19-P3IA-0002.

References

[ACTD07] ALLIEZ P., COHEN-STEINER D., TONG Y., DESBRUN M.: Voronoi-based variational reconstruction of unoriented point sets. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing, Barcelona, Spain, July 4-6, 2007* (2007), Belyaev A. G., Garland M., (Eds.), vol. 257 of *ACM International Conference Proceeding Series*, Eurographics Association, pp. 39–48. 2, 4

[BKK20] BRANDTS J., KOROTOV S., KŘÍŽEK M.: *Refinement Techniques*. Springer International Publishing, Cham, 2020, pp. 87–114. 3

[BTS*17] BERGER M., TAGLIASACCHI A., SEVERSKY L. M., ALLIEZ P., GUENNEBAUD G., LEVINE J. A., SHARF A., SILVA C. T.: A survey of surface reconstruction from point clouds. *Comput. Graph. Forum 36*, 1 (2017), 301–329. 2

[CDS13] CHENG S., DEY T. K., SHEWCHUK J. R.: *Delaunay Mesh Generation*. Chapman and Hall / CRC computer and information science series. CRC Press, 2013. 3

[CRS98] CIGNONI P., ROCCHINI C., SCOPIGNO R.: Metro: measuring error on simplified surfaces. In *Computer graphics forum* (1998), vol. 17, Wiley Online Library, pp. 167–174. 10

[CT11] CALAKLI F., TAUBIN G.: SSD: smooth signed distance surface reconstruction. *Comput. Graph. Forum 30*, 7 (2011), 1993–2002. 2, 4

[CZZ*17] CHEN J., ZHENG J., ZHENG Y., XIAO Z., SI H., YAO Y.: Tetrahedral mesh improvement by shell transformation. *Eng. Comput. 33*, 3 (2017), 393–414. 2

[Dey06] DEY T. K.: *Curve and Surface Reconstruction: Algorithms with Mathematical Analysis*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2006. 2

[DM98] DAGUM L., MENON R.: Openmp: an industry standard api for shared-memory programming. *Computational Science & Engineering, IEEE 5*, 1 (1998), 46–55. 8

[GBA*17] GEORGE P. L., BOROUCHAKI H., ALAUZET F., LAUG P., LOSEILLE A., MARCUM D., MARÉCHAL L.: *Mesh Generation and Mesh Adaptivity: Theory and Techniques*. American Cancer Society, 2017, pp. 1–51. 3

[GJ*10] GUENNEBAUD G., JACOB B., ET AL.: Eigen v3, 2010. URL: http://eigen.tuxfamily.org. 8

[HCJ19] HUANG Z., CARR N., JU T.: Variational implicit point set surfaces. *ACM Trans. Graph. 38*, 4 (2019), 124:1–124:13. 12

[HT20] HASS J., TRNKOVA M.: Approximating isosurfaces by guaranteed-quality triangular meshes. *Comput. Graph. Forum 39*, 5 (2020), 29–40. 3

[JWZ11] JIAO X., WANG D., ZHA H.: Simple and effective variational optimization of surface and volume triangulations. *Eng. Comput. 27*, 1 (2011), 81–94. 2

[KBH06] KAZHDAN M. M., BOLITHO M., HOPPE H.: Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing, Cagliari, Sardinia, Italy, June 26-28, 2006* (2006), Sheffer A., Polthier K., (Eds.), vol. 256 of *ACM International Conference Proceeding Series*, Eurographics Association, pp. 61–70. 2, 4

[KCRH20] KAZHDAN M., CHUANG M., RUSINKIEWICZ S., HOPPE H.: Poisson surface reconstruction with envelope constraints. *Comput. Graph. Forum 39*, 5 (2020), 173–182. 2

[KH13] KAZHDAN M. M., HOPPE H.: Screened poisson surface reconstruction. *ACM Trans. Graph. 32*, 3 (2013), 29:1–29:13. 2, 4

[Kli08] KLINGNER B. M.: *Improving Tetrahedral Meshes*. PhD thesis, EECS Department, University of California, Berkeley, Nov 2008. 2

[LCS09] LIU J., CHEN Y., SUN S.: Small polyhedron reconnection for mesh improvement and its implementation based on advancing front technique. *International journal for numerical methods in engineering 79*, 8 (2009), 1004–1018. 2

[MdGD*10] MULLEN P., DE GOES F., DESBRUN M., COHEN-STEINER D., ALLIEZ P.: Signing the unsigned: Robust surface reconstruction from raw pointsets. *Comput. Graph. Forum 29*, 5 (2010), 1733–1741. 2

[MON*19] MESCHEDER L. M., OECHSLE M., NIEMEYER M., NOWOZIN S., GEIGER A.: Occupancy networks: Learning 3d reconstruction in function space. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019* (2019), Computer Vision Foundation / IEEE, pp. 4460–4470. 2

[MWA*13] MUSIALSKI P., WONKA P., ALIAGA D. G., WIMMER M., GOOL L. V., PURGATHOFER W.: A survey of urban reconstruction. *Comput. Graph. Forum 32*, 6 (2013), 146–177. 2

[OBA*03] OHTAKE Y., BELYAEV A. G., ALEXA M., TURK G., SEIDEL H.: Multi-level partition of unity implicits. *ACM Trans. Graph. 22*, 3 (2003), 463–470. 2

[Qiu21] QIU Y.: Spectra, 2021. URL: https://github.com/yixuan/spectra. 8

[SA07] SORKINE O., ALEXA M.: As-rigid-as-possible surface modeling. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing, Barcelona, Spain, July 4-6, 2007* (2007), Belyaev A. G., Garland M., (Eds.), vol. 257 of *ACM International Conference Proceeding Series*, Eurographics Association, pp. 109–116. 5

[SGWJ] STEIN O., GRINSPUN E., WARDETZKY M., JACOBSON A.: Natural boundary conditions for smoothing in geometry processing. *ACM Trans. Graph. 37*, 2, 23:1–23:13. 7

[She02] SHEWCHUK J.: What is a good linear finite element? interpolation, conditioning, anisotropy, and quality measures (preprint). *University of California at Berkeley 73* (2002), 137. 2

[Si15] SI H.: Tetgen, a delaunay-based quality tetrahedral mesh generator. *ACM Trans. Math. Softw. 41*, 2 (2015), 11:1–11:36. 3

[SLS*06] SHARF A., LEWINER T., SHAMIR A., KOBBELT L., COHEN-OR D.: Competing fronts for coarse-to-fine surface reconstruction. *Comput. Graph. Forum 25*, 3 (2006), 389–398. 2

[The21] THE CGAL PROJECT: CGAL user and reference manual, 2021. URL: https://doc.cgal.org/5.2.1/Manual/packages.html. 8

[VWP13] VARTZIOTIS D., WIPPER J., PAPADRAKAKIS M.: Improving mesh quality and finite element solution accuracy by getme smoothing in solving the poisson equation. *Finite elements in analysis and design 66* (2013), 36–52. 2