4D Space-Time Delaunay Meshing for Medical Images

*

Panagiotis Foteinos · Nikos Chrisochoides

the date of receipt and acceptance should be inserted later

Abstract In this paper, we present a Delaunay refinement algorithm for 4-dimensional (3D+t) segmented images. The output mesh is proved to consist of sliver-free simplices. Assuming that the hyper-surface is a closed smooth manifold, we also guarantee faithful geometric and topological approximation. We implement and demonstrate the effectiveness of our method on publicly available segmented cardiac images. Finally, we devise a tightly-coupled parallelization technique to boost the performance of our 4-dimensional mesher, thereby taking advantage of the multi-core and many-core platforms already available in the market.

1 Introduction

Technological advances in imaging have made the acquisition of 4D medical images feasible [43, 55, 56, 58]. At the same time, pentatope capable FEM solvers [8, 45] operating directly on 4D data have been shown to be effective for advection-diffusion and Navier-Stokes formulations.

In this paper, we describe a 4-dimensional Delaunay mesh algorithm which operates directly on a 4-dimensional image $\mathscr{I}$. $\mathscr{I}$ represents the domain Ω to be meshed as the temporal evolution of a 3D object. That is, $\Omega = \bigcup_{t_i} \Omega_{t_i}$, where Ω_{t_i} is the 3D object at time t_i (i.e., the i^{th} slice of Ω).

Volume mesh generation methods can be divided into two categories: *PLC-based* and *Isosurface-based*. The PLC-based methods assume that the *surface* $\partial\Omega$ of the volume Ω (about to be meshed) is given as a *Piecewise Linear Complex* (PLC) which contains linear sub-faces embedded in 3 or 4 dimensions [15, 16, 19, 20, 37, 41, 42, 50, 53]. The limitation of this method is that the success of meshing depends on the quality of the given PLC: if

Panagiotis Foteinos
Computer Science Department, College of William and Mary, Virginia, USA
E-mail: panagiotis.foteinos@gmail.com

Nikos Chrisochoides
Computer Science Department, Old Dominion University, Virginia, USA
E-mail: nikos@cs.odu.edu

the PLC forms very small angles, then the overall mesh quality deteriorates and termination might be compromised [50, 52]. In Computed Aided Design (CAD) applications, the surface is usually given as a PLC. In biomedical Computer Aided Simulations (CAS), however, there is no reason to use this approach, since it might (depending on the geometry of the image, its segmentation, and its decimation [19]) add the additional small input angle limitation. A workaround for this small input angle limitation is to treat the surface voxels as the PLC of the domain, since those input facets meet at large angles (90 or 180 degrees). This, however, would introduce an unnecessary large number of elements and little control over the density of the domain.

The Isosurface-based methods assume that Ω is known through a function $f : \mathbb{R}^d \rightarrow \mathbb{R}$, such that points in different regions of interest evaluate f differently. This assumption covers a wide range of inputs used in modeling and simulation, such as parametric surfaces/volumes [46], level-sets and segmented multi-labeled images [12, 36, 47]. Of course, these type of functions can also represent PLCs [36], a fact that makes the Isosurface-based method a general approach. Isosurface-based methods ought to recover and mesh both the *(hyper-) isosurface* $\partial\Omega$ and the volume. This method does not suffer from any unnecessary small input angle artifacts introduced by the initial conversion to PLCs, since $\partial\Omega$ is recovered and meshed during refinement.

In this paper, we describe a space-time Delaunay Isosurface-based meshing technique for 4 dimensions. We show that the resulting mesh is sliver free consisting of pentatopes whose boundary is a correct approximation of the underlying hyper-isosurface $\partial\Omega = \bigcup_{t_i} \partial\Omega_{t_i}$. Note that space-time meshing is different from dynamic surface simulations (see [35] and the references therein for example). In those simulations, the isosurface is not known; instead, a tetrahedral mesh is adapted on each time step that describes accurately the free surface dynamics.

One way to solve the space-time 4D problem is to mesh separately each 3D object Ω_{t_i} and then connect the elements between two consecutive objects to obtain space-time elements. However, finding such correspondence—which also has to satisfy the quality criteria— is not intuitive, especially when the topology and the geometry of the two objects varies drastically. Alternatively, one could mesh a single object Ω_{t_i} and then deform the mesh to match the shape of the other temporal instances. The limitation of this approach is twofold. First, the quality of the deformed mesh might be much worse than the original; second, there is no control over the mesh density across both the spatial and the temporal direction [8], since the mesh size of the original instance determines the size of the rest of the instances.

Space-time meshing methods have already been proposed in the literature [29, 54]. They assume, however, that the evolving object Ω_{t_i} has the same spatial space across time. Furthermore, the implementation of these techniques is confined to only the 2D+t case (i.e., the space-time elements are tetrahedra). The more general 3D+t meshing has been the focus in [8, 45], but they consider only convex hyper-surfaces such as hyper-cubes or hyper-cylinders. To our knowledge, the method presented in this paper is the first to address the 3D+t problem where the topology and the geometry of the evolving object may differ substantially through time, and hence, it is allowed to form complex hyper-surfaces.

In the literature [3, 7, 10, 14, 16, 17], it is shown that given a sufficiently dense sample on a surface $\partial\Omega$, the restriction of its Delaunay triangulation to $\partial\Omega$ is a topologically good approximation, or, alternatively, it satisfies the closed-topological-ball property [28]. Their focus, however, was not on volume meshing, but rather, on surface reconstruction. In this

paper, we fill the space-time volume Ω with sliver-free pentatopes, such that $\partial\Omega$ is approximated correctly.

Computing the appropriate sample of the surface is a challenging task. In the literature, however, it is assumed that either such a sample is known [3–5] or that an initial sparse sample is given [11, 46, 49]. In this paper, we propose a method that starts directly from labelled images (of one or many more connected components) and computes the appropriate sample on the fly, respecting at the same time the quality and fidelity guarantees.

Our algorithm guarantees that the resulted pentatopes are of bounded aspect ratio. We achieve that by generating elements of low radius-edge ratio and by proving the absence of slivers. We clean the mesh from slivers by integrating into our framework the theory presented in [37]. In [37], the surface is given as an already meshed polyhedral domain (i.e., the method in [37] is a PLC-based method), a different problem than ours, since it is our algorithm's responsibility to mesh both the underlying zero-surfaces and the bounded volume with topological and geometric guarantees.

Lastly, we parallelize our sequential 4D mesher by employing a speculative execution tightly-coupled [31, 32, 44] parallelization technique which improves the performance of our code by a factor of 6.35 on 12 cores.

The rest of the paper is organized as follows: Section 2 introduces some basic terminology, and in Section 3 we present our algorithm. In Section 4 and Section 5, we prove the guarantees. Section 6 evaluates our method on segmented 4D cardiac data. Section 7 elaborates on the techniques we developed to improve the performance of our algorithm and Section 8 concludes the paper.

2 Preliminaries

The input of our algorithm is a segmented n dimensional image $\mathscr{I} \subset \mathbb{R}^n$. The object $\Omega \subseteq \mathscr{I}$ is assumed to be represented as a cut function $f : \mathbb{R}^n \mapsto \mathbb{R}$ such that its surface $\partial\Omega$ is defined by the set $\{f(p) = 0\}$ [36, 46]. Clearly, from a segmented image, the zero-surface $\{f(p) = 0\}$ can be easily computed by interpolating the voxel values.

We assume that given a point $p \in \mathbb{R}^4$, we can ask for p's closest point on $\partial\Omega$. This can be accomplished by an Euclidean Distance Transform (EDT) [25, 40]. Specifically, the EDT returns the voxel $p' \in \partial\Omega$ which is closest to p. Then, we traverse the ray pp', and we compute the intersection between the ray and $\partial\Omega$ by interpolating the positions of different signs [39]. Points on $\partial\Omega$ are referred to as *feature* points.

Definition 1 (local feature size) The *local feature size* $\mathrm{lfs}_{\partial\Omega}(x)$ of a point $x \in \partial\Omega$ is defined as the (closest) distance between x and the medial axis of $\partial\Omega$.

Remark 1 Since $\partial\Omega$ is smooth, the local feature size is bounded from below by a positive constant $\mathrm{lfs}_{\partial\Omega}$, that is,

$$\mathrm{lfs}_{\partial\Omega}(x) > \mathrm{lfs}_{\partial\Omega} > 0. \tag{1}$$

Another useful property is that the local feature size is 1-Lipschitz, that is,

$$\mathrm{lfs}_{\partial\Omega}(p) \leq |p-q| + \mathrm{lfs}_{\partial\Omega}(q). \tag{2}$$

Definition 2 (ε-sample [4]) A point set $V \subset \partial\Omega$ is called an ε-sample if for every point $p \in \partial\Omega$ there is a point $v \in V$ at a distance at most $\varepsilon \cdot \mathrm{lfs}_{\partial\Omega}(p)$ from p.

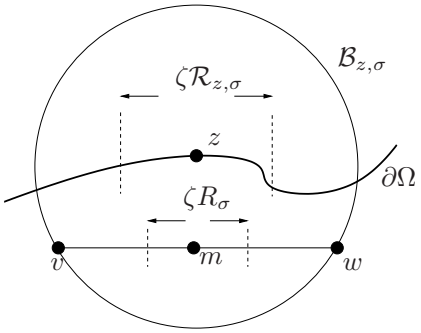


Fig. 1: A 2D illustration. The simplex $\sigma = \{v, w\}$ and its surface ball $\mathscr{B}_{z,\sigma}$. m is the midpoint of σ. Observe that since the radius $\mathscr{R}_{z,\sigma}$ of $\mathscr{B}_{z,\sigma}$ is larger than the radius $\mathscr{R}_\sigma = |m - v|$ of $\mathscr{B}_\sigma$, the picking region of σ as defined here is larger than the picking region of [37].

Let V be a finite set of vertices $V = \{v_1, \ldots, v_N\} \subset \mathbb{R}^n$. The *Delaunay triangulation* of V is denoted by $\mathscr{D}(V)$. A *k-simplex* $\sigma_k = \{v_1, \ldots, v_{k+1}\} \in \mathscr{D}(V)$ is a simplex defined by $k+1$ vertices. We denote the length of the shortest edge of a simplex σ with $l_{\min}(\sigma)$. The *circumball* $\mathscr{B}_\sigma$ of a simplex σ is the smallest closed ball circumscribing σ's vertices. $\mathscr{R}_\sigma$ is the circumradius length of the simplex and $c(\sigma)$ is its circumcenter. The radius-edge ratio of a simplex σ is defined as $\rho(\sigma) = \frac{\mathscr{R}_\sigma}{l_{\min}(\sigma)}$.

Following the definition of [37], the metric we use to characterize the quality of a simplex σ_k is $\tau_{\sigma_k} = \frac{\mathrm{Vol}_{\sigma_k}}{l_{\min}(\sigma_k)^k} \in \left[0, \frac{\sqrt{\frac{k+1}{2^k}}}{k!}\right]$ [48]. Low values of τ imply a poor-quality element.

Definition 3 (sliver [37]) Simplex σ is a sliver if it contains a k-simplex σ_k ($k \leq 4$) such that $\rho(\sigma_k) < \bar{\rho}$, $\tau_{\sigma_k} < \bar{\tau}$, and for any m-simplex σ_m of σ ($m < k$), $\rho(\sigma_m) < \bar{\rho}$, $\tau_{\sigma_m} \geq \bar{\tau}$.

The *Voronoi cell* $\mathrm{Vor}(v)$ of a vertex $v \in V$ is the set $\mathrm{Vor}(v) = \{p \in \mathbb{R}^n \mid |v - p| \leq |q - p|, \forall q \in V\}$. The *Voronoi dual* of a simplex $\sigma \in \mathscr{D}(V)$ is defined as the set $\mathrm{Vor}(\sigma) = \left\{\bigcap_i^{k+1} \mathrm{Vor}(v_i)\right\}$.

Definition 4 (restriction) The restriction of $\mathscr{D}(V)$ to space $\partial\Omega$ is denoted by $\mathscr{D}_{|\partial\Omega}(V)$ and is a *simplicial complex* (as is $\mathscr{D}(V)$) that contains simplices of $\mathscr{D}(V)$ whose Voronoi dual intersects $\partial\Omega$ in a non-empty set.

Definition 5 (surface ball [11]) Let σ be a k simplex and let $\mathrm{Vor}(\sigma)$ intersect $\partial\Omega$ at a point z. Any ball centered at z circumscribing σ is called a *surface ball*. The corresponding surface ball is denoted by $\mathscr{B}_{z,\sigma}$ and its radius by $\mathscr{R}_{z,\sigma}$ in the sequel.

By the definition of Voronoi diagrams, $\mathscr{B}_{z,\sigma}$ does not contain any vertex of V in its interior.

Definition 6 [picking region] The *picking region* $\mathscr{PR}(\sigma_4)$ of a 4-simplex σ_4 is defined as the 4-dimensional solid ball centered at $c(\sigma_4)$ with radius $\zeta\mathscr{R}_{\sigma_4}$, $\zeta < 1$. The picking region of a lower dimensional restricted simplex σ_k ($k < 4$) with surface ball $\mathscr{B}_{z,\sigma_k}$ is the intersection between $\partial\Omega$ and the 4-dimensional solid ball centered at z with radius $\zeta\mathscr{R}_{z,\sigma_k}$, $\zeta < 1$.

The exact value range of ζ will be determined in Theorem 1 of Section 4.

We note that $\mathscr{PR}(\sigma_4)$ and $\mathscr{PR}(\sigma_k)$ $(k < 4)$ are contained in $\mathscr{B}_\sigma$ and $\mathscr{B}_{z,\sigma}$, respectively. Observe that the picking region of σ_k $(k < 4)$ is a topological k-ball and does not belong (necessarily) in the affine k dimensional space defined by σ_k. This is different than the definition in [37], where the picking regions are defined inside the intersection of $\mathscr{B}_\sigma$ with the affine space of σ. The reason for this change is the fact that the input of our algorithm is not a *Piecewise Linear Complex* (PLC) but a cut function.

A good point $p \in \mathscr{PR}(\sigma)$ is a point that does not introduce smaller slivers. A sliver is small when its radius is less than $b\mathscr{R}_\sigma$. In [37], it is proved that (a) the number of small slivers $S(\sigma)$ possibly created after the insertion of p is constant, and (b) the volume $|F_\sigma|$ (the *forbidden region*) inside which p forms a small sliver is bounded from above. The same findings hold in our case, too, where the picking region of a restricted facet σ_3 is not inside the intersection of $\mathscr{B}_{\sigma_3}$ and σ_3's affine space, but inside the intersection of $\mathscr{B}_{z,\sigma_3}$ and $\partial\Omega$.

Lemma 1 *Given a mesh whose simplices have radius-edge ratios bounded from above by $\bar{\rho}$, a point p inside the picking region of a σ_k can be found in a constant number of random rounds, such that any new sliver created after the insertion of p has circumradius no smaller than $b\mathscr{R}_{\sigma_k}$ if $k = 4$, or no smaller than $b\mathscr{R}_{z,\sigma_k}$ if $k = 3$.*

Remark 2 The proof is similar to [37], since $|F_\sigma|$ and $S(\sigma)$ do not change and the volume of the intersection of $\mathscr{B}_{\sigma_3}$ and σ_3's affine space is smaller than the intersection of $\mathscr{B}_{z,\sigma_3}$ and $\partial\Omega$. See Figure 1 for an illustration.

As is the case of ζ (Definition 6), the exact value ranges of $\bar{\rho}$ and b will be determined in Theorem 1 of Section 4.

3 Algorithm

The user specifies a parameter δ. It will be clear in Section 5 that the lower δ is, the better the mesh boundary will approximate $\partial\Omega$. For brevity, the quantity $\delta \cdot \mathrm{lfs}_{\partial\Omega}(z)$ is denoted by $\Delta_{\partial\Omega}(z)$ where z is a feature point.

Our algorithm initially inserts the 16 corners of a hyper-box that contains the 4 dimensional object Ω such that the distance between a box corner x and its closest feature point $z = \mathrm{cfp}_{\partial\Omega}(x)$ is at least $2\Delta_{\partial\Omega}(z)$. After the computation of this initial triangulation, the refinement starts dictating which extra points are inserted. At any time, the Delaunay triangulation $\mathscr{D}(V)$ of the current vertices V is maintained. Note that by construction, $\mathscr{D}(V)$ always covers the entire hyper-volume and that any point on the box is separated from $\partial\Omega$ by a distance at least $2\Delta_{\partial\Omega}(z)$ where z is a feature point.

During the refinement, some vertices are inserted exactly on the box; these vertices are called *box vertices*. The box vertices might lie on 1, 2, or 3-dimensional box faces. We shall refer to the vertices that are neither box vertices nor feature vertices as *free vertices*.

The algorithm inserts new vertices for three reasons: to guarantee that (a) $\partial\Omega$ is correctly recovered, (b) all the elements have small radius-edge ratio, and (c) there are no slivers. Specifically, for a 4-simplex σ_4 in the mesh, the following rules are checked in this order:

- **R1**: Let $\mathscr{B}_{\sigma_4}$ intersect $\partial\Omega$ and z be equal to $\mathrm{cfp}_{\partial\Omega}(c(\sigma_4))$. If z is at a distance no closer than $\Delta_{\partial\Omega}(z)$ to any other feature vertex, then z is inserted.
- **R2**: Let $\mathscr{B}_{\sigma_4}$ intersect $\partial\Omega$ and z be equal to $\mathrm{cfp}_{\partial\Omega}(c(\sigma_4))$. If $\mathscr{R}_\sigma \geq 2\Delta_{\partial\Omega}(z)$, $c(\sigma_4)$ is inserted.

- **R3**: Let $c(\sigma_4)$ lie inside Ω. If $\rho(\sigma_4) \geq \bar{\rho}$, $c(\sigma_4)$ is inserted.
- **R4**: Let $c(\sigma_4)$ lie inside Ω. If σ_4 contains a sliver, a good point inside $\mathscr{P}\mathscr{R}(\sigma_4)$ is inserted.
- **R5**: Let σ_3 ($\sigma_3 \subset \sigma_4$) be a restricted facet. If the vertices of σ_3 are not feature vertices, then a good point z inside $\mathscr{P}\mathscr{R}(\sigma_3)$ is inserted. All the free vertices closer than $\Delta_{\partial\Omega}(z)$ to z are deleted.

For $i < j$, priority is given to Ri over Rj. Immediately preceding the insertion of a point because of Rj, there is no element that violates a rule Ri. Also, in R4, priority is given to the lower dimensional slivers that σ_4 might contain.

Whenever there is no simplex for which R1, R2, R3, R4 or R5 apply, the refinement process terminates. *The final mesh reported is the set of pentatopes whose circumcenters lie inside Ω.*

In summary, R1 and R5 are responsible for generating a sufficiently dense sample on $\partial\Omega$. R5 also makes sure that the vertices of the simplices restricted to $\partial\Omega$ lie on $\partial\Omega$ similarly to [46]. Lastly, R2, R3 and R4 deal with the quality guarantees. In Section 4, we will show that there are values for b, ζ, and $\bar{\rho}$ that do not compromise termination.

To prove termination, no vertices should be inserted outside the bounding box. Notice, however, that vertices inserted due to R2 may lie outside the bounding box. To deal with such cases, $c(\sigma_4)$ is rejected for insertion. Instead, its *projection* $c'(\sigma_4)$ on the box is inserted in the triangulation. That is, $c'(\sigma_4)$ is the closest to $c(\sigma_4)$ box point. In Section 4 and Section 5, we prove that the insertion of projected points do not compromise quality or fidelity. Note that these projections are different than the traditional *encroachment rules* described in [50, 51].

Recall that pentatopes with circumcenters on $\partial\Omega$ or outside Ω are not part of the final mesh, which is why rule R3 and R4 do not check them.

4 Termination and Quality

In this section, we specify the appropriate values for ζ, $\bar{\rho}$, and b so that the algorithm terminates. Specifically, we will show that, during refinement, the shortest edge introduced into the mesh cannot be arbitrarily small.

Suppose that σ violates a rule Ri. σ is called an Ri element. Ri dictates the insertion of a point p (and possibly the removal of free points). Point p is called an Ri point.

Definition 7 (insertion radius and parent [50, 51]) Let p be an Ri point inserted because a simplex σ violates Ri. The *insertion radius* $\mathscr{R}_p$ of p is defined as the length of the shortest edge incident to p created right after the end of Ri and the *parent* $Par(p)$ of p as the most recently inserted vertex incident to the shortest edge of σ.

Lemma 2 *Let p and q define the shortest edge of a simplex σ and q be inserted after p. Then $\mathscr{R}_q \leq l_{\min}(\sigma)$.*

Proof Assume that right after the insertion of q, p is the closest point to q. In this case, $\mathscr{R}_q = |p-q| = l_{\min}(\sigma)$. Otherwise, there has to be another closest vertex to q, which implies that $\mathscr{R}_q < |p-q| = l_{\min}(\sigma)$.

The following Lemmas bounds from below the shortest edge introduced into the mesh after the insertion of a vertex.

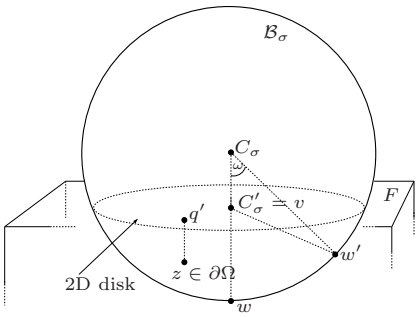


Fig. 2: Proof of Lemma 3, a 3D illustration.

Lemma 3 *Let v be a box vertex inserted into the mesh. Then, $\mathscr{R}_v \geq 2\Delta_{\partial\Omega}(z)$, where z is a feature point.*

Proof A box point v is inserted only because of R2. The circumcenter $c(\sigma)$ of a pentatope σ lies on or outside the box, and its projection $c'(\sigma) = v$ falls on the box. Without loss of generality, assume that the projection lies on the interior of a 3-face (i.e. a box tetrahedron) F. See Figure 2 for a 3D illustration. (If $c(\sigma)$ lies precisely on the box, $c'(\sigma)$ is equal to v.) Consider the (2D) disk (drawn) of $\mathscr{B}_\sigma$ which is coplanar with F. That disk contains v and separates $\mathscr{B}_\sigma$ in two sides: the side that contains $c(\sigma)$ and the side that contains a part of the box.

We claim that the closest vertex —say w— to v lies on the intersection of $\mathscr{B}_\sigma$'s boundary and the ray $\overrightarrow{c(\sigma)v}$. This can be explained by noting that $\mathscr{B}_\sigma$ is empty of vertices, and therefore, the closest to v that an arbitrary vertex w' already in the triangulation can be occurs when it lies on the boundary of $\mathscr{B}_\sigma$ and on the side of $\mathscr{B}_\sigma$ that contains a part of the box, as shown. Consider the triangle $w'vc(\sigma)$. From the law of cosines, we have that:

$$\begin{aligned}
|v-w'|^2 &= |c(\sigma)-w'|^2 + |c(\sigma)-v|^2 - 2\,|c(\sigma)-w'|\,|c(\sigma)-v|\cos\omega \\
&\geq |c(\sigma)-w'|^2 + |c(\sigma)-v|^2 - 2\,|c(\sigma)-w'|\,|c(\sigma)-v|\text{, since } \cos\omega \leq 1 \\
&= \left(|c(\sigma)-w'| - |c(\sigma)-v|\right)^2 \\
&= \left(\mathscr{R}_\sigma - |c(\sigma)-v|\right)^2\text{, since } w' \text{ lies on the sphere} \\
&= |v-w|^2,
\end{aligned}$$

and our claim is proved.

Consequently, any possible new edge connected to v has length at least $|v-w|$. Since σ triggers R2, $\mathscr{B}_\sigma$ has to intersect $\partial\Omega$, so there has to be a feature point $q \in \partial\Omega$ (illustrated) inside $\mathscr{B}_\sigma$ and on the same side of F as w. Let us denote with q' the projection of q to the box face F. By construction, $|q-q'|$ is at least $2\Delta_{\partial\Omega}(z)$ where z is a feature point; however, $|v-w|$ is always larger than $|q-q'|$, because $vw \parallel qq'$, and the statement holds. Similar reasoning applies in the case where $c'(\sigma)$ lies on a box triangle or a box edge.

The following Lemma proves a lower bound on the lengths created into the mesh because of R1 and R2:

Lemma 4 *Let p be a vertex inserted into the mesh because of R1 or R2. Then, $\mathscr{R}_p \geq \Delta_{\partial\Omega}(z)$, where z is the closest feature point to p.*

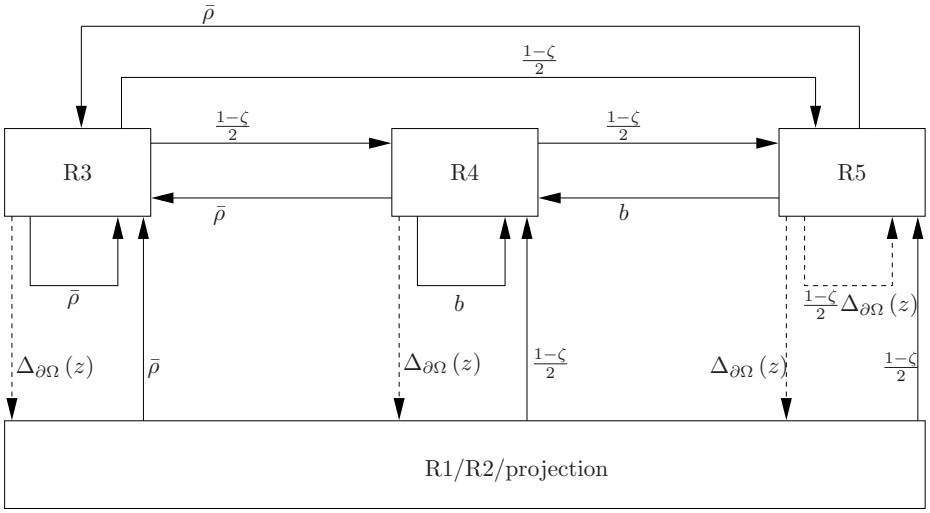


Fig. 3: Flow diagram depicting the relationship among the rules. No solid cycle should have a product less than 1. The dashed arrows break the cycle.

Proof If R1 is triggered, then p is equal to z. Since there is no other feature point already inserted in the mesh closer than $\Delta_{\partial\Omega}(p)$ to p, the statement holds. Otherwise, R2 applies for a simplex σ_4, and p is equal to $c(\sigma_4)$. Due to the empty ball property, $\mathscr{R}_p$ is at least $\mathscr{R}_{\sigma_4} \geq 2\Delta_{\partial\Omega}(\mathrm{cfp}_{\partial\Omega}(p))$, and the statement holds.

Lemma 5 *Let p be a vertex inserted into the mesh because R3 applies for an element σ. Then, $\mathscr{R}_p \geq \bar{\rho}\mathscr{R}_{Par(p)}$.*

Proof Since p is equal to $c(\sigma)$, $\mathscr{R}_p \geq \mathscr{R}_\sigma = \rho(\sigma) l_{\min}(\sigma) \geq \bar{\rho} l_{\min}(\sigma)$. Lemma 2 suggests that $l_{\min}(\sigma) \geq \mathscr{R}_{Par(p)}$, and the results follows.

Lemma 6 *Let p be inserted into the mesh because of R4. Then,*

- $\mathscr{R}_p \geq \frac{1-\zeta}{2}\mathscr{R}_{Par(p)}$*, if $Par(p)$ is neither R4 nor R5,*
- $\mathscr{R}_p \geq b\mathscr{R}_{Par(p)}$*, otherwise.*

Proof Let σ be the simplex that violates R4.

Suppose that $Par(p)$ is neither R4 nor R5. Since p belongs to the picking region of σ, $\mathscr{R}_p \geq (1-\zeta)\mathscr{R}_\sigma \geq \frac{1-\zeta}{2} l_{\min}(\sigma)$. From Lemma 2, we have that $\mathscr{R}_p \geq \frac{1-\zeta}{2}\mathscr{R}_{Par(p)}$.

Otherwise, consider the case $Par(p)$ is an R4 point. From Lemma 1, we know that the circumradius of σ is more than b times the circumradius of the R4 simplex σ' that inserted $Par(p)$. Therefore, $\mathscr{R}_p \geq (1-\zeta)\mathscr{R}_\sigma \geq (1-\zeta)b\mathscr{R}_{\sigma'}$. The quantity $(1-\zeta)\mathscr{R}_{\sigma'}$ is equal to $\mathscr{R}_{Par(p)}$, and the statement holds.

The exact same logic holds when $Par(p)$ is an R5 point, by just substituting $\mathscr{R}_{z,\sigma'}$ for $\mathscr{R}_{\sigma'}$ where σ' is an R5 simplex.

Lemma 7 *Let p be inserted into the mesh because of R5. Then,*

- $\mathscr{R}_p \geq \frac{1-\zeta}{2}\mathscr{R}_{Par(p)}$*, if* $Par(p)$ *is not an R5 point,*
- $\mathscr{R}_p \geq \frac{1-\zeta}{2}\Delta_{\partial\Omega}(Par(p))$*, otherwise.*

Proof Let σ_3 be the simplex that violates R5.

Suppose that $Par(p)$ is not an R5 point. Because of Lemma 2, the shortest edge of σ_3 is at least $\mathscr{R}_{Par(p)}$. Therefore, any surface ball of σ_3 has radius at least $\frac{1}{2}\mathscr{R}_{Par(p)}$. Since the surface ball does not contain any vertex in its interior, $\mathscr{R}_p \geq \frac{1-\zeta}{2}\mathscr{R}_{Par(p)}$.

Suppose that $Par(p)$ is an R5 point. Note that when $Par(p)$ is inserted, all the free vertices closer than $\Delta_{\partial\Omega}(Par(p))$ to $Par(p)$ are deleted. Due to R5, σ_3 contains at least one free vertex. Since $Par(p)$ is the most recently inserted vertex incident to the closest edge of σ_3, the edge that contains $Par(p)$ and the free vertex has to be at least $\Delta_{\partial\Omega}(Par(p))$. Therefore, any surface ball of σ_3 has radius at least $\frac{1}{2}\Delta_{\partial\Omega}(Par(p))$. Hence, $\mathscr{R}_p \geq \frac{1-\zeta}{2}\Delta_{\partial\Omega}(Par(p))$.

Putting all the Lemmas together, the solid arrows of Figure 3 show the insertion radius of the inserted point as a fraction of the insertion radius of its parent. An arrow from Ri to Rj with label x implies that the insertion radius of an Rj point p is at least x times larger than the insertion radius of its Ri parent $Par(p)$. The label x of the dashed arrows is the absolute value of $\mathscr{R}_p$. Note that the labels of the dashed arrows depend on the local feature size of $\partial\Omega$ and as such are always positive constants.

Recall that during refinement, free vertices might be deleted (because of R5). Nevertheless, such deletions of vertices are always preceded by insertion of feature points. Considering the fact that feature vertices are never deleted from the mesh, termination is guaranteed if we prove that the insertion radii of the inserted vertices cannot decrease indefinitely. Clearly [37, 50, 51], if there is no solid cycle of product less than 1, termination is guaranteed.

Theorem 1 *The algorithm terminates producing simplices of bounded aspect ratio, if*

- $\frac{(1-\zeta)^2}{4}\bar{\rho} \geq 1$*, and*
- $\frac{1-\zeta}{2}b \geq 1$.

Proof See Figure 3. The smallest product is produced by the solid cycles R3→R4→R5→R3 and R4→R5→R4. By requiring the label product of these loops to be more than 1, the desired result follows.

5 Accuracy

In this section, we prove that the mesh boundary is equal to the restriction of a $\partial\Omega$ sample Z to $\partial\Omega$. In the literature, it is proved that these tetrahedra approximate the surface correctly in geometric and topological sense [3, 10, 16].

First, we show that δ directly controls the density of the feature vertices. Let V be the set of vertices in the final mesh and Z be equal to $V \cap \partial\Omega$, i.e., Z is equal to the set of all the feature vertices.

Lemma 8 *Let* $\delta < \frac{1}{4}$*. Then* Z *is a* $\frac{5\delta}{1-4\delta}$*-sample of* $\partial\Omega$.

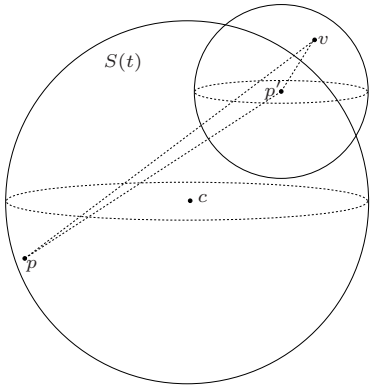


Fig. 4: Proof of Lemma 8.

Proof Recall that upon termination, there is no tetrahedron for which R1, R2, R3, R4, or R5 apply.

See Figure 4. Let p be an arbitrary point on $\partial\Omega$. Since $\mathscr{D}(V)$ covers all the domain, point p has to lie on or inside the circumsphere of a pentatope σ (not shown). Hence, $\mathscr{B}_\sigma$ intersects $\partial\Omega$. Let point p' be the feature point closest to $c(\sigma)$. Note that $|c(\sigma)-p| \geq |c(\sigma)-p'|$ and therefore p' lies on or inside σ's circumsphere. We also know that σ's circumradius has to be less than $2\Delta_{\partial\Omega}(p')$, since otherwise R2 would apply for t. Therefore, we have the following:

$$\begin{aligned}
|p-p'| &< 2\mathscr{R}_\sigma && \text{(because both } p \text{ and } p' \text{ lie on or inside } \mathscr{B}_\sigma) \\
&< 4\Delta_{\partial\Omega}(p') && \text{(because of R2)} \\
&\leq 4\delta\left(|p-p'| + \mathrm{lfs}_{\partial\Omega}(p)\right) && \text{(from Inequality (2)),}
\end{aligned}$$

and by reordering the terms, we obtain that:

$$\left|p-p'\right| < \frac{4\delta}{1-4\delta}\mathrm{lfs}_{\partial\Omega}(p), \text{ with } \delta < \frac{1}{4}. \tag{3}$$

Moreover, there must exist a feature vertex v in the triangulation closer than $\Delta_{\partial\Omega}(p') = \delta \cdot \mathrm{lfs}_{\partial\Omega}(p')$ to p' since otherwise R1 would apply for σ. Hence, $|v-p'| < \delta \cdot \mathrm{lfs}_{\partial\Omega}(p')$, and using Inequality (2), we have that:

$$\left|v-p'\right| < \delta\left(\left|p-p'\right| + \mathrm{lfs}_{\partial\Omega}(p)\right) \tag{4}$$

Applying the triangle inequality for $\triangle pvp'$ yields the following:

$$\begin{aligned}
|p-v| &\leq |-pp'| + |v-p'| \\
&< |p-p'| + \delta\left(|p-p'| + \mathrm{lfs}_{\partial\Omega}(p)\right) && \text{(from Inequality (4))} \\
&= |p-p'|(1+\delta) + \delta \cdot \mathrm{lfs}_{\partial\Omega}(p) \\
&< \tfrac{4\delta}{1-4\delta}\mathrm{lfs}_{\partial\Omega}(p)(1+\delta) + \delta \cdot \mathrm{lfs}_{\partial\Omega}(p) && \text{(from Inequality (3))} \\
&= \left(\tfrac{4\delta(1+\delta)}{1-4\delta} + \delta\right)\mathrm{lfs}_{\partial\Omega}(p) \\
&= \tfrac{5\delta}{1-4\delta}\mathrm{lfs}_{\partial\Omega}(p),
\end{aligned}$$

and the proof is complete.

Let us denote with ω_i one of the n connected components that Ω consists of: $\Omega = \bigcup_{i=1}^{n} \omega_i$.

The next two Lemmas prove a few useful properties for the mesh $\mathscr{M}$ and its boundary $\partial\mathscr{M}$. Our goal is to show that $\partial\mathscr{M}$ is always non-empty and does not have boundary (Lemma 10), a fact that will be used for proving the fidelity guarantees (Theorem 2).

Lemma 9 *Let $\delta \leq \frac{1}{4}$. Then, for every ω_i there is a pentatope $\sigma \in \mathscr{D}(V)$ such that $c(\sigma)$ lies inside ω_i.*

Proof Let us consider a single connected component ω_i. The same reasoning applies for any connected component of Ω.

For the sake of contradiction, assume that there is no pentatope whose circumcenter lies inside ω_i. Since the triangulation $\mathscr{D}(V)$ covers all the domain, the circumballs of the pentatopes in $\mathscr{D}(V)$ also cover the domain ω_i. Therefore, there has to be a circumball $\mathscr{B}_\sigma$ ($\sigma \in \mathscr{D}(V)$) which intersects a point m on the medial axis of $\partial\omega_i$ such that m lies inside ω_i. By our assumption, the circumcenter $c(\sigma)$ cannot lie inside ω_i. Therefore, $\mathscr{B}_\sigma$ intersects $\partial\omega_i$. Also, recall that R2 cannot apply to any pentatope. Hence, we have the following:

$$
\begin{aligned}
2 \cdot \delta \cdot \mathrm{lfs}_{\partial\Omega}\left(\mathrm{cfp}_{\partial\Omega}\left(c(\sigma)\right)\right) &> \mathscr{R}_\sigma && \text{(from R2)} \\
&\geq \tfrac{|\mathrm{cfp}_{\partial\Omega}(c(\sigma)) - m|}{2} && \text{(since } m \text{ and } \mathrm{cfp}_{\partial\Omega}(c(\sigma)) \text{ do not lie outside } \mathscr{B}_\sigma\text{)} \\
&\geq \tfrac{\mathrm{lfs}_{\partial\Omega}(\mathrm{cfp}_{\partial\Omega}(c(\sigma)))}{2} && \text{(since } m \text{ is on the medial axis)} \quad \Rightarrow \\
\delta &> \tfrac{1}{4},
\end{aligned}
$$

which raises a contradiction.

Lemma 10 *Let $\delta \leq \frac{1}{4}$. Then, $\partial\mathscr{M}$ is a non-empty set and does not have boundary.*

Proof The fact that $\partial\mathscr{M}$ is a non-empty set follows directly from Lemma 9: since $\mathscr{M}$ cannot be empty, its boundary $\partial\mathscr{M}$ cannot be empty too. For the other part, since $\partial\mathscr{M}$ is the boundary of a set of pentatopes, it cannot have a boundary.

The following Theorem proves the fidelity guarantees:

Theorem 2 *The mesh boundary $\partial\mathscr{M}$ is the restriction to $\partial\Omega$ of $Z = V \cap \partial\Omega$.*

Proof Let f be a tetrahedron σ_3 in $\partial\mathscr{M}$. As such, $\mathrm{Vor}(\sigma_3)$ intersects $\partial\Omega$. Due to R5, the vertices of σ_3 lie on $\partial\Omega$. Recall that the surface ball $\mathscr{B}_{z,\sigma_3}$ does not contain vertices in its interior. Therefore, $\mathscr{B}_{z,\sigma_3}$ is empty of vertices in $V \cap \partial\Omega$ also. Without loss of generality, assume that the vertices in V are in general position. Since there is a ball that circumscribes σ_3 and does not contain vertices of $V \cap \partial\Omega$ in its interior, σ_3 has to appear as a simplex in $\mathscr{D}(V \cap \partial\Omega)$. Since the center z of the surface ball lies on $\partial\Omega$, then the Voronoi dual of σ_3 intersects $\partial\Omega$ in $\mathscr{D}_{|\partial\Omega}(\partial\Omega)\,\partial\Omega \cap V$, as well. Hence, $\partial\mathscr{M} \subseteq \mathscr{D}_{|\partial\Omega}(\partial\Omega)\,\partial\Omega \cap V$.

For the other direction, we will prove that $\partial\mathscr{M}$ cannot be a proper subset of $\mathscr{D}_{|\partial\Omega}(\partial\Omega)\,\partial\Omega \cap V$, so equality between these 2 sets is forced. We will prove that any proper non-empty subset of $\mathscr{D}_{|\partial\Omega}(\partial\Omega)\,\partial\Omega \cap V$ has boundary; this is sufficient, because we have proved in Lemma 10 that $\partial\mathscr{M}$ is non-empty and does not have boundary.

$\mathscr{D}_{|\partial\Omega}(\partial\Omega)\,\partial\Omega \cap V$ is the restriction of a sample of a closed manifold $\partial\Omega$, and therefore it is a 3-manifold without boundary [3], meaning that any 2-simplex in $\mathscr{D}_{|\partial\Omega}(\partial\Omega)\,\partial\Omega \cap V$

Table 1: Information about the images of the five patients used in this section. The spacing for all the images is $(1.77, 1.77, 6, 1)$.

Case	Pat1	Pat2	Pat3	Pat4	Pat5
#Voxels	$(100 \times 100 \times 44 \times 15)$	$(100 \times 100 \times 34 \times 15)$	$(100 \times 100 \times 26 \times 15)$	$(100 \times 100 \times 31 \times 15)$	$(100 \times 100 \times 29 \times 15)$

Table 2: Statistics of the output meshes generated for each patient.

	Pat1	Pat2	Pat3	Pat4	Pat5
#Pentatopes	49,479	43,673	8,883	63,016	56,528
#Boundary Tetrahedra	30,758	29,089	8,271	36,281	33,308
#Vertices	4,709	4,314	1,362	5,567	5,132
Shortest edge (*mm*)	3.45	3.87	3.90	3.5	4.63
Radius-edge ratio (maximum, average, deviation)	(1.93, 1.02, 0.17)	(1.78, 0.98, 0.15)	(1.54, 0.92, 0.10)	(2.20, 1.06, 0.18)	(1.87, 1.05, 0.18)
Normalized volume (minimum, average, deviation)	(0.01, 0.34, 0.18)	(0.01, 0.38, 0.18)	(0.02, 0.43, 0.17)	(0.01, 0.32, 0.17)	(0.01, 0.33, 0.17)

is incident to exactly two 3-simplices of $\mathscr{D}_{|\partial\Omega}(\partial\Omega)\,\partial\Omega \cap V$. Since any proper non-empty subset $\mathscr{A}$ of $\mathscr{D}_{|\partial\Omega}(\partial\Omega)\,\partial\Omega \cap V$ has fewer 3-simplices, $\mathscr{A}$ contains at least a 2-simplex σ_2 incident to only one 3-simplex. But this implies that σ_2 belongs to the boundary of $\mathscr{A}$, and the proof is complete.

6 Experimental Evaluation

The algorithm is implemented in C++. We employed the Bowyer-Watson kernel [13, 57] for point insertions. The removal of a point p is implemented by computing the *small* Delaunay triangulation of the vertices incident to p [26], such that the vertices inserted earlier in the triangulation are inserted into the small triangulation first. It can be shown [30] that these new created pentatopes can always be connected back to the original triangulation without introducing invalid elements. For the Euclidean Distance Transform, we made use of the related filter implemented in ITK [2] and described in [40]. Lastly, we borrowed CGAL's [1] exact predicates for the accurate computation of the 4D in-sphere tests.

We ran our code on five (segmented) images obtained from the 4D Heart Database [43]. The first three represent the moving left ventricle of the patients, while the last two represent the ventricle together with the myocardium for 15 cardiac cycles (see Table 1).

Recall that our algorithm needs the distance of any point on $\partial\Omega$ from the medial axis. The robust computation of the medial axis is a very difficult problem (see [27, 33] for computing the exact medial axis, [23] for a review of image-based medial axis methods, and [5] for computing the medial axis given a set of surface points) and out of the scope of this work. In the implementation, we assume that $\mathrm{lfs}_{\partial\Omega}(p)$ is uniform and equal to the unit, which implies that $\Delta_{\partial\Omega}(p)$ becomes equal to δ. That is, in practice, δ determines a uniform and (if small enough) dense sample of the surface. We experimentally verified that a δ value equal to 5 (the length of five consecutive voxels along the fourth dimension) yielded manifold mesh boundaries with vertices lying precisely on the iso-surface in accordance with Theorem 2.

The quantity τ_σ determines the aspect ratio of pentatope σ [37], but it is not normalized, and is therefore difficult to draw comparative conclusions. Consequently, for a pentatope σ of the final mesh, we report its *normalized volume* $\hat{\tau}_\sigma$ defined as the ratio of its volume over the volume of a regular pentatope with circumradius equal to the circumradius of σ

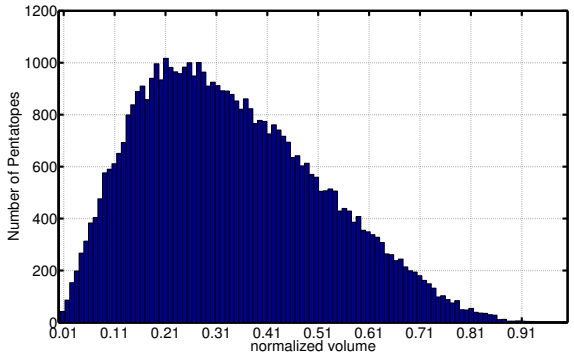


Fig. 5: Normalized volume histogram of the output mesh obtained for the input image Pat1.

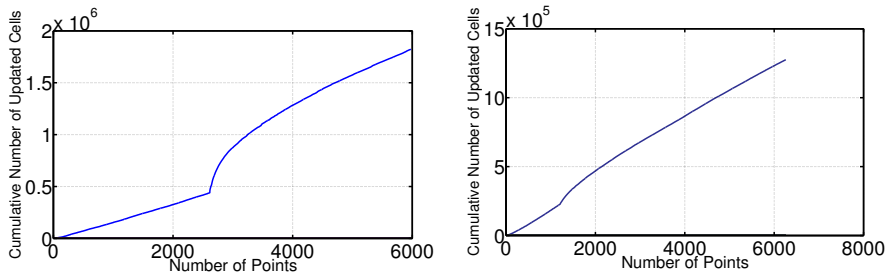
(or alternatively $\hat{\tau}_\sigma = \frac{384\text{Vol}_\sigma}{24\mathscr{R}_\sigma^4\sqrt{5}}$). Clearly, $\hat{\tau}_\sigma \in [0,1]$, where a value of 0 implies a degenerate pentatope, while 1 implies a perfect quality.

Table 2 shows quantitative data for the mesh generated on each image. We set the radius of the picking regions equal to $\zeta = \frac{1}{2}$. Theorem 1 suggests that $\bar{\rho}$ be at least 16 and b at least 4. We experimentally observed that by selecting 4 to 10 random points within the picking regions (both the 4- and the 3-topological balls), no small element σ was created with $\hat{\tau}_\sigma$ less than 0.01. Despite the fact a value of 0.01 is rather small, it is three orders of magnitude larger than the minimum normalized volume reported in the case where no picking regions are employed at all. Also, we notice that the average normalized volume is much higher than the minimum value. This fact together with the observed small standard deviation implies that most pentatopes have normalized volume away from the minimum value and very close to the average. Figure 5 shows the histogram of the normalized volumes for the first experiment of Table 2 (i.e, when the input image Pat1 was used). Similar histograms are observed for all the other inputs as well.

7 Improving the 4D Meshing Performance

During the development of the 4D Delaunay refinement code, we realized that its performance behaves very differently than the performance of the optimized 3D code we developed and described in the past [31, 32]. This is due to mainly two reasons: (a) the storage requirements and computations involved in a point insertion or removal are higher because of the increased dimensionality, and (b) the 4D CGAL predicates we employed to enforce robustness are not optimized as well as their 3D counterparts. Indeed, the achieved rate of meshing a 4D hyper-sphere with 40,000 elements is 145 pentatopes per second, while the achieved rate of meshing the hypersphere's equator with the same number of elements is 107,037 tetrahedra per second.

In this Section, we improve the speed of our 4D code by optimizing its complexity and by parallelizing the whole process. Since point removals account for approximately less than 1% of the total number of operations in all the cases we investigated, we focus on 4D Delaunay point insertions.



(a) (b)

Fig. 6: The complexity of the 4D code (a) before, and (b) after the Rule reordering.

7.1 Complexity

Ignoring the time involved for locating the first element in a point's cavity, the optimal complexity of a Delaunay insertion is constant. Therefore, inserting n points costs $\Omega(n)$ time. Although the 3D code reaches the optimal complexity in all the case studies used in our experiment, its 4D counterpart behaves very differently. Indeed, Figure 6a shows the number of deleted and created elements involved so far with respect to the number of inserted points on the 4D hyper-sphere. If the complexity was optimal, then the curve should look like a straight line; however, the complexity is far from ideal after the insertion of approximately 2,500 points. We obtained similar results when we ran our code on other inputs, such as hyper-torus and the five 4D hearts of Section 6.

Nevertheless, it can be proved [41] that it is possible to reach the optimal complexity if, at any given moment of refinement, the radius-edge ratio is bounded from above. In fact, this technique has already been applied successfully in the literature [34]. Therefore, we reordered the Rules of our algorithm (see Section 3) such that rule R3 has the highest priority among all the rest of the Rules. In this way, the mesh is of bounded radius-edge ratio at any given time, and as such, the expected complexity should be close to the optimal. Indeed, Figure 6b shows that the complexity curve behaves linearly. This improvement boosted the performance of the 4D code by 27% on the hyper-sphere mentioned above, bringing the rate of 145 pentatopes per second up to 184 for the 40,000 element mesh generated.

7.2 Parallelization

In this Subsection, we parallelize the 4D algorithm to take advantage of the multi-core and many-core platforms already available in the market. To our knowledge, this is the first attempt to parallelize the mesh generation and refinement of 4D space-time domains.

We employed a tightly-coupled approach similar to the concept presented in [31, 32]. Specifically, each thread T_i maintains its own list of poor elements, and it attempts to improve them by inserting or removing the appropriate vertices according to the Rules described in Section 3. To protect the cavities from concurrent accesses and guarantee valid meshes, T_i has to lock the vertices associated to that cavity. If T_i attempts to lock a vertex already acquired by another thread, it rolls back: the changes are discarded and T_i moves on refining the next element. However, as shown in our past work [31], this optimistic speculative approach might cause livelocks compromising in this way termination. We resolve this

issue by implementing special Contention Managers. The goal of the Contention Managers is to guarantee that at least one thread T_i makes progress, i.e., T_i is not stuck in consecutive rollbacks for an undefined period of time. Additionally, the 4D parallel tightly-coupled approach has to address load imbalances. Indeed, a thread T_i might run out of elements to refine. We chose the traditional Work Stealing [9] as our base load balance technique, since it was proven to behave very well in our optimistic setting [31]. Briefly, when T_i has no elements to refine, it asks poor elements from another thread. See [32] for detailed explanations about the Contention Managers and load balance techniques.

We deactivated the picking region technique described in detail in the previous Sections because we wanted to perform a 1-to-1 comparison with the 3D code to investigate which parallelization techniques applied successfully in 3D benefit the parallelization of the 4D problem as well. Keeping the picking regions would imply more than one round per insertion causing a considerable increase in the number of rollbacks, a fact not relevant to the nature of the 4D problem but to the technique of eliminating slivers.

Table 3a illustrates the weak scaling performance of the 4D parallel implementation on the Pat5 input 4D heart. The speedup is computed with respect to the rate of generated pentatopes (*number of elements per second*). The table also shows the average total overhead seconds per thread (last row) and the exact source of the overhead. As described in [31, 32], *contention seconds* is the time wasted by a thread invoking the Contention Manager, *balance seconds* is the time spent by a thread waiting for extra work to arrive, and *rollback seconds* is the time elapsed for the partial expansion of a cavity.

Although the same parallelization techniques scaled the 3D counterpart for a core count higher than 128, we observe that intensive overhead hampers scalability even on 12 cores in 4D domains. For example, more than 58% of the total execution time on 12 cores consisted of waiting on contention lists, balance lists, and rollbacks. Interestingly, the overhead of the 3D counterpart on a slice of the same 4D input was only 40% on 12 cores, when it generated a mesh of approximately the same size. This different behavior could be attributed to the fact that now the size of the cavity is much larger in 4D than it is in 3D. Indeed, we computed that the average size of the 4D cavity (4D Pat5 heart) is about 72.9 pentatopes, while the average size of the 3D cavity (slice of Pat5) is 18.0.

Nevertheless, the fact that most of the time is spent idling on contention and balance lists gives us the opportunity to perform cavity expansions in parallel. When a thread is working on inserting a point, it invites idling threads to perform the operation in parallel. This parallelization scheme is called *fine grained* parallelization and was successfully employed in the past by our group [6].

Table 3b shows the fine grained performance of our implementation. We observe that the overhead seconds were greatly reduced. For example, on 12 cores, the overhead seconds were reduced by 2.9X simply because threads help active threads to do useful work, and thus they wait on the contention/balance lists much less. As an immediate result, the number of elements per second (i.e., rate) of the fine grained implementation is 1.3X and 1.7X faster on 6 and 12 cores respectively when compared to the non fine grained version.

8 Conclusions

In this paper, we presented a space-time meshing method for (3D+t) image data. The method is able to provably clean up slivers and recover the hyper-surfaces faithfully. Experiments on five 4D cardiac images show that the resulting meshes consist of elements of bounded aspect ratio.

Table 3: The weak scaling performance of the parallel 4D method (a) without, and (b) with fine grained parallelism.

(a)

Threads	1	6	12
#Elements	301,336	1,559,480	2,870,670
Time (secs)	1687.80	2681.41	4283.94
Elements per second	178.54	581.59	670.10
Speedup	1.00	3.26	3.75
Contention seconds per thread	0.00	992.12	2,399.01
Balance seconds per thread	0.00	2.68	42.84
Rollback seconds per thread	0.00	53.63	85.68
Total overhead seconds per thread	0.00	1,048.43	2,527.52

(b)

Threads	1	6	12
#Elements	301,336	1,558,020	2,837,830
Time (secs)	1679.33	2039.95	2489.33
Elements per second	179.44	763.75	1,140.00
Speedup	1.00	4.26	6.35
Contention seconds per thread	0.00	346.79	771.69
Balance seconds per thread	0.00	2.04	24.89
Rollback seconds per thread	0.00	81.60	74.68
Total overhead seconds per thread	0.00	430.43	871.27

Efficient Discontinuous Galerkin formulations require that not only the hyper-surface $\partial\Omega = \bigcup_{t_i} \partial\Omega_{t_i}$ should be recovered but also the evolving 3D object Ω_{t_i} at certain time steps t_i [22]. This is a more challenging task considering the non-manifold nature of the underlying space-time domain at the intersection of $\partial\Omega$ and Ω_{t_i} and is left as future work.

Because of the increased memory space needed for high dimensional meshing, our 4D algorithm is rather slow compared to the optimized three dimensional Delaunay mesher described in our past work [31, 32]. Nevertheless, the tightly-coupled fine grained parallelization of the 4D code did yield a 6.35 speedup on 12 cores. We argue that the main bottleneck for its scalability is the excessive amount of contention, a fact that we did not observe in the 3D counterpart. We attribute this difference in behavior between the 3D and 4D implementation to the fact that the cavity size increases in higher dimensions, and therefore, tightly-coupled techniques need to lock many more vertices. In the future, we plan

to investigate other parallelization techniques, such as data decomposition [21] and domain decomposition [18, 38] since they are expected to alleviate the increased synchronization overhead observed in high dimensional meshing. In the future, we also plan to theoretically characterize the complexity of our parallel methods described in Section 7.2, determining their scalability on machines of different architecture [24].

Acknowledgements

The authors would like to thank Dr. Marek Behr, RWTH Aachen University, for the constructive discussions, and the anonymous reviewers for their comments and insight that helped the presentation of this paper. This work is supported in part by NSF grants: CCF-1139864, CCF-1136538, CSI-1136536 and CCF-1439079 and by the John Simon Guggenheim Foundation and the Richard T. Cheng Endowment.

References

1. CGAL, Computational Geometry Algorithms Library. `http://www.cgal.org`, v4.0.
2. ITK, Insight Segmentation and Registration Toolkit. `http://www.itk.org`, v4.1.0.
3. Nina Amenta and Marshall Bern. Surface reconstruction by Voronoi filtering. In *SCG '98: Proceedings of the fourteenth annual symposium on Computational geometry*, pages 39–48, New York, NY, USA, 1998. ACM.
4. Nina Amenta, Sunghee Choi, Tamal K. Dey, and N. Leekha. A Simple Algorithm for Homeomorphic Surface Reconstruction. *International Journal of Computational Geometry and Applications*, 12(1-2):125–141, 2002.
5. Nina Amenta, Sunghee Choi, and Ravi Krishna Kolluri. The power crust. In *Proceedings of the sixth ACM symposium on Solid modeling and applications*, SMA '01, pages 249–266, New York, NY, USA, 2001. ACM.
6. Christos Antonopoulos, Filip Blagojevic, Andrey Chernikov, Nikos Chrisochoides, and Dimitris Nikolopoulos. A multigrain Delaunay mesh generation method for multicore smt-based architectures. *Journal on Parallel and Distributed Computing*, 69:589–600, 2009.
7. Dominique Attali, Herbert Edelsbrunner, and Yuriy Mileyko. Weak witnesses for delaunay triangulations of submanifolds. In *Proceedings of the 2007 ACM symposium on Solid and physical modeling*, SPM '07, pages 143–150, New York, NY, USA, 2007. ACM.
8. Marek Behr. Simplex space-time meshes in finite element simulations. *International Journal for Numerical Methods in Fluids*, 57:1421–1434, 2008.
9. Robert D. Blumofe, Christopher F. Joerg, Bradley C. Kuszmaul, Charles E. Leiserson, Keith H. Randall, and Yuli Zhou. Cilk: an efficient multithreaded runtime system. In *Proceedings of the fifth ACM SIGPLAN symposium on Principles and practice of parallel programming*, PPoPP '95, pages 207–216, New York, NY, USA, 1995. ACM.
10. Jean-Daniel Boissonnat, Leonidas J. Guibas, and Steve Y. Oudot. Manifold reconstruction in arbitrary dimensions using witness complexes. *Discrete Comput. Geom.*, 42:37–70, May 2009.
11. Jean-Daniel Boissonnat and Steve Oudot. Provably good sampling and meshing of surfaces. *Graphical Models*, 67(5):405–451, 2005.
12. Dobrina Boltcheva, Mariette Yvinec, and Jean-Daniel Boissonnat. Mesh Generation from 3D Multi-material Images. In *Medical Image Computing and Computer-Assisted Intervention*, pages 283–290. Springer, September 2009.
13. Adrian Bowyer. Computing Dirichlet tesselations. *Computer Journal*, 24:162–166, 1981.
14. Frëdëric Cazals and Joachim Giesen. Delaunay triangulation based surface reconstruction: Ideas and algorithms. In *Effective Computational Geometry for Curves and surfaces*, pages 231–273. Springer, 2006.
15. Siu-Wing Cheng, Tamal K. Dey, Herbert Edelsbrunner, Michael A. Facello, and Shang-Hua Teng. Sliver exudation. *Journal of the ACM*, 47(5):883–904, 2000.
16. Siu-Wing Cheng, Tamal K. Dey, and Edgar A. Ramos. Manifold reconstruction from point samples. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '05, pages 1018–1027, Philadelphia, PA, USA, 2005. Society for Industrial and Applied Mathematics.

17. Siu-Wing Cheng, Tamal K. Dey, and Edgar A. Ramos. Delaunay refinement for piecewise smooth complexes. In *Proc. 18th Annu. ACM-SIAM Sympos. Discrete Algorithms*, pages 1096–1105. ACM Press, 2007.
18. Andrey Chernikov and Nikos Chrisochoides. Algorithm 872: Parallel 2D constrained Delaunay mesh generation. *ACM Transactions on Mathematical Software*, 34:6–25, January 2008.
19. Andrey Chernikov and Nikos Chrisochoides. Multitissue tetrahedral image-to-mesh conversion with guaranteed quality and fidelity. *SIAM Journal on Scientific Computing*, 33:3491–3508, 2011.
20. Andrey Chernikov and Nikos Chrisochoides. Generalized insertion region guides for Delaunay mesh refinement. *SIAM Journal on Scientific Computing*, 34(3):A1333–A1350, 2012.
21. Andrey N. Chernikov and Nikos P. Chrisochoides. Three-dimensional Delaunay refinement for multi-core processors. In *Proceedings of the 22nd annual international Conference on Supercomputing*, ICS '08, pages 214–224, New York, NY, USA, 2008. ACM.
22. Bernardo Cockburn, George E. Karniadakis, and Chi-Wang Shu. Discontinuous galerkin methods: theory, computation and applications. *Lecture notes in Computational Science and Engineering*, 11, 2000.
23. David Coeurjolly and Annick Montanvert. Optimal separable algorithms to compute the reverse euclidean distance transformation and discrete medial axis in arbitrary dimension. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29:437–448, March 2007.
24. David Culler, Richard Karp, David Patterson, Abhijit Sahay, Klaus Erik Schauser, Eunice Santos, Ramesh Subramonian, and Thorsten von Eicken. Logp: Towards a realistic model of parallel computation. *SIGPLAN Not.*, 28(7):1–12, July 1993.
25. Per Eric Danielsson. Euclidean Distance Mapping. *Computer Graphics and Image Processing*, 14:227–248, 1980.
26. Olivier Devillers and Monique Teillaud. Perturbations and vertex removal in a 3D Delaunay triangulation. In *Proceedings of the* 14^{th} *ACM-SIAM Symposium on Discrete algorithms*, SODA '03, pages 313–319. SIAM, 2003.
27. Tamal K. Dey and Wulue Zhao. Approximate medial axis as a voronoi subcomplex. *Computer-Aided Design*, 36(2):195–202, 2004.
28. Herbert Edelsbrunner and Nimish R. Shah. Triangulating topological spaces. In *SCG '94: Proceedings of the tenth annual symposium on Computational geometry*, pages 285–292, New York, NY, USA, 1994. ACM.
29. Jeff Erickson, Damrong Guoy, John M. Sullivan, and Alper Üngör. Building spacetime meshes over arbitrary spatial domains. *Eng. with Comput.*, 20(4):342–353, August 2005.
30. Panagiotis Foteinos and Nikos Chrisochoides. Dynamic parallel 3D Delaunay triangulation. In *International Meshing Roundtable*, pages 3–20, Paris, France, October 2012. Springer Berlin Heidelberg.
31. Panagiotis Foteinos and Nikos Chrisochoides. High quality real-time image-to-mesh conversion for finite element simulations. In *Proceedings of the 27th International ACM Conference on International Conference on Supercomputing*, ICS '13, pages 233–242, New York, NY, USA, 2013. ACM.
32. Panagiotis A. Foteinos and Nikos P. Chrisochoides. High quality real-time image-to-mesh conversion for finite element simulations. *Journal of Parallel and Distributed Computing*, 74(2):2123–2140, 2014.
33. Peter Giblin and Benjamin B. Kimia. A formal classification of 3D medial axis points and their local geometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:238–251, January 2004.
34. Benoît Hudson, Gary Miller, and Todd Phillips. Sparse voronoi refinement. In *Proceedings of the 15th International Meshing Roundtable*, pages 339–356. Springer Berlin Heidelberg, 2006.
35. Xiangmin Jiao, Andrew Colombi, Xinlai Ni, and John Hart. Anisotropic mesh adaptation for evolving triangulated surfaces. *Eng. with Comput.*, 26(4):363–376, 2010.
36. François Labelle and Jonathan Richard Shewchuk. Isosurface stuffing: fast tetrahedral meshes with good dihedral angles. *ACM Transactions on Graphics*, 26(3):57.1–57.10, 2007.
37. Xiang-Yang Li. Generating Well-Shaped D-dimensional Delaunay Meshes. In Jie Wang, editor, *Computing and Combinatorics*, volume 2108 of *Lecture Notes in Computer Science*, pages 91–100. Springer Berlin / Heidelberg, 2001.
38. Leonidas Linardakis and Nikos Chrisochoides. Graded Delaunay decoupling method for parallel guaranteed quality planar mesh generation. *SIAM Journal on Scientific Computing*, 30(4):1875–1891, March 2008.
39. William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *SIGGRAPH Computer Graphics*, 21(4):163–169, 1987.
40. Calvin .R. Maurer, Qi Rensheng, and Vijay Raghavan. A linear time algorithm for computing exact euclidean distance transforms of binary images in arbitrary dimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(2):265 – 270, feb 2003.
41. Gary L. Miller, Dafna Talmor, Shang-Hua Teng, and Noel Walkington. A Delaunay based numerical method for three dimensions: generation, formulation, and partition. In *Proceedings of the 27th Annu. ACM Sympos. Theory Comput*, pages 683–692. ACM, 1995.

42. Scott A. Mitchell and Stephen A. Vavasis. Quality mesh generation in higher dimensions. *SIAM J. Comput.*, 29(4):1334–1370, February 2000.
43. L. Najman, J. Cousty, M. Couprie, H. Talbot, S. Clment-Guinaudeau, T. Goissen, and J. Garot. An open, clinically-validated database of 3D+t cine-mr images of the left ventricle with associated manual and automated segmentation. `http://www.laurentnajman.org/heart/index.html`.
44. Demian Nave, Nikos Chrisochoides, and Paul Chew. Parallel Delaunay refinement for restricted polyhedral domains. *Computational Geometry: Theory and Applications*, 28:191–215, 2004.
45. Martin Neumüller and Olaf Steinbach. Refinement of flexible spacetime finite element meshes and discontinuous Galerkin methods. *Computing and Visualization in Science*, 14:189–205, 2011.
46. Steve Oudot, Laurent Rineau, and Mariette Yvinec. Meshing volumes bounded by smooth surfaces. In *Proceedings of the International Meshing Roundtable*, pages 203–219. Springer-Verlag, September 2005.
47. Jean-Philippe Pons, Florent Ségonne, Jean-Daniel Boissonnat, Laurent Rineau, Mariette Yvinec, and Renaud Keriven. High-Quality Consistent Meshing of Multi-label Datasets. In *Information Processing in Medical Imaging*, pages 198–210. Springer Berlin Heidelberg, 2007.
48. Stanley Rabinowitz. The volume of an n-simplex with many equal edges. *Missouri Journal of Mathematical Sciences*, pages 11–17, 1989.
49. Laurent Rineau and Mariette Yvinec. Meshing 3D domains bounded by piecewise smooth surfaces. In *Proceedings of the International Meshing Roundtable*, pages 443–460, 2007.
50. Jonathan Richard Shewchuk. Tetrahedral mesh generation by Delaunay refinement. In *Proceedings of the* 14^{th} *ACM Symposium on Computational Geometry*, pages 86–95, Minneapolis, MN, 1998. ACM.
51. Jonathan Richard Shewchuk. Delaunay refinement algorithms for triangular mesh generation. *Computational Geometry: Theory and Applications*, 22(1-3):21–74, May 2002.
52. Hang Si. Constrained Delaunay tetrahedral mesh generation and refinement. *Finite Elements in Analysis and Design*, 46:33–46, 2010.
53. Hang Si. TetGen, A Quality Tetrahedral Mesh Generator and a 3D Delaunay Triangulator. `http://tetgen.berlios.de/`, v1.4.3.
54. Shripad Thite. Efficient spacetime meshing with nonlocal cone constraints. In *13th International Meshing Roundtable*, pages 47–58, 2004.
55. Yanghai Tsin, Klaus Kirchberg, Guenter Lauritsch, and Chenyang Xu. A deformation tracking approach to 4d coronary artery tree reconstruction. In Guang-Zhong Yang, David Hawkes, Daniel Rueckert, Alison Noble, and Chris Taylor, editors, *Medical Image Computing and Computer-Assisted Intervention MICCAI 2009*, volume 5762 of *Lecture Notes in Computer Science*, pages 68–75. Springer Berlin / Heidelberg, 2009.
56. M von Siebenthal, G Székely, U Gamper, P Boesiger, A Lomax, and Ph Cattin. 4D MR imaging of respiratory organ motion and its variability. *Physics in Medicine and Biology*, 52(6):1547–1564, 2007.
57. David F. Watson. Computing the n-dimensional Delaunay tesselation with application to Voronoi polytopes. *Computer Journal*, 24:167–172, 1981.
58. Ernst Weigang, Fabian A. Kari, Friedhelm Beyersdorf, Maximilian Luehr, Christian D. Etz, Alex Frydrychowicz, Andreas Harloff, and Michael Markl. Flow-sensitive four-dimensional magnetic resonance imaging: flow patterns in ascending aortic aneurysms. *European Journal of Cardio-Thoracic Surgery*, 34(1):11–16, 2008.