Polytopal Complex Construction and Use in Persistent Homology

Rohit P. Singh and Philip A. Wilsey
University of Cincinnati, Cincinnati, OH 45221, USA
Email: singh2ro@mail.uc.edu, wilseypa@gmail.comn

***Abstract*—Topological Data Analysis (TDA) is a data mining technique to characterize the topological features of data. *Persistent Homology* (PH) is an important tool of TDA that has been applied to a wide range of applications. However its time and space complexities motivates a need for new methods to compute the PH of high-dimensional data. An important, and memory intensive, element in the computation of PH is the *complex* constructed from the input data. In general, PH tools use and focus on optimizing simplicial complexes; less frequently cubical complexes are also studied. This paper develops a method to construct *polytopal complexes* (or complexes constructed of any mix of convex polytopes) in any dimension $\mathbb{R}^n$. In general, polytopal complexes are significantly smaller than simplicial or cubical complexes. This paper includes an experimental assessment of the impact that polytopal complexes have on memory complexity and output results of a PH computation.**

***Index Terms*—Polytopal Complex, Persistent Homology, Convex Decomposition, Stereo-graphic projection, Data Mining.**

I. Introduction

The motivation for this paper is to reduce the memory requirement of complexes representing the topological space under observation. The paper explores the construction of polytopal complexes for use in the computation of *Persistent Homology (PH)*. Polytopal complexes are a generalized representation of complexes that include polytopes as a basic building unit. While similar to the simplices of simplicial complexes, polytopal complexes can contain a mix of polytopes [1]–[8]. That is, polytopal complexes represents the boundary maps using vertices, edges, polygons, polyhedrons, and higher dimensional polytopes.

Preliminaries. The topology of the underlying space is concerned with properties that are preserved under continuous deformations such as stretching and bending without gluing holes together and tearing surface apart. In general topologies are unaffected by the metric used for measurement but are mainly concerned with the continuity of the space.

A *Topological Space* $T(X, \tau)$ is defined for the set X and its topology $\tau = \{\tau_i \mid \tau_i \subset X\}$ such that:

- ϕ, the empty set, and X are in τ,
- The union of any two elements of τ are in τ, and
- The intersection of two elements of τ are in τ.

The topological space $T(X, (\phi, X))$, where τ is just a collection of ϕ and X is known as *chaotic topology*. This kind of topology has no notion of distance and can not distinguish far and near in topological spaces. The only concerned aspect is the continuum, as long the points lies on same continuous surface they are indistinguishable. Another topological space of interest is $T(X, P(X))$, where τ is a subset of power set of X and is known as *discrete topology*.

Point Cloud Data (PCD) is the primary format to represent the discrete topology governed by the notion of a distance metric. The PCD in conjunction with a distance metric is expected to preserve the topological invariants of the underlying space. The effect of PCD sub-sampling has been studied to reduce the space required for computing PH while preserving the topological features in the data [9]–[11].

TDA is an approach to data science that identifies topological features of data to provide insight into the topological invariants of the space [12]. While there are multiple methods that perform TDA, the most widely used method is the computation of PH. PH characterizes the persistence of the homologies that appear and disappear at discrete connectivity distances. The output of PH is a set of *Persistence Intervals* (PIs) of the form $\langle dim, birth, death \rangle$, that characterize the *dim*ension of the feature, the connectivity distance (*birth*) when the feature first forms, and the connectivity distance (*death*) when the feature disappears [13], [14]. Computing PH using various complexes has been explored [15]–[20].

Data Reduction and Sparsification manipulate the input point cloud to enable the computation of PH on large data sets. Sparsification by sub-sampling can be achieved either by a *Sparsified Rips complex* [21], *Graph Induced Complex* [22] or *Partitioned Persistence Homology* [10], [23]. The *Alpha complex* is a sparsified representation that supports PH computation on large data sets [24]. In general, the complexity of complex construction increases as their size reduces, *e.g.,* the Alpha complex size reduction comes at the cost of first computing a Delaunay triangulation. Likewise, a CW-complex can provide a succinct representation of the topology of space but it is difficult to generate [25].

Applications. As more efficient algorithms for TDA computations emerge, its utilization to various application domains have expanded. TDA provides data science practitioners important tools for certain data science problems. In *computational biology*, TDA can help identify the cell differentiation trajectories, signatures for protein folding, signatures to different cancer types in ontology, gene sequences, and cardiovascular diseases [26]. Other areas that also draw useful insights from TDA analysis include: network analysis [27],

Support for this work was provided in part by the National Science Foundation under grant IIS–1909096.

[28], digital images [29]–[32], and many others.

Contribution. Polytopal complexes and their geometric realization has been studied [6]–[8]. However, no previous work has developed a practical mechanism for their construction in $\mathbb{R}^n$. This paper develops a method to construct polytopal complexes suitable for use in the computation of PH in any dimension $\mathbb{R}^n$. A polytopal complex differs from a simplicial complex or cubical complex in that a polytopal complex can be constructed of a mix of cells formed from any convex polytope. In this work, the polytopal complex is obtained from Delaunay simplices by gluing adjacent simplices into polytopes using *Convex Decomposition (CD)*. To compute lower dimensional convex decompositions, a series of stereo-graphic projections are used to obtain a more efficient polytopal complex containing the highest degree polytopes possible.

The remainder of this paper is organized as follows. Section II presents a background discussion of the different types of complexes used in TDA. Section III presents the work related to convex decomposition and convex polytopes in the context of a polytopal complex. Section IV describes the method used in this paper to construct polytopal complexes. Section V evaluates the impact that using a polytopal complex has on: (i) the size of the complexes constructed (contrasted to Delaunay Complexes) and (ii) the impact that using triangulated polytopal complexes has on the output of a PH computation. Finally, Section VI provide some closing remarks on this work and discuss future work possibilities.

II. Background : Topological Complexes

This section reviews simplicial, cubical, Δ, polytopal, and CW-complexes. *Simplicial complexes* are sets of simplices that are simple to construct but large in size. Vietoris-Rips (VR) complexes are combinatorial simplicial complexes that are easy and fast to construct but often result in the largest complexes of this type [33]. Čech complex, Alpha complex, and Clique complexes are more compact than VR complexes; however, their construction is more computational expensive and difficult, especially in higher dimensions [34], [35].

Cubical complexes are analogous to simplicial complexes except that they are composed of points, lines, squares, cubes and their n-dimensional analogues. The bottom up construction of cubical complexes using lifting and cubifications operation has been discussed [36]. The disadvantage of this approach is that it is not known whether every even faced cubical $d-1$ sphere admits a d-cubifications [37], [38]. Cubical complexes are most suitable for digital image processing where data is cubical as 2D image consist of square pixels and 3D image consist of 3D voxels [39]–[41].

Δ-*complexes* are similar to simplicial complexes but removes the restrictions that (i) every $n+1$ vertices must represent one unique n-simplex, and (ii) triangulation needs not to have a geometric realization. A Δ-complex can be simplicial or cellular. A Δ-complex is *regular* if every k-cell in complex has $k+1$ distinct vertices. A regular Δ-complex is *proper* (or simplicial) if each subset of $k+1$ vertices is incident to at most one k-cell, or equivalently, if the intersection of any two cells in X is (the image of) a face of both cells. A proper Δ-complex is equivalent to a geometric simplicial complex.

Polytopal complexes (the main topic of this paper) use convex d-polytopes as their constituent building blocks. A general discussion on different types of polytopes and their significance in the construction of the polytopal complex can be found in [1]–[5]. However, for the purposes of this paper, convex polytopes form the boundary maps of the polytopal complex. Since every non-convex polytope has a convex decomposition, it is always possible to construct polytopal complex in any dimension. Polytopal complexes and their geometric realization has been studied in [6]–[8]. The ordered d-polytope can be represented by its lower dimensional convex polytopal face vectors [42].

CW complexes are a more generalized complex and are often known for much smaller representations than simplicial complexes. The CW complex drops the requirement that gluing maps are cellular [43]–[45]. A CW complex in 2D is equivalent to graphs with self loops and parallel edges.

Regular CW complex are a restricted version of a CW complex where parallel edges and self loops are not permitted. They are somewhat easier to construct and, like CW complexes, require fewer cells to represent a topological space [46]. Simplicial complexes are related to regular CW complexes as triangulations are related to pseudo-triangulations.

The complexes discussed above have an inclusion order by their restrictive gluing rules:

$$Complex_{\Delta} \xrightarrow{\Delta} Complex_{regCW} \xrightarrow{embedding} Complex_{CW}$$

The regular CW complex can be subdivide a into a Δ-complex using boundary cone structures by induction over skeleta. A CW complex is regular if its characteristic maps can be chosen to be embeddings. Furthermore:

$$Complex_{Simp} \xrightarrow{simp} Complex_{Poly} \xrightarrow{poly} Complex_{CW}$$

Polytopal complexes are restrictive CW complexes having gluing maps on convex ordered polytopes.

III. Related Work

The construction of polytopes and other geometrical objects can be found in [47]. A wide collection of techniques and algorithms regarding combinatorial and computational aspects of polytopes and polytopal complexes are presented in [48], [49]. To the best of our knowledge, no literature exists for the construction of polytopal complexes from PCD in any dimension $\mathbb{R}^n$. This work uses higher dimensional convex decomposition to generate polytopal complexes. Generating an optimal convex decomposition is an NP-hard problem [50], [51]. Fortunately, good approximate methods of convex decomposition exist and are sufficient [52], [53].

The Qhull convex decomposition algorithm is used in this work [54]. The approximation of *Convex Decomposition on Spherical Surfaces (CDSS)* is achieved by stereo-graphic projections in general dimension [55]. The stereo-graphic projection techniques are topology preserving conformal projections

Algorithm 1 MERGENEIGHBORS($poly$,$simplices$)

Input: $polytope$; $simplices$
Output: convex polytope with neighbors merged

1: $poly_{neigh} \leftarrow \phi$
▷ Find all immediate neighboring simplices
2: **for each** simplex $\sigma \in simplices$ **do**
3: **if** $\mathsf{len}(\cap(poly, \sigma)) = (\mathsf{len}(\sigma) - 1)$ **then**
4: $neighbors = neighbors \cup \sigma$
▷ Add simplices that grow the convex polytope
5: **for each** simplex $\sigma \in neighbors$ **do**
6: **if** $poly \cup \sigma$ **is** *convex-polytope* **then**
7: $poly = poly \cup \sigma$
8: **return** $poly$

Algorithm 2 CONVEXIZATION($simplices$)

Input: $simplices$
Output: set of maximal polytopes

1: $convex_{polys} \leftarrow \phi$
2: **for each** simplex $\sigma \in simplices$ **do**
3: $poly \leftarrow \sigma$
▷ Grow polytope by adding neighboring simplices
4: **while** true **do**
5: $P \leftarrow$ MERGENEIGHBORS($poly, simplices$)
6: **if** $P = poly$ **then break**
7: $poly \leftarrow P$
8: $convex_{polys} = convex_{polys} \cup poly$
9: **return** sortByMaxSizeAndFacet($convex_{polys}$)

Algorithm 3 CONVEXDECOMP($simp_{tri}$)

Input: $simp_{tri}$: a triangulated mesh
Output: ordered set of convex polytopes

1: $polytopes \leftarrow \phi$
2: $rem \leftarrow simp_{tri}$
3: **while** true **do**
4: $convex_{polys} \leftarrow$ CONVEXIZATION($simp_{tri}$)
▷ Collect non-overlapping polytopes from ordered list
5: **for** polytope cp_i in $convex_{polys}$ **do**
6: **if** $cp_i \not\subset (polytopes)$ **then**
7: $polytopes \leftarrow polytopes \cup cp_i$
▷ Remove simplices that are in a polytope
8: **for each** polytope $poly_i$ **in** $polytopes$ **do**
9: **for each** simplex $s \in poly_i$ **do**
10: $rem \leftarrow rem \backslash s$
11: **if** $rem = simp_{tri}$ **then**
12: **return** $polytopes \cup rem$
13: **else**
14: $simp_{tri} \leftarrow rem$

[56], [57]. The Schlegel diagram is a projection of a polytope from $\mathbb{R}^d$ into $\mathbb{R}^{d-1}$ and can be generated using stereo-graphic projections [58]. Algorithms to generate Schlegel diagrams have been studied in Chebyshev d-spheres for the convex polytope are utilized to guarantee non-overlapping edges in the projection space [59]. Stereo-graphically projected points on a d-hyperplane then can be reduced to d-1 dimension using Principal Component Analysis (PCA). computation of PH on polytopal complexes have been studied in [60].

IV. POLYTOPAL COMPLEX GENERATION AND PH

This work defines polytopal complexes that use convex polytopes as their cellular maps. This modification in the cellular mapping provide polytopal complexes a reduced memory representation compared to simplicial complex representations. Polytopal complexes consist of d-polytopes instead of d-simplices and, in this work, are generated using convex decomposition of PCD triangulation. The polytopes and their cellular maps generate the underlying manifold of the PCD.

This approach constructs polytopal complexes by first constructing the highest order polytopes from the highest order Delaunay cells. The *lower dimensional polytope surfaces* are then recursively constructed dimension by dimension. In particular, this approach uses stereo-graphic projection to project the polytopes in the complex from $\mathbb{R}^d$ to $\mathbb{R}^{d-1}$. The convex decomposition on polytope surfaces are then constructed in dimension $\mathbb{R}^{d-1}$. This process repeats until $d - 1 = 2$. The remainder of this section presents this method in detail.

A. Convex Decomposition

The convex decomposition code is constructed using two helper functions, namely: MERGENEIGHBORS (Algorithm 1) and CONVEXIZATION (Algorithm 2). MERGENEIGHBORS extends a polytope by merging any immediate neighboring simplices that produces a larger convex polytope. The algorithm first locates all immediate neighboring simplices of the polytope. A simplex is an immediate neighbor if it shares a common facet with the polytope (Line 3) (a d-simplex with $d + 1$ vertices has another d-simplex as neighbor if both of them shares d vertices). The algorithm then extends the current polytope by merging the immediate neighboring simplices that also produces a convex polytope.

Algorithm 2 (CONVEXIZATION) generates a maximal polytope for *each* simplex in the triangulation.[1] The algorithm examines each simplex and grows the maximum convex polytope from that simplex by iteratively merging the immediate neighbors (using MERGENEIGHBORS) until the convex polytope is no longer able to add new simplices to itself. This maximum convex polytope is then added to the candidate polytope set. Finally, the algorithm returns a list of candidate polytopes sorted in order by their sizes (number of vertices) and then by minimum polygon weight (defined as the maximum length facet on its boundary).

Algorithm 3 (CONVEXDECOMP) inputs a triangulated mesh of the data and builds the largest set of convex polytopes found in that data. The algorithm uses the CONVEXIZATION function to collect an ordered set of maximal convex polytopes from the triangulated mesh. These convex polytopes are then filtered to the largest set of non-overlapping convex polytopes (stored in *polytopes* at Line 7). That is, any smaller convex polytopes containing simplices that are also contained in some larger convex polytope are removed from the list. The subset of simplices that are part of the *polytopes* set are then removed from $simp_{tri}$ so that the CONVEXIZATION function can be

[1]Note this will grow a set of maximal polytopes possible from each simplex (what are called *candidate polytopes*); this will be filtered into a final polytope set in Algorithm 3.

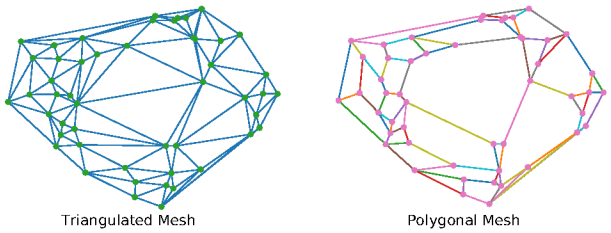


Fig. 1: Convex decomposition of 2D point cloud. Input triangulated mesh transformed to a polygonal mesh.

reinvoked to attempt the construction of more polytopes. This process is repeated until no additional polytopes are found. The function returns the set of maximal convex polytopes and any remaining simplices that did not merge into a polytope.

The convex decomposition of a point cloud in $\mathbb{R}^2$ is shown in Figure 1. The input triangulated mesh is shown in the left graphic and polygonal composed output is shown to the right. Algorithm 3 is organized to build polytopes by size and weight instead of by minimizing the total number of convex parts. The convex decomposition obtained by Algorithm 3 is unique and reproducible for given ordered input triangulated mesh. This paper focuses on constructing an exact convex decomposition to avoid any change in the resultant features. While additional reductions may be possible with *Approximate Convex Decomposition (ACD)*, the algorithms that generalize ACD to dimensions above $\mathbb{R}^3$ have not yet been developed.

B. Lower Dimensional Decomposition

The next step in the construction of polytopal complex is to generate a face vector for the convex polytopes in the convex decomposition at $\mathbb{R}^n$. The face vector for a 2-polytope can not be reduced as every edge in the convex hull contributes to the lower boundary map. For d-polytopes with $d > 2$, the $d-1$ surface of a d-polytope can exhibit convex decomposition and can be represented by $d-1$ convex polytopes. This surface reduction can be obtained by computing convexization on polytopal face vectors. Convex decomposing the hyper-spherical surface is required to obtain the lower dimensional polytopes. As discussed in Section III, the decomposition on the surface of the hyper-sphere is extremely difficult due to positive curvature. Furthermore, the presence of antipodes on spherical surfaces adds to this complexity.

To obtain lower dimensional facets, a conformal stereo-graphic projection technique is utilized. Conformal stereo-graphic projection of the hyper-spherical surface preserve the surface topology but is not isometric. To increase the efficiency and to generate maximal convex facets on hyper-spherical surface, a convex decomposition on multiple rotations of orthogonal stereo-graphic projections are performed. The selection of the projection points on the hyper-spherical surface is important as it can impact the projection efficacy.

The lower dimensional decomposition is performed by Algorithm 5 and the StereographicProj helper function (Algorithm 4). Algorithm 4 inputs the convex d-polytope having vertices and facets and returns a projected mesh of facets in d-1 dimension. The algorithm first computes the *half-space* (Line 1) representation of the convex polytope from its vertex representation. The hyper-plane for the projection facet is computed at Line 2. Next, the python package *pypoman* [61] is used to compute the Chebyshev hyper-sphere[2] radius (Line 3) and center (Line 4). The polytope vertices and their centroid are then projected onto the surface of the Chebyshev hyper-sphere (Lines 5 and 6). This projection step is important in order to avoid overlapping edges in the projected space. Finally, Lines 7–10 compute the projection of the polytopal point on projection hyper-plane computed in Line 2. The projection point for each vertex v_i on the polytope surface is identified by the intersection of the hyper-plane $facet_{plane}$ and the line joining $proj_{point}$ and v_i. The algorithm returns the projected lower dimensional surface mesh of the polytope in Line 11. The function returns a PCA reduction of the projected data to the next lower dimensional coordinates; the facets of the original polytope define the facets of the projected mesh.

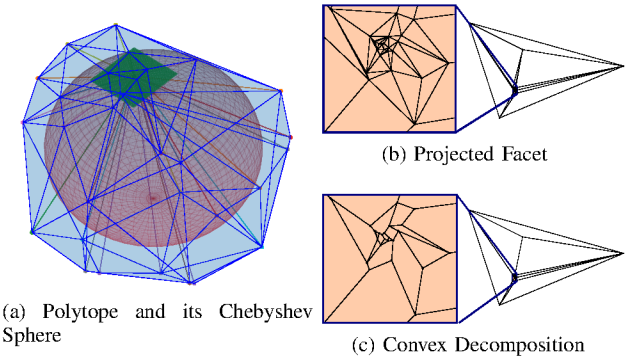


*Fig. 2: Lower dimensional decomposition. **(a)** The convex 2-polytope is projected into its Chebyshev sphere; **(b)** a stereo-graphic projection of the polytope vertices on 2-simplex; and **(c)** the corresponding lower dimensional convex decomposition of the projected vertices.*

Algorithm 4 StereographicProj($poly$, $facet$)

Input: $poly$: convex polytope; $facet$: projection facet
Output: stereo-graphic projected facet mesh in $\mathbb{R}^{d-1}$

1: $hs \leftarrow$halfspaces($poly$)
2: $facet_{plane} \leftarrow$hyperplane($facet$)
3: $chR \leftarrow hs.chR$ ▷ Chebyshev radius
4: $chXc \leftarrow hs.chXc$ ▷ Chebyshev center
▷ Project onto Chebyshev sphere
5: $proj_{poly} \leftarrow \frac{((poly.vertices - chXc) * chR)}{dist(chXc, poly.vertices)} + chXc$
6: $proj_{point} \leftarrow \frac{((facet.centroid - chXc) * chR)}{dist(chXc, facet.centroid)} + chXc$
7: $projections \leftarrow \phi$
8: **for each** $v_i \in proj_{poly}.vertices$ **do**
9: $p \leftarrow$intersection($facet_{plane}, line(proj_{point}, v_i)$)
10: $projections \leftarrow projections \cup p$
11: **return** ToMesh(PCA($projections$, d-1), poly.facets)

Figure 2a depicts a polytope surface (blue) in $\mathbb{R}^3$, its Chebyshev sphere (red), and the projection plane (green). Figure 2b represents the stereo-graphic projection of polytopal surface on the projection plane. The extreme points on the projected plane represent the polytope facet selected for projection. Finally,

[2]The Chebyshev sphere represents the largest hyper-sphere within a polytope surface boundary.

Algorithm 5 LOWDECOMP($poly, numProj$)

Input: $poly$: convex polytopes; $numProj$: number of projection trials
Output: $lower_{level}$: polytopal surface decomposition

1: $facets \leftarrow$SELECTPROJECTIONFACETS($poly, numProj$)
2: $polytopes \leftarrow \phi$
▷ Perform multiple stereo-graphic projections
3: **for each** face $f \in facets$ **do**
4: $mesh \leftarrow$STEREOGRAPHICPROJ($poly, f$)
5: $polys \leftarrow$CONVEXDECOMP($mesh$)
6: $polytopes \leftarrow$MERGEMAX($polytopes, polys$)
7: $lower_{level} \leftarrow \phi$
▷ Capture the maximal polytopes from the projections
8: **for each** polytop $P \in polytopes$ **do**
9: $proj_{poly}$ =AlignPlaneToPolytope($poly, P$)
10: $lower_{level}$.append($proj_{poly}$)
11: **return** $lower_{level}$

Figure 2c shows the convex components obtained from the projected facets. It should be noted that the convex components for projected points do not undergo efficient decomposition for regions close to extreme points; thus, it becomes necessary to perform multiple stereo-graphic projections from different projection facets to obtain the best convex face vectors.

Algorithm 5 attempts to obtain the maximal convex decomposition of the polytopal surface. The first task of Algorithm 5 is to generate the candidate facets for the projection target. These candidates are identified by the function SELECTPROJECTIONFACETS (implementation omitted). These selected facets are selected based on the distribution of vertices on the polytopal surface (ideally regions of low density are best targeted). As each projection is performed, the MERGEMAX (implementation omitted) function selects the maximal convex decomposition from each successive projection. Finally, Algorithm 5 rebuilds the centroid aligned projections of the convex facets at Lines 8–11 and returns the result.

The main steps of Algorithm 5 are depicted in Figure 3; he left column of images show a tetra-sphere (top left) and a cubical-sphere (bottom left) with vertices shown in green. The tetra-sphere is shown with 4 (d+1) projection facets and the cubical sphere is shown with 6 (2*d) projection facets. The graphics in the center column correspond to the stereo-graphic projections obtained on the candidate projection facets. The projections captures one opposite polygonal loop for the tetra-sphere and three polygonal loops for the cubic-sphere. The projections appear similar due to the symmetry of the tetra- and cubical-spheres; however, they actually represent different loops that appear opposite to the selected projection facet. These projections are then combined to obtain maximal convex components. The resulting lower dimensional polygonal candidate facets are shown in the right column of Figure 3. Algorithm 5 is successful in capturing the 4-polygonal facets from the tetra-sphere data, but captures more than 6 polygonal facets for the cubical-sphere. This error occurs because some of the facets are repeated as multiple projection can capture redundant facets. This error can be mitigated by using either additional projections or by selecting better candidate projection facets. The number of projections required can be

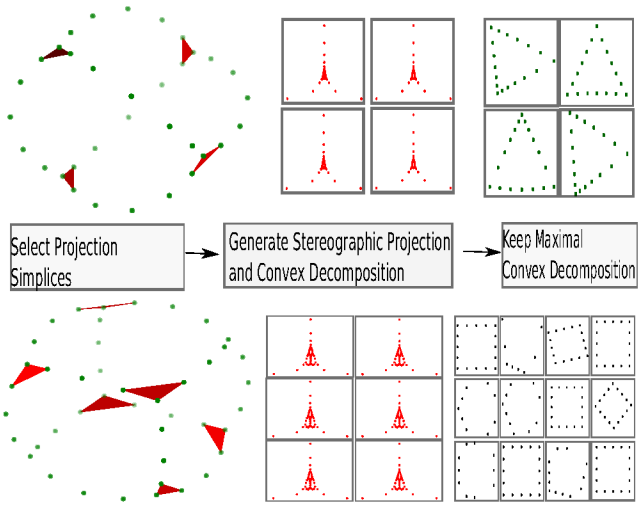


Fig. 3: Obtain lower order decomposition for tetra-sphere (top row) and cubical-sphere (bottom row). The left images shows the 2-simplices selected for the stereo-graphic projection step. The images in the middle show the corresponding projection on 2-simplices. The right image shows the convex decomposition of 2-tetra-sphere surface into 4 1-polygons and the loops on the 2-cubical-sphere surface.

reduced by computing the convex parts based on $d$spheres (points under stereo-graphic projection remain spherical but grows with projection angle) based convex components [62].

The complexity of Algorithm 4 is dependent on the computation complexity of the Chebyshev-sphere and the size of convex polytope. Computing the Chebyshev-sphere is an NP-hard problem for $d > 2$, but approximate computation can reduce the complexity [63]. This can be further reduced as the computation of Chebyshev-sphere can be avoided and any hyper-sphere with a center inside the polytope is sufficient. The overall complexity of the Algorithm 4 is $\mathcal{O}(n)$, where n is the number of polytope vertices. The complexity of Algorithm 5 is proportional to number of stereo-graphic projections performed; in particular, the complexity is $\mathcal{O}(kS)$, where k is the number of projections and S is the number of facets.

C. Polytopal Complex

The polytopal complex consists of the highest dimension polytopes and their lower dimensional face vectors. Algorithm 6 will generate a complete polytopal from an input PCD in $\mathbb{R}^n$. The algorithm generates the maximal convex facets dimension by dimension to obtain a reduced polytopal representation. The algorithm first computes the convex decomposition of a Delaunay triangulation of the PCD to obtain the highest dimension polytopes. The algorithm then works dimension by dimension to generate the lower dimensional face vectors for each polytope in the highest dimension polytopes. The space complexity of the Algorithm 6 is output sensitive. Convex polytopes with smaller face vectors will undergo maximum reductions; convex polytopes with large facets (in term of vertices) will experience larger reductions.

D. Persistent Homology

Most of the recent work with algorithm development and optimizations for persist homology use simplicial complexes.

Algorithm 6 POLYTOPALCOMPLEX(pcd)

Input: pcd: point cloud data; dim: dimension of pcd
Output: Polytopal Complex

```
poly_complex[dim] ←CONVEXDECOMP(Delaunay(pcd))
for d in [dim-1, ··· , 2, 1] do
    poly_complex[d] ← φ
    for pol_i in poly_complex[d+1] do
        polys ←LOWDECOMP(pol_i)
        poly_complex[d] ← poly_complex[d] ∪ polys
return poly_complex
```

While, *Singular homology* does permit the definition of the homology for a topological space that is independent of simplices or cells representations [64], there are not well developed tools for singular homology and its use is outside the scope of this paper. As a result, this paper studies the impact that polytopal complexes have on the computation of PH using a corresponding canonical simplicial subdivision of the polytopal complex. That is, since simplicial complex and polytopal complex have the same underlying spaces and complexes with the same underlying spaces have same homology groups, the polytopal complex can be replaced by a corresponding canonical simplicial complex. While decomposition of polytopal complex to simplicial complex expands the number of cells in the complex, the simplicial complex obtained from polytopal complex is still smaller than the corresponding Delaunay complex as only the surface facets are utilized for lower dimensional simplices. Thus, in this work this simplicial representation is used to compute the PH and show that a polytopal complex preserves the PCD homology.

V. EXPERIMENTAL RESULTS

Polytopal and Delaunay complexes are compared to address two issues, namely: (i) complex size comparisons, and (ii) PH outputs. The comparisons are made using 5 synthetic and 2 real-world data sets. The 5 synthetic data sets are: (i) a *Tetrahedron-Sphere*, (ii) a *Cubical-Sphere*, (iii) a *Permutahedron*, (iv) a *Fibonacci-Sphere*, and (v) a *d-Sphere* ($d = 2$ and $d = 3$). The Tetrahedron-Sphere, Cubical-Sphere and Permutahedron are generated by distributing points uniformly on their edges and projecting them to unit hyper-Sphere. The Fibonacci-sphere and d-sphere data is generated from the python libraries FIBLAT and TADASETS. The 2 real-world data sets are the *lion* and *flamingo* triangulated data sets taken from the UCI triangulated shapes database [65]. PH results are compared using the *Sliced Wasserstein* (SW) distance [66].

The complex size comparisons for the synthetic and real-world data is summarized in Tables I and II. Since the synthetic test data is projected onto a unit hyper-sphere, it can be organized and studied in any dimension. In this experiment the data is projected on to a sphere in $\mathbb{R}^3$ (the first two rows) and a sphere in $\mathbb{R}^4$ (the last two rows). The test set (spheres in $\mathbb{R}^3$ and $\mathbb{R}^4$) contain two trials, one with $d + 1$ stereo-graphic projections and one with $2d$ stereo-graphic projections (projections are not required for the 1-cell [edge] and 2-cell [polygon]). This is done to assess the impact that higher projection degree have on the results. The 0-cell (points) is not shown as it is equal in all cases (although they are included in the row totals). In all test cases and in all dimensional decompositions, the polytopal complex contains fewer cells than the corresponding Delaunay complex. Finally, increased projections do not always result in larger reductions.

The real-world data in Table II shows lower reductions than achieved with the synthetic data. As with the synthetic data, comparisons with $d+1$ and $2d$ stereo-graphic projections were performed (shown column wise). Interestingly the additional projections did not significantly impact the reduction. The reductions achieved, however, are significant and this could have dramatic impact on growing the size of point clouds that can be processed by PH tools. Furthermore, this expanded capability should come at a better fidelity of the results from the PH computation than is achieved with sub-sampling.

The output of a computation of PH is a set of *Persistence Intervals* (PIs) that characterize the *dimension* of a topological feature as well as the connectivity distance at which the feature first appears (its *birth*) and the connectivity distance where the feature no longer exists (its *death*). This second aspect of the experimental assessment of polytopal complexes computes the PH twice: once using a Delaunay complex, and once using a polytopal complex. The outputs from these trials are separated by homology group (H_0, H_1, etc) and then compared (homology group to homology group) using the SW distance. The experiments were performed using $d+1$ stereo-graphic projections. In a manner similar to the size studies, the synthetic data testing was performed with the data in $\mathbb{R}^3$ and $\mathbb{R}^4$. Of course the real-world test data is only in $\mathbb{R}^3$.

The results from the $\mathbb{R}^3$ test data are shown in Table III and from $\mathbb{R}^4$ in Table IV. The results from all of the $\mathbb{R}^3$ data show very good results. The SW distance numbers show that the PI results are very close to each other; the H_2 data is identical. The results from the $\mathbb{R}^4$ tests are similar for all tests except for the H_1 and H_2 results for the Permutahedron; most likely this is due inefficiency of stereo-graphic projections.

VI. CONCLUSION AND FUTURE WORK

A polytopal complex will reduce the memory requirement of the complex used to compute Persistent Homology. The method utilizes two well studied areas of *Convex Decomposition (CD)* and *Convex Decomposition on Spherical Surfaces (CDSS)*. Both of these problems are hard to generalize with CDSS known to be NP-hard problem. Previous studies for both have been limited to $\mathbb{R}^3$. This paper explores convex decomposition and stereo-graphic projection based spherical surface decomposition algorithms suitable for the computation of PH. The main result of this paper is the capability of this approach to drastically reduce memory requirements for computing PH by using a polytopal complex instead of the more traditional simplicial complex. Finally, the use of approximate methods for CD and CDSS may well provide an additional significant reduction in complex size. The construction of generalized *approximate polytopal complexes* using Approximate Convex Decomposition and Convex Approximation of Spherical Surfaces techniques is an interesting future research direction.

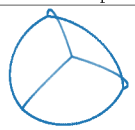
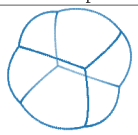
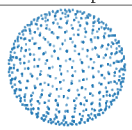
Parameters			Synthetic Datasets														
			Tetrahedron-sphere			Cubical-sphere			Permuatahedron			Fibonacci-sphere			d-Sphere		
Data Dim	No. of Projections	Simplicial Dim															
Complex Type			Poly	Del	$\frac{Poly}{Del}$%	Poly	Del	$\frac{Poly}{Del}$%	Poly	Del	$\frac{Poly}{Del}$%	Poly	Del	$\frac{Poly}{Del}$%	Poly	Del	$\frac{Poly}{Del}$%
$\mathbb{R}^3$	—	1-cell	176	769	22.88%	107	787	13.59%	261	803	32.50%	22	787	2.79%	55	788	6.98%
	—	2-cell	4	954	0.41%	53	978	5.41%	46	998	4.60%	260	978	26.58%	205	980	20.91%
	4	3-cell	1	380	0.26%	1	390	0.25%	1	398	0.25%	1	390	0.25%	1	391	0.25%
	d+1	**Total**	**377**	**2103**	**17.93%**	**361**	**2155**	**16.75%**	**512**	**2199**	**23.28%**	**483**	**2155**	**22.41%**	**461**	**2159**	**21.35%**
$\mathbb{R}^3$	—	1-cell	192	796	24.12%	116	787	14.73%	276	803	34.37%	22	787	2.79%	37	789	4.68%
	—	2-cell	10	954	1.04%	24	978	2.45%	34	998	3.44%	327	978	33.43%	244	982	24.84%
	6	3-cell	1	380	0.26%	1	390	0.25%	1	398	0.25%	1	390	0.25%	1	392	0.25%
	2d	**Total**	**399**	**2103**	**18.97%**	**341**	**2155**	**15.82%**	**515**	**2199**	**23.41%**	**550**	**2155**	**25.52%**	**482**	**2163**	**22.28%**
$\mathbb{R}^4$	—	1-cell	417	561	74.33%	451	592	76.18%	541	777	69.62%	548	901	60.82%	550	937	58.69%
	—	2-cell	515	1109	46.43%	642	1217	52.75%	964	1707	56.47%	1222	2099	58.21%	1161	2209	52.55%
	4	3-cell	145	990	14.60%	180	1101	16.34%	251	1546	16.23%	247	1970	12.53%	249	2082	11.95%
	5	4-cell	1	328	0.30%	1	365	0.27%	1	500	0.20%	1	653	0.15%	1	691	0.14%
	d+1	**Total**	**1193**	**2988**	**39.92%**	**1386**	**3275**	**42.32%**	**1874**	**4530**	**41.36%**	**2138**	**5623**	**38.02%**	**2081**	**5919**	**35.15%**
$\mathbb{R}^4$	—	1-cell	391	561	69.69%	435	592	73.47%	443	777	57.01%	389	901	43.17%	426	925	46.05%
	—	2-cell	456	1109	41.11%	731	1217	60.06%	1089	1707	63.79%	1636	2099	77.94%	1573	2175	72.32%
	6	3-cell	118	990	11.91%	155	1101	14.07%	199	1546	12.87%	221	1970	11.21%	230	2050	11.21%
	8	4-cell	1	328	0.30%	1	365	0.27%	1	500	0.20%	1	653	0.15%	1	681	0.14%
	2d	**Total**	**1081**	**2988**	**36.17%**	**1434**	**3275**	**43.78%**	**1849**	**4530**	**40.81%**	**2367**	**5623**	**42.09%**	**2350**	**5831**	**40.30%**

TABLE I: Size comparisons of the polytopal complex to a Delaunay complex for the synthetic test data. The test data sets in $\mathbb{R}^3$ contain 200 points and the test data sets in $\mathbb{R}^4$ contain 120 points.

DataSet		**Delaunay**	$(d+1)$**proj**	$\frac{Poly}{Del}$%	$2d$ **proj**	$\frac{Poly}{Del}$%
Flamingo	1-cell	3718	3438	92.46%	3452	92.84%
	2-cell	6351	5823	91.68%	5656	89.05%
	3-cell	3132	2538	81.00%	2538	81.00%
	Total	**13201**	**11799**	**89.37%**	**11646**	**88.22%**
Lion	1-cell	3585	3307	92.24%	3314	92.44%
	2-cell	6100	5593	91.68%	5364	87.93%
	3-cell	3014	2357	78.20%	2357	78.2%
	Total	**12699**	**11257**	**88.64%**	**11035**	**86.89%**

TABLE II: Size comparisons of the polytopal complex to a Delaunay complex for the real-world test data with 500 points each in $\mathbb{R}^3$.

Point Cloud Data	SW H_0	SW H_1	SW H_2
Tertahedron-Sphere	0.065292	1.500187	0.0
Cubical-Sphere	0.101513	0.595516	0.0
Permutahedron	0.132509	1.854796	0.0
Fibonacci-Sphere	0.097517	0.064931	0.0
2-Sphere	0.253665	1.337124	0.0
Lion	0.043305	0.464385	0.0
Flamingo	0.041982	0.629074	0.0

TABLE III: SW distance between the output PIs of a polytopal and Delaunay complex. The 200 point synthetic data is in $\mathbb{R}^3$ (0.1 noise). The 500 point real-world data is in $\mathbb{R}^3$

Point Cloud Data	SW H_0	SW H_1	SW H_2	SW H_3
Tertahedron-Sphere	0.253969	2.244554	1.181420	0.0
Cubical-Sphere	0.101676	0.246004	1.145022	0.0
Permutahedron	0.063563	10.281318	3.520064	0.0
Fibonacci-Sphere	0.219374	1.157760	0.133991	0.0
3-Sphere	0.212961	0.225861	0.641286	0.0

TABLE IV: SW distance between the PIs of a polytopal and Delaunay complex. The 120 point synthetic data is in $\mathbb{R}^4$ (0.1 noise).

REFERENCES

[1] A. Choudhary, M. Kerber, and S. Raghvendra, "Polynomial-sized topological approximations using the permutahedron," *Discrete & Computational Geometry*, vol. 61, pp. 42–80, Jan. 2019.

[2] H. Edelsbrunner and K. Ölsböck, "Tri-partitions and bases of an ordered complex," *Discrete & Computational Geometry*, vol. 64, no. 3, pp. 759–775, Mar. 2020.

[3] M. Goff, S. Klee, and I. Novik, "Balanced complexes and complexes without large missing faces," *Arkiv för Matematik*, vol. 49, no. 2, pp. 335–350, Oct. 2011.

[4] K. Iriye and D. Kishimoto, "Fat-wedge filtration and decomposition of polyhedral products," *Kyoto Journal of Mathematics*, vol. 59, no. 1, pp. 1–51, Apr. 2019.

[5] A. Postnikov, "Permutohedra, associahedra, and beyond," *International Mathematics Research Notices*, vol. 2009, no. 6, pp. 1026–1106, 2009.

[6] E. Katz, "Tropical realization spaces for polyhedral complexes," in *Algebraic and Combinatorial Aspects of Tropical Geometry, Contemporary Mathematics*, E. Brugallé, M. A. Cueto, A. Dickenstein, E.-M. Feichtner, and I. Itenberg, Eds., vol. 589. Providence, RI: American Mathematical Society, 2013, pp. 235–251.

[7] I. Pak and S. Wilson, "Geometric realizations of polyhedral complexes," Department of Mathematics, UCLA, Tech. Rep., 2014.

[8] S. M. Wilson, "Embeddings of polytopes and polyhedral complexes," Ph.D. dissertation, University of California, Los Angeles, 2012.

[9] F. Chazal, B. T. Fasy, F. Lecci, B. Michel, A. Rinaldo, and L. Wasserman, "Subsampling methods for persistent homology," in *Int. Conference on Machine Learning*, ser. ICML 2015, Lille, France, Jul. 2015.

[10] N. O. Malott, A. Sens, and P. A. Wilsey, "Topology preserving data reduction for computing persistent homology," in *Int. Workshop on Big Data Reduction*. Piscataway, New Jersey: IEEE, 2020, pp. 2681–2690.

[11] V. de Silva and G. Carlsson, "Topological estimation using witness complexes," in *Eurographics Symposium on Point-Based Graphics*, ser. SPBG '04, M. Gross, H. Pfister, M. Alexa, and S. Rusinkiewicz, Eds. Goslar, DEU: The Eurographics Association, 2004, pp. 157–166.

[12] G. Carlsson, "Topology and data," *Bulletin of the American Mathematical Society*, vol. 46, no. 3, pp. 255–308, Apr. 2009.

[13] G. Carlsson, A. Zomorodian, A. Collins, and L. J. Guibas, "Persistence barcodes for shapes," *International Journal of Shape Modeling*, vol. 11, no. 02, pp. 149–187, 2005.

[14] R. Ghrist, "Barcodes: The persistent topology of data," *Bulletin of the American Mathematical Society*, vol. 45, no. 1, pp. 61–75, 2008.

[15] I. Itenberg, L. Katzarkov, G. Mikhalkin, and I. Zharkov, "Tropical homology," *Mathematische Annalen*, vol. 374, pp. 963–1006, Jun. 2019.

[16] M. Banagl, T. Mäder, and F. Sadlo, "Stratified formal deformations and intersection homology of data point clouds," May 2020.

[17] R. Gonalez-Diaz, J. Lamar, and R. Umble, "Cups products in z2-cohomology of 3d polyhedral complexes," *arXiv preprint arXiv:1207.2346*, 2012.

[18] P. Dłotko and T. Wanner, "Rigorous cubical approximation and persistent homology of continuous functions," *Computers & Mathematics with Applications*, vol. 75, no. 5, pp. 1648–1666, Mar. 2018.

[19] E. Saucan and J. Jost, "Network topology vs. geometry: from persistent

homology to curvature," in *30th Conference on Neural Information Processing Systems (NIPS 2016)*, Dec. 2016.
[20] H. Edelsbrunner and G. Osang, "The multi-cover persistence of euclidean balls," *Discrete & Computational Geometry*, vol. 65, no. 4, pp. 1296–1313, Mar. 2021.
[21] D. R. Sheehy, "Linear-size approximations to the vietoris–rips filtration," *Discrete & Computational Geometry*, vol. 49, no. 4, pp. 778–796, Jun. 2013.
[22] T. K. Dey, F. Fan, and Y. Wang, "Graph induced complex on point data," in *Proceedings of the Twenty-Ninth Annual Symposium on Computational Geometry*, ser. SoCG '13. New York, NY, USA: Association for Computing Machinery, 2013, pp. 107–116.
[23] N. O. Malott, "Partitioned persistent homology," Master's thesis, Department of Electrical Engineering and Computer Science, University of Cincinnati, Cincinnati, OH, Nov. 2020.
[24] H. Edelsbrunner, "Shape reconstruction with delaunay complex," in *Latin American Symposium on Theoretical Informatics*, C. L. Lucchesi and A. V. Moura, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 119–132.
[25] P. Dłotko, T. Kaczynski, M. Mrozek, and T. Wanner, "Coreduction homology algorithm for regular cw-complexes," *Discrete & Computational Geometry*, vol. 46, no. 2, pp. 361–388, Sep. 2011.
[26] Y. Skaf and R. Laubenbacher, "Topological data analysis in biomedicine: A review," *Journal of Biomedical Informatics*, vol. 130, Jun. 2022.
[27] D. Horak, S. Maletić, and M. Rajković, "Persistent homology of complex networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2009, no. 3, p. P03034, Mar. 2009.
[28] G. Petri, M. Scolamiero, I. Donato, and F. Vaccarino, "Topological strata of weighted complex networks," *PLOS ONE*, vol. 8, no. 6, pp. 1–8, Jun. 2013.
[29] P. Bendich, H. Edelsbrunner, and M. Kerber, "Computing robustness and persistence for images," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 1251–1260, Nov. 2010.
[30] G. Carlsson, T. Ishkhanov, V. de Silva, and A. Zomorodian, "On the local behavior of spaces of natural images," *International Journal of Computer Vision*, vol. 76, no. 1, pp. 1–12, Jan. 2008.
[31] K. Mischaikow and V. Nanda, "Morse theory for filtrations and efficient computation of persistent homology," *Discrete & Computational Geometry*, vol. 50, no. 2, pp. 330–353, 2013.
[32] H. Wagner, C. Chen, and E. Vuçini, "Efficient computation of persistent homology for cubical data," in *Topological Methods in Data Analysis and Visualization II: Theory, Algorithms, and Applications*, R. Peikert, H. Hauser, H. Carr, and R. Fuchs, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 91–106.
[33] A. Zomorodian, "Fast construction of the vietoris–rips complex," *Computer and Graphics*, vol. 34, pp. 263–271, Jun. 2010.
[34] J. F. Espinoza, R. Hernández-Amador, H. A. Hernández-Hernández, and B. Ramonetti-Valencia, "A numerical approach for the filtered generalized čech complex," *Algorithms*, vol. 13, no. 1, 2020.
[35] M. Lavrov, "Complexity of constructing the clique complex of the graph — mathematics stack exchange," https://math.stackexchange.com/questions/2466075/complexity-of-constructing-the-clique-complex-of-the-graph, Oct. 2017.
[36] A. Schwartz and G. M. Ziegler, "Construction techniques for cubical complexes, odd cubical 4-polytopes, and prescribed dual manifolds," *Experimental Mathematics*, vol. 13, no. 4, pp. 385–413, 2004.
[37] M. Bern, D. Eppstein, and J. Erickson, "Flipping cubical meshes," *Engineering with Computers*, vol. 18, pp. 173–187, Oct. 2002.
[38] D. Eppstein, "Linear complexity hexahedral mesh generation," *Computational Geometry*, vol. 12, no. 1–2, pp. 3–16, Feb. 1999.
[39] R. Gonzalez-Diaz, M. J. Jimenez, and B. Medrano, "Cubical cohomology ring of 3d photographs," *International Journal of Imaging Systems and Technology*, vol. 21, no. 1, pp. 76–85, 2011.
[40] I. Karaca and O. Ege, "Cubical homology in digital images," *International Journal of Information and Computer Science*, vol. 1, pp. 178–187, 2012.
[41] D. Ziou and M. Allili, "Generating cubical complexes from image data and computation of the euler number," *Pattern Recognition*, vol. 35, no. 12, pp. 2833–2839, 2002.
[42] M. Szilvási-Nagy, "Two algorithms for decomposing a polyhedron into convex parts," *Computer Graphics Forum*, vol. 5, no. 3, pp. 197–201, Sep. 1986.
[43] P. Brendel, P. Dłotko, G. Ellis, M. Juda, and M. Mrozek, "Computing fundamental groups from point clouds," *Applicable Algebra in Engineering, Communication and Computing*, vol. 26, no. 1-2, pp. 27–48, Jan. 2015.
[44] R. Forman, "Morse theory for cell complexes," *Advances in Mathematics*, vol. 134, no. 1, pp. 90–145, 1998.
[45] J. C. Hart, "Using the cw-complex to represent the topological structure of implicit surfaces and solids," in *ACM SIGGRAPH 2005 Courses*, ser. SIGGRAPH '05. New York, NY, USA: ACM, Jul. 2005, pp. 52–es.
[46] P. Desbarats and S. Gueorguieva, "Cw complexes: Topological mainframe for numerical representations of objects," in *Computational Science and Its Applications*, ser. ICCSA 2003, V. Kumar, M. L. Gavrilova, C. J. K. Tan, and P. L'Ecuyer, Eds. Berlin, Heidelberg: Springer, 2003, pp. 498–507.
[47] J.-D. Boissonnat and M. Teillaud, Eds., *Effective computational geometry for curves and surfaces*, ser. Mathematics and Visualization. Springer Berlin, Heidelberg, 2007.
[48] A. Bjorner and G. M. Ziegler, "Combinatorial stratification of complex arrangements," *Journal of the American Mathematical Society*, vol. 5, no. 1, pp. 105–149, Jan. 1992.
[49] A. Paoluzzi, V. Shapiro, and A. DiCarlo, "Regularized arrangements of cellular complexes," Aug. 2017.
[50] C. L. Bajaj and T. K. Dey, "Convex decomposition of polyhedra and robustness," *SIAM Journal on Computing*, vol. 21, no. 2, pp. 339–364, 1992.
[51] B. Chazelle and D. P. Dobkin, "Optimal convex decompositions," in *Computational Geometry*, ser. Machine Intelligence and Pattern Recognition, G. T. Toussaint, Ed. North-Holland, 1985, vol. 2, pp. 63–133.
[52] H. Liu, W. Liu, and L. J. Latecki, "Convex shape decomposition," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, Jun. 2010, pp. 97–104.
[53] K. Mamou, "Volumetric hierarchical approximate convex decomposition," in *Game Engine Gems 3*, E. Lengyel, Ed. AK Peters, 2016, pp. 141–158.
[54] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software*, vol. 22, no. 4, pp. 469–483, Dec. 1996.
[55] M. Zamboj, "Interactive 4-d visualization of stereographic images from the double orthogonal projection," in *International Conference on Geometry and Graphics*, ser. ICGG 2020. Springer International Publishing, 2021, pp. 115–126.
[56] I. M. Gel'fand, R. A. Minlos, and Z. Y. Shapiro, *Representations of the Rotation and Lorentz Groups and their Applications*. Courier Dover Publications, 2018.
[57] D. M. German, L. Burchill, A. Duret-Lutz, S. Pérez-Duarte, E. Pérez-Duarte, and J. Sommers, "Flattening the viewable sphere," in *Computational Aesthetics in Graphics, Visualization, and Imaging*, D. W. Cunningham, G. Meyer, and L. Neumann, Eds. The Eurographics Association, 2007, pp. 23–28.
[58] V. Schlegel, *Theorie der homogen zusammengesetzten Raumgebilde*. Engelmann, 1883, vol. 44.
[59] N. D. Botkin and V. Turova-Botkina, "An algorithm for finding the chebyshev center of a convex polyhedron," *Applied Mathematics and Optimization*, vol. 29, no. 2, pp. 211–222, Mar. 1994.
[60] D. Du, "Contributions to persistence theory," *Annals of West University of Timisoara-Mathematics and Computer Science*, vol. 52, no. 2, pp. 13–95, Dec. 2014.
[61] S. Caron, "Pypoman, a python module for polyhedral manipulations," 2018. [Online]. Available: https://scaron.info/doc/pypoman/
[62] J.-D. Boissonnat, A. Cérézo, O. Devillers, J. Duquesne, and M. Yvinec, "An algorithm for constructing the convex hull of a set of spheres in dimension d," *Computational Geometry*, vol. 6, no. 2, pp. 123–130, 1996.
[63] Y. Xia, M. Yang, and S. Wang, "Chebyshev center of the intersection of balls: complexity, relaxation and approximation," *Mathematical Programming*, vol. 187, no. 1-2, pp. 287–315, Feb. 2020.
[64] A. Hatcher, *Algebraic Topology*. Cambridge University Press, 2002, (available online: pi.math.cornell.edu/ hatcher/AT/ATpage.html).
[65] R. W. Sumner and J. Popovic, "Mesh data from deformation transfer for triangle meshes," 2004. [Online]. Available: https://people.csail.mit.edu/sumner/research/deftransfer/data.html
[66] M. Carriere, M. Cuturi, and S. Oudot, "Sliced wasserstein kernel for persistence diagrams," Nov. 2017.