Topological Consistency, Not Fidelity, Bounds the Cost Relief of Simplified EEG Brain Maps

An Evaluation of Five Boundary-Simplification Algorithms Across Six Cortical Atlases

Friso B. H. van der Veen[1]
Supervisor(s): Ricardo Marroquim[1]**, Arthur-Ervin Avramiea**[2]
[1]EEMCS, Delft University of Technology, The Netherlands
[2]Vrije Universiteit Amsterdam, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 21, 2026

Name of the student: Friso B. H. van der Veen
Final project course: CSE3000 Research Project
Thesis committee: Ricardo Marroquim, Arthur-Ervin Avramiea, Thomas Abeel

Abstract

The Neurophysiological Biomarker Toolbox (NBT) visualises biomarkers on two-dimensional cortical parcellations, where rendering cost limits use at clinical scale. Currently, NBT fits each region's boundary independently, leaving gaps and overlaps between adjacent regions that simplification only worsens. This work applies a shared-arc representation that is gap- and overlap-free by construction. Because boundaries are shared, simplification acts on both sides of the boundary at once, so gaps and overlaps cannot form during simplification. Two failure modes remain: a boundary can cross another, or a region can collapse to a degenerate polygon. Five established simplification algorithms are evaluated on the representation across six atlases and across three axes: geometric fidelity, topological consistency, and rendering cost.

The evaluation shows that the algorithms differ little on fidelity. The axis separating them is topological consistency, specifically whether they allow collapse. This matters for cost because in the evaluated algorithms the dominant predictor of rendering time is the number of faces, not the number of vertices. The algorithms that provide the strongest topological guarantees therefore deliver the least cost relief, since they collapse the fewest faces. On this representation, simplification stays free of gaps and overlaps at a small, bounded fidelity cost.

1 Introduction

During the past hundred years, electroencephalography (EEG) has been a cornerstone of the study of brain function and the diagnosis of brain disorders [19]. As a non-invasive method, it allows researchers and physicians to measure brain activity in live subjects. The Neurophysiological Biomarker Toolbox (NBT) [14, 24] is an open-source toolbox for performing statistical tests on EEG data and visualising them using brain atlases projected onto two-dimensional parcellated regions. As NBT aims to bridge the gap between research and clinical use, performance is critical. Slow performance for one researcher is inconvenient. For hospital-wide deployment, it could make the platform unsuitable.

Although brain atlases provide detailed maps of brain regions, EEG has inherently limited spatial resolution primarily due to volume conduction through the skull and scalp [2]; electrode count further limits what can be reliably localised in EEG recordings [17]. Rendering parcellations at full-atlas resolution plausibly exceeds the detail that can be meaningfully resolved from the underlying signal, so simplifying boundaries reduces the drawn geometry at limited cost to the detail the signal can support. This makes simplification a defensible design choice. The boundaries cannot be simplified as they stand, however. The existing boundary representation in NBT fits each region's boundary independently. Because of this, adjacent regions can have gaps or overlaps along their shared boundaries, and simplifying each boundary on its own only worsens them by driving adjacent regions further apart. A representation that merely starts gap- and overlap-free is not enough. It has to stay that way as its boundaries are simplified.

Representing a division of the plane into regions is a well-established topic in computational geometry [3]. Contour and polygon simplification has been studied for decades [7, 20, 23], including under constraints that keep a line consistent with neighbouring features [21]. The intersection of these fields, the simplification of an entire subdivision while preserving topological consistency, has been studied in its own right [4, 9]. However, the application of these techniques to projected cortical parcellations remains largely unexplored. The closest prior work, Hao et al. [13], refines region boundaries on three-dimensional morphological surfaces. NBT renders a two-dimensional projection sampled to a labelled point cloud, which is a different setting from the one they address.

This work's main contribution is the application of subdivision simplification to two-dimensional cortical parcellations using a representation that is gap- and overlap-free by construction. It further evaluates five established simplification algorithms on six cortical atlases along three axes: geometric fidelity, topological consistency, and rendering cost.

Its central research question is: *On a gap- and overlap-free representation of cortical parcellations, how does simplification affect geometric fidelity, topological consistency, and rendering cost?*

This decomposes into three sub-questions:

RQ1: How do simplification algorithms affect the geometric fidelity of the simplified boundaries?

RQ2: Under this representation, what topological failures remain possible, and when do they occur?

RQ3: What drives NBT's rendering cost, and how do the simplification algorithms relate to it?

The remainder of this work is organised as follows. Section 2 describes how NBT represents parcellation boundaries and why its per-arc fitting allows gaps and overlaps. Section 3 presents the shared-arc model which precludes them and introduces the simplification algorithms. Section 4 reports the evaluation across the three axes and shows that topological consistency rather than geometric fidelity separates the algorithms. Section 5 combines these axes and shows how the algorithms' topological guarantees relate to the cost relief they provide. Section 6 states the conclusion and open questions. Section 7 reflects on representational honesty and reproducibility.

2 Problem Description

EEG signals recorded on the scalp are attributed to specific cortical regions through source localisation. These sources are grouped as a *cortical parcellation*: a labelling of the cortical surface into regions of interest. NBT displays these parcellations on a 2D projection called a *flatmap* (shown in Figure 1), which is a sampling of the 3D vertices of the parcellated cortical surface to a 2D point cloud. The rendering of this projection requires the boundaries between adjacent labelled regions.

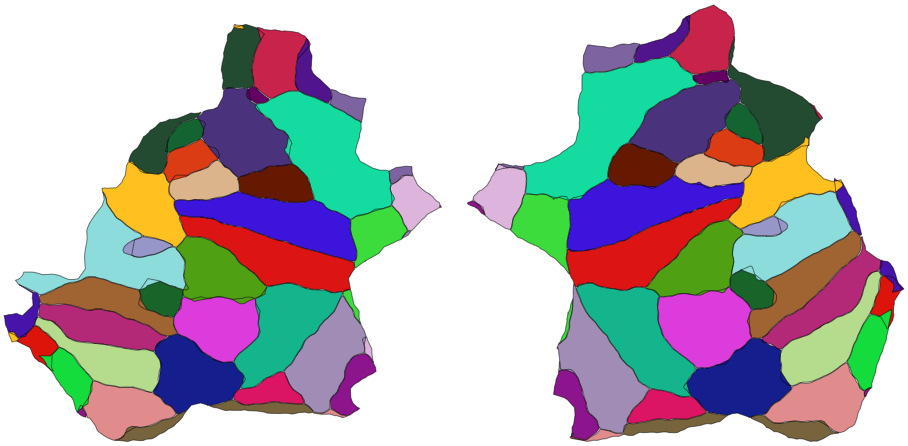

Figure 1: The Desikan-Killiany parcellation [5] projected to a flatmap, as currently used by NBT.

The existing parcellation boundary pipeline in NBT generates one independent *alpha-shape* per region. An alpha-shape [8] is a generalisation of the convex hull whose parameter α controls how tightly the boundary follows the points. Large α recovers the convex hull, while small α allows the hull to contract inwards, following concavities. At low enough α values the hull can start fragmenting. Each region's points are fitted by such a polygon independently, serialised into a `.patches` file and then rendered.

2.1 Boundary Representation

The current alpha-shape representation in NBT produces topological errors such as gaps and overlaps between adjacent regions. Because each region is fitted independently, the boundary shared by two regions is encoded as two separate copies with no guarantee that they coincide. Without explicit region adjacency, aggressive simplification can cause these boundaries to drift apart.

These errors are failures to create a valid *planar subdivision*: a partitioning of the plane into faces by a graph of straight-line edges, with faces labelled by region. A *gap* is a region of the plane not attributed to any face, while an *overlap* is a region attributed to more than one face. These are the two ways NBT's existing pipeline fails. Figure 2 shows these errors as they occur in NBT.

The dual of the planar subdivision is the *adjacency graph*. It tracks which regions border each other. Regions are represented by a single vertex, with edges representing adjacencies between regions.

Figure 2: Boundary representation for the frontal pole region in the left hemisphere of the Desikan-Killiany [5] atlas. Left: NBT's existing per-region boundaries that leave gaps (areas attributed to no regions), and overlaps (areas attributed to more than one region). Right: the Voronoi-based boundaries that cover the entire plane without gaps or overlap, producing a valid planar subdivision. Within-region boundaries are left out.

The labelled point cloud presents a natural alternative. Given a set of sites, a *Voronoi diagram* partitions the plane into cells. Each point on the plane is assigned to the cell of its nearest site. The partitioning covers the entire plane, so the cells on the periphery are unbounded. When computed over a labelled point cloud the Voronoi edges between cells of different labels provide the 1-nearest-neighbour boundary between region labels [1]. By construction, a Voronoi diagram covers the plane and therefore leaves no gaps. Additionally, each inter-region boundary is an edge between two cells, so no overlap occurs anywhere. Figure 2 contrasts NBT's existing boundaries, which exhibit gaps and overlaps, with those produced using a Voronoi diagram.

Constructing the boundaries through a Voronoi diagram implicitly handles the union of cells belonging to the same region. Ridges between cells of the same label are discarded, leaving only inter-region boundaries.

2.2 The Shared-ArcNode Model

A Voronoi diagram produces gap- and overlap-free boundaries, but only at construction. Simplifying each region's boundary independently splits the shared boundary into two copies that drift apart. Preserving the property therefore requires a model that stores each boundary once and shares it between two regions, so that simplification updates it for both at once.

Inter-region edges are collected into a topological model of arcs and nodes. The degree of a boundary vertex is the number of incident inter-region edges. A *node* is a boundary vertex with degree other than two: a junction where three or more regions meet, or the point at which a boundary intersects the diagram boundary. An *arc* is a polyline between two nodes separating exactly two regions, one of which may be the exterior. A region fully enclosed by a neighbour, an *island*, has a boundary with no junctions at all. It forms a *closed arc*: a loop broken at an arbitrarily seeded *pseudo-node* (a degree-two breakpoint that is not a true junction). The closed arc is seeded so that it has endpoints just like any other arc.

This model is realised as a Doubly Connected Edge List (DCEL) [18], a data structure that stores each arc once and references the two regions it separates.

Simplifying a subdivision is a well-studied problem. De Berg, van Kreveld, and Schirra model the boundary network as chains (arcs) meeting at fixed vertices (nodes) and simplify each chain independently while holding the vertices in place. A topological test then rejects any simplification that would introduce a crossing, where a feature ends up on the wrong side of the boundary [4].

Simplification is still necessary; however, simplifications produced this way are not optimal. Given a per-vertex error bound, finding the simplification that retains the fewest vertices globally while staying within error tolerance is an NP-hard problem. In fact, the problem is even Min-PB-complete, thus even finding an approximation is hard unless P=NP [9].

Optimal vertex-minimal simplification is therefore not a practical target. However, it is not the property that matters here. The goal is not the fewest vertices but a boundary that stays topologically consistent.

3 Topologically Consistent Parcellation Simplification

This work introduces neither a new boundary structure, nor a new simplification scheme. Its novelty, and main contribution, lies in their application to 2D-projected cortical parcellation boundaries, the construction of a

shared-arc diagram from a labelled point cloud, and the design decisions this application forces. It achieves this by operating on a single shared representation, as fitting each region independently cannot guarantee a valid parcellation, and solving for a global optimum is intractable.

This pipeline has the following stages:

1. Bound the diagram by computing a concave hull over the entire point cloud.
2. Filter sub-threshold components from the point cloud.
3. Compute a Voronoi diagram over the filtered point cloud.
4. Extract inter-region boundaries: discard ridges between same-label cells, clip the remainder to the hull, and walk these into arcs.
5. Add hull segments along the exterior so peripheral region polygons close.
6. Assemble the arcs into a DCEL.
7. Simplify each arc (subject to constraints).
8. Reconstruct per-region polygons from the simplified arcs and serialise to a `.patches` file.

3.1 Correctness of Shared-Arcs

The pipeline represents inter-region boundaries through a DCEL. These boundaries become shared-arcs, which reference the regions they border. Modifications apply to both regions at once, meaning there are no second copies to drift apart. Nodes are pinned, meaning simplification never moves them, and regions are kept incident at the nodes they share. As a consequence of these properties, the region adjacency graph, as a combinatorial structure, is invariant under simplification. Its vertices and edges are the regions and shared-arcs the DCEL stores. Taken together, this makes adjacency a constructive property of the representation and not one that has to be recovered after the fact.

A notable case is the closed arc, which fully encloses a single region. Combinatorially, closed arcs behave just like regular arcs. They are shared between exactly two regions, and their faces and incidences remain in the DCEL regardless of what simplification does. Simplification can still substantially alter these arcs geometrically. As closed arcs carry only a single pseudo-node, the faces they bound are the least constrained in the diagram, and under aggressive simplification they could be reduced to a single point, their pseudo-node.

There is a single notion of correctness: whether the DCEL realises a valid planar subdivision. The validity of the planar subdivision consists of two layers. The *combinatorial* layer is the region adjacency graph which is held invariant by the DCEL through sharing and pinning. The *embedded* layer is the geometry the DCEL realises. As boundaries are shared between regions, gaps cannot form between them and the regions cannot overlap along a boundary. This holds for any simplification scheme, but simplification can still break the embedding. There are two ways in which it can be broken, and both are governed by how each arc is simplified rather than by the representation.

The first of these is a *crossing*: a simplified arc intersects either itself or a different arc. When crossings occur, the embedding no longer realises the stored subdivision, as adjacencies can appear that do not exist in the combinatorial structure. The second is a *collapse*: a face's simplified boundary contracts to a degenerate polygon. All arcs, faces, and adjacencies remain part of the DCEL, so while the face collapses to zero area, the stored structure is untouched. Shared boundaries alone prevent neither failure mode (Figure 3), as both are governed by how simplification modifies each arc.

Collapse is governed by the number of pinned nodes in a region's boundary. As pinning is the only constraint the representation imposes, the maximal simplification an arc can undergo is to reduce to the chord spanning its pinned nodes; the most a face can be reduced is to the polygon whose vertices are its pinned nodes in boundary order. This polygon provides a limit that all simplification tends toward, and simplification can cause collapse when that polygon is degenerate. For islands, with their pseudo-node, this means that simplification can reduce them to a single point. For two-arc lenses, with two pinned nodes, this means that they can be reduced to the chord spanning them. The limit of a face consisting of three or more arcs, and thus three or more nodes, is a positive-area polygon unless the nodes are near-collinear. The near-collinear case, where the limit polygon itself is degenerate, is left open by this argument.

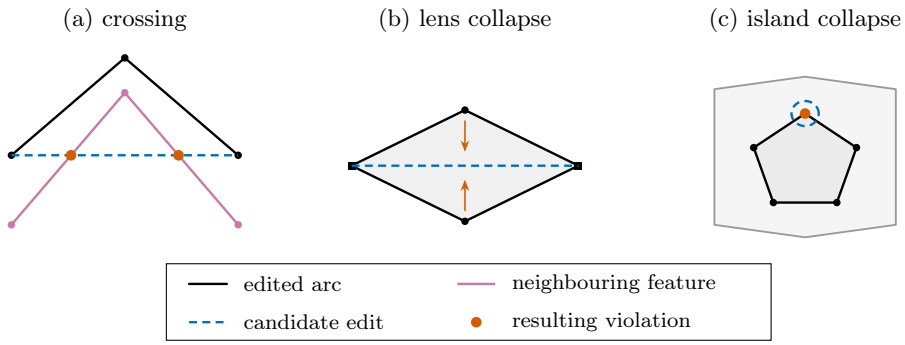

Figure 3: Failure modes a candidate simplification edit can introduce (line styles and colours in the legend). **(a)** an arc shortcuts across a neighbour; **(b)** both sides of a lens reduce to the spanning chord, collapsing its area; **(c)** an island reduces to its pseudo-node.

3.2 The Hull and Clipping

A Voronoi diagram is an unbounded partitioning of the plane. The regions on the periphery have no outer edge. To address this, the diagram is clipped to a concave hull of the point cloud. The hull supplies that outer edge. For each atlas, the hull is computed once, from the full point cloud, so that it depends only on the atlas geometry and not on what regions survive filtering (as discussed in Section 3.3). The hull shape is atlas-specific, while the parameter controlling its concavity is held constant across atlases, so that boundary construction requires no per-atlas tuning. Section 4 will show that a single value suffices for all evaluated atlases. Clipping closes each periphery region against the hull. The hull segments between consecutive points where the boundary meets the hull become the arcs separating the interior from the exterior. The exterior is the diagram's unbounded outer face, so peripheral regions close against it exactly as interior regions close against their neighbours.

3.3 Component Filtering

Before the Voronoi diagram is constructed, the input point cloud may be filtered to remove small components. The filter operates on connected components, not on labelled regions, because one region can project to several disjoint components. Individual components are tested against the threshold and removed when their size falls below a bound. Size is measured either as the number of points that make up a component or as the flatmap area a component spans. At the lowest setting no component is removed, and every component enters the diagram. Raising the bound removes progressively larger ones.

Because filtering precedes construction, it cannot break the partition. The points of a removed component are simply absent when the diagram is built. The plane they would have occupied is claimed by whichever surviving sites are now closest. That area generally passes to the enclosing region, or, for a component lying near a boundary, it is split among the several regions surrounding it. As long as at least one point survives, the Voronoi construction leaves no part of the plane unassigned, so no gap opens and no overlap is created. Filtering removes input components before any face, arc, or adjacency exists between them. Collapse reduces a face that already exists in the DCEL to a degenerate polygon. Filtering and collapse act on different objects at opposite ends of the pipeline.

That filtering is combinatorially correct does not make it a worthwhile preprocessing step. It is performed for the same reason as simplification: to reduce drawn geometry. Filtering reduces the number of primitives that have to be drawn by preventing components from ever entering the combinatorial structure. Filtering introduces its own trade-off, since by definition it removes source structure. This structure can be either genuine or an artefact of the flatmap projection. Size bounds what is filtered, but whether that is genuine structure or not cannot be resolved through size alone. Even if a region is genuine structure, it might be worth dropping if the rendering cost exceeds its representational value. The filtering threshold is therefore exposed as a tunable parameter and examined in Section 4.4.

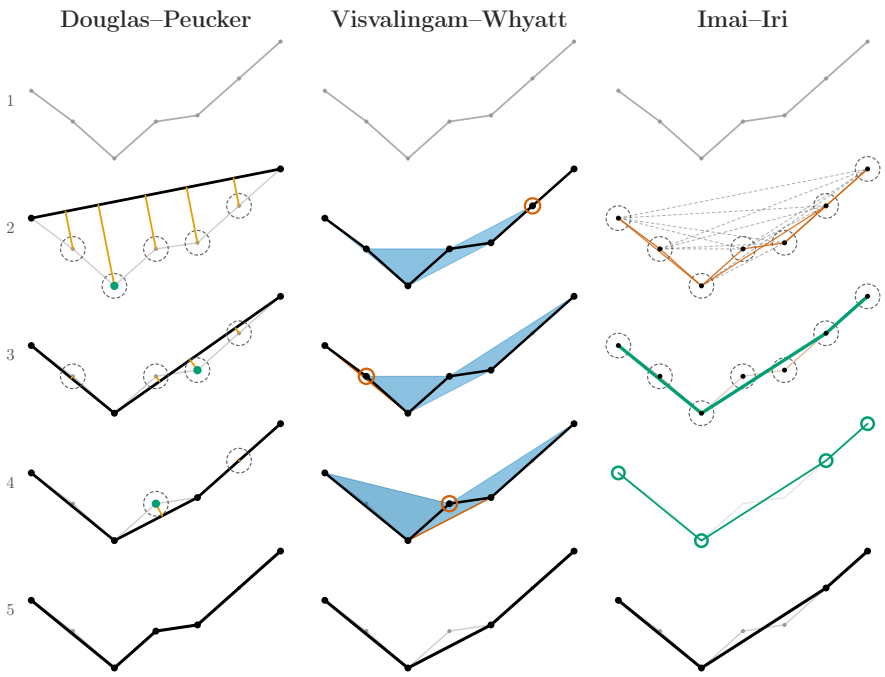

Figure 4: Three polyline simplification algorithms on a shared 7-vertex polyline (rows: original through result). Douglas–Peucker keeps the farthest point from each chord (orange); Visvalingam–Whyatt removes the lowest-area vertex (triangles shaded by area); Imai–Iri keeps the minimum-length path through the graph of ε-valid shortcuts.

3.4 Arc Simplification

The combinatorial layer of correctness is settled by the representation alone. Any simplification algorithm that holds the pinned nodes fixed and only modifies the interior vertices of an arc leaves the adjacency graph invariant. The embedded layer's failure modes have distinct causes. Whether a crossing occurs depends on each arc and its relationship to the rest of the subdivision, while whether a collapse occurs depends on the number of pinned nodes in a face. Editing arcs in isolation guarantees neither. The two baselines treat each arc individually and thus offer no guarantees, while the topology-aware variants add tests aimed at preserving the validity of the embedded layer. Figure 5 shows the evaluated algorithms and the topological guarantees they provide.

Baselines The two baselines are classical polyline simplification algorithms. The Ramer-Douglas-Peucker (RDP) algorithm [7, 20] works top-down. Given a polyline, it draws a chord between its two endpoints and finds the vertex farthest from that chord. If the distance is below a threshold ε_{RDP}, the polyline is replaced by the chord. Otherwise the vertex is kept and the algorithm recurses on the two halves it splits. Visvalingam-Whyatt (VW) [23] instead assigns each vertex the area of the triangle it makes with its two neighbours. The vertex with the smallest area is repeatedly removed until none remain below an area threshold ε_{VW}. Figure 4 illustrates the selection rule of each baseline; its third column shows the minimum-link rule that de Berg et al. build on, discussed below. Both operate on a single polyline, without any reference to its surroundings. When applied to the DCEL, they leave adjacency invariant. However, as they have no view beyond the single arc they are editing, they can offer no such guarantees about crossing prevention or region collapse.

Note that ε encodes a different scale in each algorithm: a distance threshold in RDP, an area threshold in VW. With no direct correspondence, the algorithms cannot be compared at equal parameter settings, so they are instead compared across vertex retention rates: the number of vertices surviving after simplification.

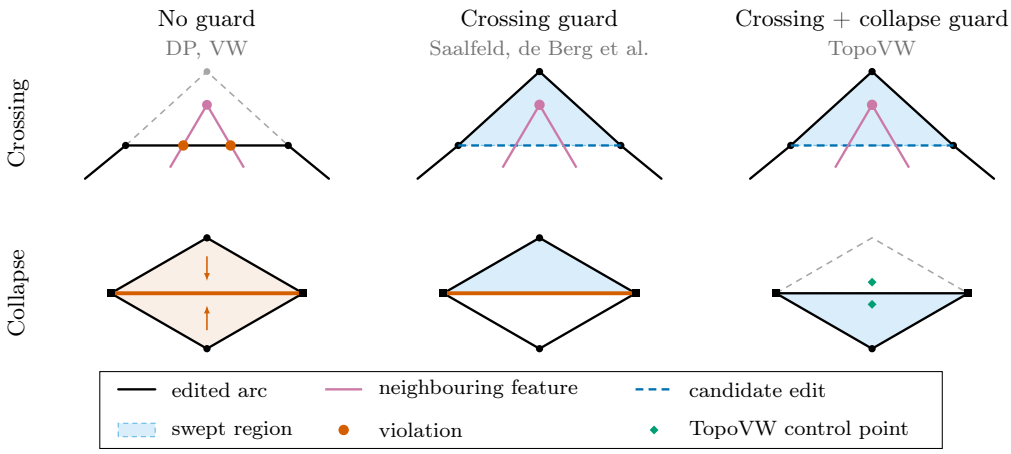

Figure 5: Accept/reject behaviour of the three guard classes on the two failure modes (cf. Figure 3); the mechanism is detailed in Section 3.4. Columns group the algorithms by guard: none (RDP, VW), crossing (Saalfeld, de Berg et al.), and crossing + collapse (TopoVW). A guard rejects an edit when a feature falls in the swept region. Every guard catches a crossing (top); only TopoVW's control points catch a collapse (bottom), which sweeps no foreign feature.

Topology Aware Simplification Algorithms The three topology-aware algorithms all extend a simplification rule with a test that consults the surrounding subdivision and rejects edits that would introduce crossings. They differ in what rule they guard and how they guard it. The algorithms all share one detection principle: crossings are caught by detecting a neighbouring feature that, after the candidate edit, would lie on the wrong side of the boundary. A collapse moves no such feature, so the test cannot catch it. A face simply contracts to the polygon defined by its nodes. No feature ever falls on the wrong side of the boundary, making the sidedness test blind to collapse. To this test an island contracting to its pseudo-node or a two-arc lens contracting to its spanning chord each looks like a valid simplification.

The first two algorithms pair naturally with the baselines. Saalfeld's algorithm [21] is RDP with a topological guard. The recursion is unchanged; a chord only replaces a sub-chain when every discarded vertex lies within ε_{RDP} of it. Replacements are accepted only if no neighbouring feature changes side relative to the chord. If such a vertex exists the chord is rejected and sub-chains are split as in plain RDP. The guard prevents an arc from sweeping across a neighbouring point. It does not offer a guard for region collapse. A closed arc contracts when all of its vertices fall within ε_{RDP} of its chord. This is only caught when its interior happens to contain a retained vertex, which is incidental, not ensured.

TopoVW [12] is a similar topology-preserving guard that extends the earlier Grid-Gen heuristic with VW's effective-area ranking. Interior vertices of all polylines are kept in a shared global priority queue ranked by their effective area, and the algorithm maintains a dynamic obstacle set of the currently active polyline vertices and control points. At each step, the globally least-significant vertex becomes the candidate for removal. The triangle the candidate spans with its neighbours is then tested against all points in the obstacle set, and the removal is rejected if any obstacle lies within it. The triangle is rejected because it is exactly the region the boundary sweeps under simplification, and accepting it would put a vertex on the wrong side of the boundary. This rejects crossings. TopoVW prevents region collapse by inserting dummy control points. They are inserted at an infinitesimal distance around one of the segments of a closed arc to prevent the island from degenerating. A second pair is added around the chord of an arc once it has simplified to it, so no second arc can collapse to it.

De Berg et al.'s algorithm [4] instantiates a topology-aware per-arc framework. Its underlying simplification is the minimum-link method of Imai and Iri (Figure 4), which finds the minimum number of vertices, for a single arc, admissible within an error tolerance, rather than removing them greedily [15]. De Berg et al.'s algorithm admits a simplified arc only when every point on the original lies within a fixed distance ε_{BKS} of it. This is done by considering all admissible shortcuts and selecting the one with the minimum number of links. Crossings are prevented by a separate consistency test which rejects any edit moving a feature to the

wrong side of the arc. The algorithm does not have a collapse guard. Collapse simply occurs when an arc's vertices fall entirely within ε_{BKS} of its spanning chord. An island wider than the bound cannot contract to its pseudo-node without leaving its far side out of tolerance and is kept. One narrower than the bound may contract until its polygon is degenerate. That same per-arc bound is blind to lenses: each of its two arcs is tested only against its own original. So, if both lie within ε_{BKS} of the chord spanning their two nodes, they may simplify to that chord and the enclosed area vanishes.

All three algorithms guard against crossings through the same type of sidedness test. They differ on which simplification principle they build on, and more importantly on how they handle region collapse. Saalfeld and de Berg et al. both leave it unguarded. It occurs when an arc's vertices all fall within their respective distance bounds. TopoVW prevents it completely. The sidedness test is what makes these algorithms topology-aware, but it only catches crossings. A collapse moves no foreign vertex relative to the boundary, so the test is blind to it. Because each algorithm has the same kind of sidedness test, they can only differ on how they handle collapse. The dividing line between them therefore is not whether they preserve topology, but to what extent. Collapse itself is also bounded: it can only occur when each arc independently reaches its chord. These guarantees form an ordering of restrictiveness. As crossing and collapse guards each constrain what edits the algorithms accept, they are expected to affect the retention levels each algorithm can reach. The baselines should reach the lowest retention, the crossing-guarded pair higher, and TopoVW the highest.

4 Experimental Setup and Results

As Section 3.4 established, the algorithms' parameters cannot be compared directly. Each is therefore run over a range of parameters, producing a population of simplified parcellations with the metrics computed on each. The parcellations considered are Destrieux [6], DK [5], DKT [16], Yeo7, Yeo17 [25] (distributed with FreeSurfer 7.4.1 [10]) and Schaefer100 (7-Network) [22] (obtained separately).

Each atlas is provided as per-vertex annotations (FreeSurfer `.annot` files) on the fsaverage subject's flattened cortical surface (`lh.cortex.patch.flat` and `rh.cortex.patch.flat`), distributed with FreeSurfer 7.4.1. The left and right hemispheres are processed independently. Their point clouds are normalised to span 100 units across each axis separately, so aspect ratio is not preserved. They are then translated into a shared frame. The left hemisphere is reflected and shifted to $x \in [-103, -3]$. The right is shifted to $x \in [3, 103]$. This gives the overall bounds of $([-103, 103], [0, 100])$. After translation each hemisphere is processed independently in its own DCEL, and merged for metric computation.

The concave hull for each hemisphere is constructed as an alpha-shape from the Delaunay triangulation. Triangles whose circumradius exceeds $1/\alpha$ are discarded and the rest are polygonised. A larger α produces a tighter, more concave hull. α is set to the largest value in $\{1.0, 1.5, 2.0, 2.5, 2.8, 3.0\}$ for which the hull remains a single connected polygon on all six atlases. This value is 2.8.

4.1 Geometric Comparison

The geometric fidelity of the simplified parcellations is compared using two metrics. First is the Intersection over Union (IoU), which is a measure of the similarity between two sets. An IoU score of 1 means that two sets are exactly the same set, while a score of 0 means that the two sets are completely disjoint. It is computed as the mean per-region IoU of the original and simplified parcellations. The second is the mean-max Hausdorff distance (mean-max-HD): for each region, the symmetric (two-sided) Hausdorff distance between its original and simplified boundary, taken as the maximum over each region's surviving faces, then averaged across regions. Because boundaries are shared, the arcs of a dropped face persist on its neighbours and their deviation is still scored. Any residual survivorship bias can only flatter scores and cannot conceal deviations larger than those already reported. Both are shown for one representative atlas (DK) in Figure 6. Appendix B and Appendix C show them for all atlases.

As vertex counts differ between atlases, IoU and Hausdorff values are compared on *vertex retention*, which is the proportion of surviving vertices after simplification relative to the original vertex count of the pooled hemispheres. This allows for comparisons between atlases. Each retention value for each algorithm-atlas combination corresponds to a single data point. Parameter values were selected manually. An initial set was refined by adding values in sparsely sampled retention ranges. The sampling is therefore non-uniform, and

reported crossovers and runaway bands are given at the resolution of sampled points. Values between two data points are linearly interpolated. The full range of parameters can be found in Appendix A. The algorithms are compared on two distinct retention ranges. The full range is the maximal range over which there is data for at least one of the algorithms. The shared support is the full pairwise overlap of the compared algorithms.

All algorithms are pushed to the minimum retention they can reach on each atlas. For algorithms with a distance-based parameter this is achieved by using an ε value of 10 000. The parcellations live in a coordinate space with bounds $([-103, 103], [0, 100])$, so by setting ε this high, simplification is able to reach every possible vertex. For VW the same value of 10 000 is used as the area threshold. Every ranked triangle is spanned by three vertices of a single hemisphere, so it lies within that hemisphere's 100×100 bounding box. No such triangle can exceed 5000 in area, so the threshold removes every possible vertex. TopoVW uses a removal-proportion-based parameter; here a parameter of 1 is used, so it only stops when all possible vertices have been removed.

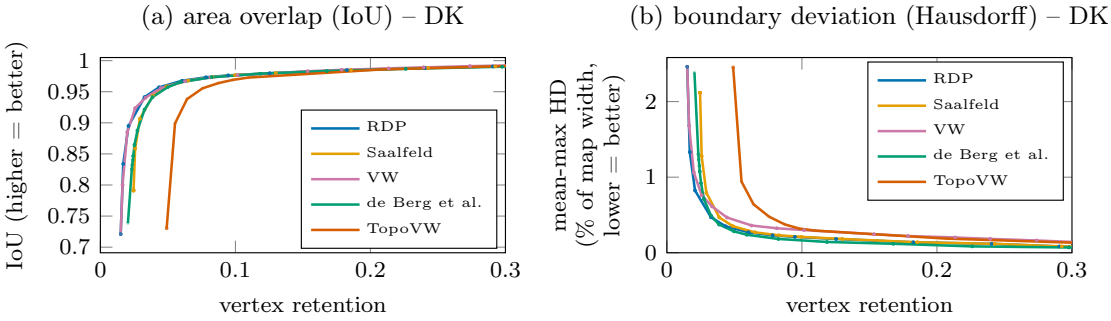

Figure 6: Two complementary fidelity metrics versus vertex retention for a representative atlas (DK), under the unfiltered setting; hemispheres combined. Faint points are each algorithm's ε-sweep measurements. Curves are connected in order of increasing retention (no fitting) and reach leftward only to each algorithm's retention floor. **(a)** Area overlap (mean IoU, higher is better), **(b)** Boundary deviation (per-region mean Hausdorff distance of surviving components as a fraction of characteristic map width, lower is better). Per-atlas curves can be found in Appendix B and Appendix C.

IoU Figure 6 (a) shows the IoU curves for the representative atlas. Four algorithms, RDP, VW, Saalfeld, and de Berg et al., agree closely across most of the range. The worst-case gap (the difference between the highest- and lowest-scoring algorithm) across these four algorithms and in all six atlases does not exceed 0.01 IoU at any retention value above approximately 0.07. In five out of six atlases this holds for all retention values above 0.05, with DKT setting the 0.07 threshold. The largest spread within this pack at retention values where the four algorithms all have data is 0.09 IoU, observed in Schaefer100 at a retention of 0.02.

TopoVW falls outside of this pack in two respects, both concentrated at the low end of its retention range. Over the upper part of its retention range, it tracks the other algorithms to within the same margin. Between retention values of 0.06 and 0.20, depending on the atlas, it breaks away from the pack. At its retention floor the gap between TopoVW and the pack mean ranges from 0.20 to 0.29 IoU.

The minimum retention each algorithm reaches is ordered monotonically by the strength of each algorithm's topological guarantees, with stronger guarantees yielding a higher floor. In all six atlases RDP and VW both reach a lower minimum retention than the other algorithms, and in all six Saalfeld and de Berg et al. both reach a lower minimum retention than TopoVW. This matches the prediction made in Section 3.4. The ratio of TopoVW's retention floor to the lowest floor on the same atlas has a median of 3.37 across atlases, and is largest in Yeo7 and Yeo17.

Mean-max-HD Figure 6 (b) shows the mean per-region Hausdorff distance computed over faces that survived simplification for the representative atlas. Hausdorff distances are reported relative to the characteristic length (map width) $L = \sqrt{A_{\text{total}}}$, where A_{total} is the total parcellation area. L is the side-length of a square of equivalent area to the parcellation, and serves as a stable scale for interpreting deviation magnitudes. The total area of each parcellation lies in the range $[11\,127, 11\,221]$, and on average the map width L is 105.6.

Above a retention of 0.10, all algorithms achieve a mean-max-HD of less than 1 % of the map width across all atlases. This bound is set by TopoVW; the other algorithms all achieve a mean-max-HD of less than 1 % for all retention values above 0.05.

De Berg et al. achieves the lowest mean rank across all atlases[1] and strictly the lowest mean-max-HD over the shared support in four out of six atlases. Only TopoVW beats it in narrow bands in Yeo7 and Yeo17 (Appendix C). VW is consistently the weakest of the four, reaching up to twice de Berg et al.'s mean-max-HD over the full range. Even in the worst case, none of these mean-max-HD scores exceeds 0.6 % of any atlas's map width.

The region mean conceals the worst-region behaviour. The per-region maximum reaches several times the mean (up to 2.15 % in DKT). On the tail, TopoVW attains a lower 95th-percentile (p95) and worst-region mean-max-HD than de Berg et al. in Yeo7 and Yeo17 despite its higher mean. On the other four atlases de Berg et al. holds the lowest p95.

4.2 Collapse and Topological Consistency

The geometric measures of Section 4.1 quantify how closely a simplified boundary follows the original, but they do not capture whether the simplified boundary remains topologically consistent. A simplified parcellation can score well on IoU even if there are topological errors such as a crossing, or if a face collapses. This section reports a second axis of fidelity that these metrics do not see: face collapse and topological consistency.

Collapse (Section 3.1) affects the simplified geometry in two distinct ways: the *number* of faces destroyed, and the *area* those faces represent. These metrics do not necessarily have to align; removing many small faces could affect the same area as removing one large face. Thus, they are reported separately.

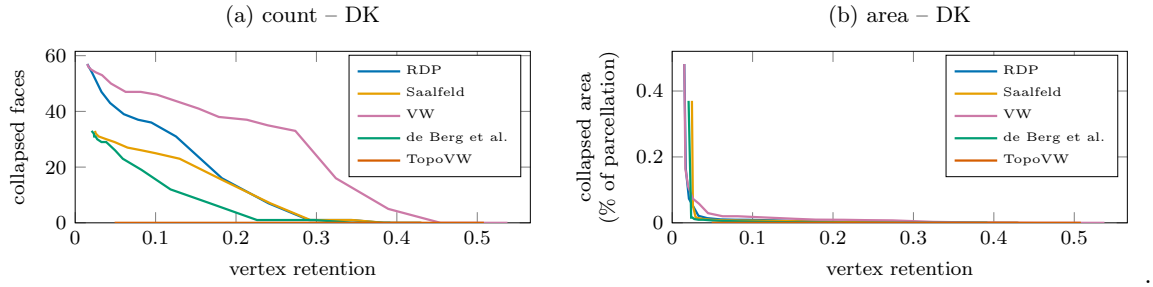

Figure 7: Two complementary collapse metrics versus vertex retention for the DK atlas (unfiltered settings; hemispheres combined). **(a)** Number of collapsed faces: a face collapse occurs when simplification reduces a polygon to a degenerate zero-area geometry. **(b)** Total area of collapsed faces as a percentage of the original parcellation area: this metric weights collapse by how much of the map is affected, not just by face count. TopoVW prevents collapse by construction and remains at zero throughout its entire retention range in both panels. Per-atlas curves for all six atlases are shown in Appendix D and Appendix E.

Collapse Figure 7 shows the number of collapsed faces and the total collapsed area as a function of vertex retention for the DK atlas. For four algorithms the number of collapsed faces rises as retention decreases. Over the whole range and across all atlases VW collapses at least as many faces as RDP. In the middle of the sweep it is strictly more. The two converge to an identical result at the lowest retention, where maximal simplification removes everything that can be removed. Saalfeld and de Berg et al. have a similar relation, with Saalfeld collapsing more faces. Unlike RDP and VW, they do not converge at their floors in DKT. The faces they collapse are on average smaller than those collapsed by RDP and VW. On Schaefer100, for example, RDP and VW collapse 52 faces totalling 37.71 units of area, while Saalfeld and de Berg et al. collapse 29 faces totalling only 1.07. Across all atlases TopoVW does not collapse any face, as guaranteed by construction.

On the area axis the picture is very different. Even at maximal simplification, the collapsed area never exceeds 3 % of the parcellation. The maximum across all algorithms and atlases is 2.98 % (RDP and VW,

[1] Mean rank is computed by sampling 5000 points along the shared support, ranking the algorithms by mean-max-HD for each, then averaging. Values between data points are linearly interpolated.

Destrieux), while for Saalfeld and de Berg et al. it remains under 1.1 % across all atlases. This holds despite them collapsing hundreds of faces in Yeo7, Yeo17, and DKT, because the collapsed faces overwhelmingly have small areas. At the floor RDP and VW collapse 407 faces in Yeo17, despite the affected area being less than 0.9 % of the total parcellation. This can clearly be seen in Figure 7, where collapse count increases steadily as vertex retention decreases while the collapsed area remains nearly flat throughout. Even in its sharply increasing tail the collapsed area never exceeds 0.5 %. This pattern holds for all atlases, where most of the increase in collapsed area clusters around the retention floor.

Topological Errors The topological behaviour the experiment records matches the guarantees established in Section 3.1 and Section 3.4 (Figure 5). As established, the combinatorial layer of the subdivision is held invariant by the construction (Section 3.1). The along-boundary gaps and overlaps this precludes are not observed on any atlas, at any retention.

Crossings occur under RDP and VW, and nowhere else. Under RDP crossings occur in 63.7 % of outputs. Under VW the rate is 43.4 %. The observed areas are small, never exceeding 0.1 % of the parcellation area. As established, collapse occurs under every algorithm except TopoVW. It only affects the realised geometry, not the stored subdivision. Collapse manifests as errors at different scales: a reduction in face count; the disappearance of regions, which occurs when all of a region's faces collapse; and the disappearance of adjacency in the realised geometry, which occurs when all faces of a region that border a different region collapse. The scale difference becomes apparent when looking at the rates of these errors. In the algorithms that allow collapse, completely losing a region occurs in 1.1 % to 12 % of outputs, while losing a face occurs in 95.1 % to 98.1 % of outputs. Most collapse is therefore the loss of individual small faces from regions that otherwise survive intact.

4.3 Rendering Cost

The rendering cost that is measured is the cost of constructing the browser-consumable representation of the parcellations. This is the direct cost the platform pays for displaying the parcellations, and the stage simplification is meant to relieve. The benchmark constructs a representative 5×5 grid of parcellations. In NBT the rows and columns correspond to frequency bands and biomarkers respectively. Dummy biomarker values are used for each parcellation, generated by a random number generator seeded at 0. The same generator was used throughout the whole benchmark. Each of the 25 grid cells shows the same parcellation; the parcellation only varies between test cases. Each test case's render time is the minimum over five repetitions; reported times are averaged over these minima.

The test cases are divided into two regimes. The simplification regime contains the same parcellations as were evaluated in the previous section, and the filter regime contains all the parcellations that will be discussed in Section 4.4.

The benchmarks were run on an AMD Ryzen 7 5800H with 16 GB of RAM. They were run single-threaded without the use of a GPU. The environment is WSL2 (Ubuntu 24.04.4 LTS) on Windows. Python 3.14.4 was used with NumPy 2.4.4, SciPy 1.17.1, Plotly 6.7.0, Pandas 3.0.2, Shapely 2.1.2, and Matplotlib 3.10.9.

The entire dataset was run on three renderers: NBT's pre-existing renderer, and two proof-of-concept renderers (`fast`, `svg`) to probe the cost model. NBT's renderer rebuilds a full figure for every grid cell: extracting geometry from the parcellation, turning it into individual scatter plots, and combining them into the final figure. `fast` and `svg` extract the geometry once and reuse it between renders, only filling in colour information. They differ in the output they produce. `fast` produces a Plotly figure like NBT's renderer, and `svg` produces a raw SVG string. The proof-of-concept renderers are not at feature parity with NBT's renderer; they serve as an indicator of the cost model and rendering headroom.

The evaluated renderers all share a common cost model: cost is dominated by the number of faces, not the number of vertices. As each face is rendered as a single output unit, the cost of producing the 25 grid cells scales with the number of faces they contain. Table 1 reports, for each renderer and regime, the R^2 of a model using both counts, the partial correlation of face count with render time, and the per-face cost. The partial correlation measures how strongly face count predicts render time after vertex count is accounted for. Partial values near 1 mean that face count explains almost all of the variation that vertex count leaves unexplained. A face-only model reaches an R^2 of 0.95 or higher in every case, and even when accounting for vertex count,

Table 1: Per-renderer rendering cost model (linear scale). R^2(both) is the joint face-and-vertex model; Partial (faces) is the fraction of otherwise-unexplained variance added by faces given vertices. Per-face cost is the face coefficient of the joint model (ms per face, holding vertices fixed), with its 95% confidence interval from the same OLS fit.

Renderer	Dataset	R^2(both)	Partial (faces)	Per-face cost (ms)	95% CI
nbt	simplification	0.96	0.95	20.0	[19.6, 20.4]
	filtering	0.97	0.89	25.3	[24.2, 26.3]
fast	simplification	0.99	0.99	6.6	[6.5, 6.7]
	filtering	0.98	0.95	8.0	[7.8, 8.2]
svg	simplification	0.97	0.96	1.6	[1.5, 1.6]
	filtering	0.99	0.97	1.6	[1.5, 1.6]

face count still explains most of the remaining variation. A vertex-only model reaches an R^2 of roughly 0.3 under simplification and roughly 0.7 under filtering. When pooling the regimes it falls to under 0.05 while a face-only model remains 0.9 or higher (Appendix F). This effect is returned to in Section 5.

Despite sharing a cost model, each renderer has markedly different overall performance. The difference between the mean rendering time of NBT's current renderer and `svg` is over an order of magnitude. On average NBT renders a grid in 5169 ms, while `svg` takes 339 ms on identical geometry. `fast` occupies an intermediate position, taking on average 1556 ms. Per-face rendering cost shows a similar difference, with NBT taking over 12 times as long as `svg` per face. These per-face costs are tightly bounded, and the NBT and svg intervals are disjoint in both regimes (Table 1), so this separation is not a sampling artifact. This shows that a significant speedup over NBT is possible while staying within the Plotly framework, and further gains are possible when leaving it. However, this benchmark cannot separate how much of the speedup comes from geometry reuse and how much from the omitted features.

Render cost is set by the number of faces, so reducing faces reduces render cost. Under simplification face reduction only happens as a side effect of collapse, which occurs incidentally rather than by choice. With collapsed faces only covering a small area (Section 4.2), removing faces directly emerges as an alternative. The next subsection examines this approach through filtering.

4.4 Filtering

Rendering cost follows the number of faces. Section 3.3 established that filtering removes faces in a controlled manner. Simplification, on the other hand, only removes faces incidentally. This section compares the two on fidelity against face count, and thus rendering cost.

For each atlas a best-achievable frontier is computed per method. For filtering this is the envelope over its parameter grid (Appendix G); for simplification this is the envelope over all five algorithms. The fidelity metrics align with those from Section 4.1. IoU and directed Hausdorff distance are computed against the unsimplified and unfiltered original. Here the Hausdorff distance is the distance from the simplified or filtered boundary to the original, which requires no correspondence between faces and so applies to filtering, where face identity is not preserved. As such, these values are not directly comparable to those of Section 4.1.

On four atlases (Destrieux, DKT, Yeo7, and Yeo17), filtering preserves more of the original geometry than simplification at matched face count. Over the range both methods reach, filtering achieves higher IoU and lower directed Hausdorff scores, and it reaches lower minimum face counts than any simplification algorithm. Figure 8 shows these plots for the DK atlas; the remaining plots appear in Appendix H and Appendix I.

On the remaining two atlases the methods do not separate. On DK, filtering and simplification interleave across the shared range on both metrics. Neither consistently holds the advantage. On Schaefer100 the evaluated range of filtering parameters populates only a narrow band of the face count, so the comparison is not resolved over the shared range. At the most aggressive setting filtering's directed Hausdorff distance exceeds simplification's, but this is a single data point. These two cases reflect the coverage of the evaluated filtering grid, rather than a property distinguishing the methods.

Where the comparison is resolved, filtering is at least as faithful as simplification at matched rendering

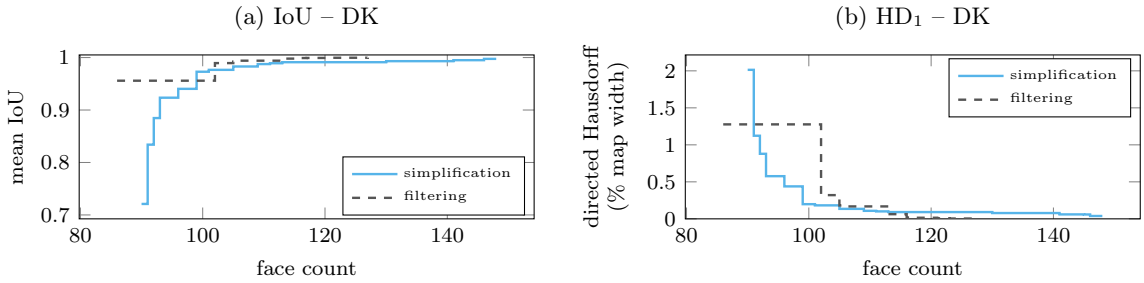

Figure 8: Best-achieved-value front for simplification (best over all algorithms, solid) and filtering (dashed) for the DK atlas. **(a)** Mean IoU versus face count (higher is better): at each face count the curve shows the maximum IoU achieved by any configuration using at most that many faces. **(b)** Directed Hausdorff HD$_1$ (simplified $\rightarrow$ original boundary) versus face count (lower is better), expressed as % of characteristic map width $\sqrt{A_{\text{total}}}$. Per-atlas grids for all six atlases are shown in Appendix H and Appendix I.

cost. On most atlases it is more faithful. A parameter grid dense enough to resolve every atlas is left to future work.

5 Discussion

Read together, the results in Section 4 show that the axis separating the evaluated algorithms is not how faithfully they simplify. RDP, VW, Saalfeld, and de Berg et al. agree to within 0.01 IoU over most of the range. Even on the metric where the four do diverge, mean-max-HD, they agree to within 0.6 % of the map width. What does separate these algorithms is the number of collapsed faces they allow, which directly governs rendering cost.

Collapse is a benign failure mode. Even at maximal simplification the total collapsed area never exceeds 3 % of the total parcellation area, and remains under 1.1 % for the crossing-guarded algorithms. Yet the faces that constitute this small area are numerous and collapsing them disproportionately relieves cost compared to the area given up. This asymmetry makes collapse an effective cost lever.

The evaluated algorithms do not optimise for this cost lever. On all three evaluated renderers, cost tracks faces, not vertices. Pooled across regimes a vertex-only model loses nearly all of its predictive power while a face-only model does not (Section 4.3). This does not make vertex count inert. Within each regime it proxies face count and retains moderate power; the two regimes' vertex-cost relationships simply do not align, so pooling cancels the signal. Each evaluated algorithm removes vertices by its own rule and none of them reference face count, which only moves incidentally, via collapse.

The same guards that prevent the algorithms from simplifying certain vertices prevent them from collapsing certain faces through different means. TopoVW's collapse guard is a direct answer, while Saalfeld's and de Berg et al.'s crossing guards only prevent it incidentally. Faces are what rendering cost tracks; thus the ordering of how much cost relief these algorithms can deliver runs opposite to the topological guarantees they provide. These algorithms are optimised for vertex reduction under topological constraints, which is a different objective from the one the cost model rewards. The relief does not come for free: the algorithms that reach furthest do so by rejecting the fewest edits, and they are the only ones that allow crossings.

Filtering exposes a more direct method to pull on the cost lever, discarding the small components whose cost is disproportionate to the area they cover. It is the cost lever pulled on purpose. On four out of six atlases, at matched face count, and thus rendering cost, it retains more of the original parcellation than simplification. On DK and Schaefer100, the evaluated filtering grid is too sparse to resolve the comparison. The methods are also never composed, only set against each other, so which of the two is better is not something these results fully settle.

The largest speedup comes from neither simplification nor filtering. Just by switching the renderer an average speedup of 15 times is achievable, far more than any simplification can provide by itself. This does not place the two in competition. Face count and fidelity are properties of the parcellation; the renderer only draws it. On the established face-dominated cost model the renderer sets the cost per face while the parcellation sets

how many faces that cost is paid on. Of the evaluated renderers, the faster renderers rescale the budget but do not change the cost model's structure.

These cost results are bounded in two ways. They only measure the construction of the displayable representation, and the evaluated renderers, apart from NBT's own, are proof-of-concept rather than production-grade. Face count dominates rendering cost, but whether this holds for a production-grade renderer or over the full path from construction to display is untested. Vertex count is a dominated proxy, not an inert one, and a model that restores its influence, such as one including serialisation, would sit alongside these results rather than against them. Where this leaves the boundary between face- and vertex-driven cost is an open question raised by these results.

6 Conclusions and Future Work

This work asks how simplification affects geometric fidelity, topological consistency, and rendering cost, on a gap- and overlap-free representation of cortical parcellations. Of the three axes, fidelity barely separates the algorithms. Topological consistency, on the other hand, does: in a face-dominated renderer, the strength of an algorithm's topological guarantees is inversely related to the cost relief it can deliver.

Geometric fidelity (RQ1). The governing axis behind fidelity is how far simplification is pushed, not what algorithm is used. Across all atlases four out of the five evaluated algorithms agree to within 0.01 IoU over most of the range, and at higher retention values all five agree. The algorithms are thus near-indistinguishable on fidelity.

Topological consistency (RQ2). The shared-arc model and node pinning make gaps and overlaps along shared boundaries impossible by construction. The two remaining failure modes, crossings and collapse, are governed by how the algorithm simplifies. Crossings only occur in the unguarded baselines, RDP and VW. Collapse is allowed by all algorithms except TopoVW, and is near-universal in them, though it stays benign, never exceeding 3 % of the parcellation area. For crossing-guarded algorithms the collapsed area stays under 1.1 %.

Rendering cost (RQ3). Across all three evaluated renderers, face count governs rendering cost, with vertex count being a dominated proxy. The evaluated algorithms optimise for vertex count, not face count. They only reduce face count incidentally, through collapse. The algorithms that provide the strongest topological guarantees collapse the fewest faces. Filtering, which removes faces directly, is at least as faithful as simplification wherever there is enough data to resolve the comparison.

Taken together, the shared-arc model prevents the representation from opening gaps and overlaps at any simplification level. The choice of simplification algorithm then dictates what topological failures are admissible. Simplification itself carries a small, bounded fidelity cost.

In practical terms, the largest speedup available is renderer-side, not geometry-side: switching the renderer relieves cost several-fold, far more than any simplification is able to reach.

Contribution. This work's main contribution is the application of subdivision simplification in the context of 2D-projected cortical parcellations, the construction of a shared-arc diagram from a labelled point cloud, and the evaluation across five simplification algorithms (RDP, VW, Saalfeld, de Berg et al., and TopoVW) and six atlases (Destrieux, DK, DKT, Schaefer100, Yeo7, and Yeo17).

This work leaves several open questions. The fidelity reported here is purely geometric. It is measured against a source boundary that is itself an approximation. Whether this translates to anatomical fidelity is left open. Quantifying it would require either an outside anatomical reference or a downstream task where the effect of simplification could be measured.

NBT serves as motivating context; the fidelity and topology findings do not depend on it. What remains is NBT-specific: what level of simplification is appropriate under NBT's fidelity and cost constraints. Further, NBT's end-to-end cost model has not been characterised. The face-dominant cost model only measures the construction cost of the displayable representation. Stages like network serialisation could plausibly restore vertex count to a dominant cost variable.

The characterisation of the cost model leaves two questions open. Filtering alone needs a denser parameter grid to settle the ambiguities left by DK and Schaefer100. The combination of filtering and simplification changes the base parcellation that simplification acts on, so the resulting behaviour cannot be predicted from measuring either alone.

7 Responsible Research

7.1 Ethical Considerations

This work operates on cortical parcellations derived from publicly available brain atlases, and on two-dimensional flatmap point clouds projected from them. Individual EEG recordings and other personally identifiable information are not handled by this work; the atlases used are standardised anatomical templates. The direct ethical stakes of this work are therefore limited. It introduces no privacy or consent surface beyond that of the public templates it builds on. The one consideration that does warrant reflection is representational honesty.

Cortical regions do not have exact physical borders; a parcellation imposes boundaries according to anatomical or functional criteria that differ between atlases and are themselves models, with no single correct boundary. The atlases used in this work are annotations on top of FreeSurfer's fsaverage, a cortical surface template averaged over multiple subjects [11]. The boundaries drawn are approximations stacked several layers deep. The flatmap is a sampled two-dimensional projection of a three-dimensional surface. The Voronoi construction recovers inter-region boundaries as the one-nearest-neighbour partition of the sampled points rather than using the atlas boundaries themselves. Boundary simplification then adds another, deliberate, approximation. As none of the layers beneath it are exact either, the risk is not that simplification departs from a true boundary, but that the simplified parcellation is taken as anatomical ground truth.

What this work can and does take responsibility for is the layer it introduces. Full atlas-vertex resolution plausibly exceeds the spatial detail EEG can resolve [2, 17], suggesting that simplification discards geometric detail that the signal cannot support. This work, however, does not identify that limit or tune simplification to it, and makes no claim to have matched it. This work can offer two guarantees. First, when using the topology-aware algorithms, simplification cannot fabricate structure. The topological guarantees established in Section 3.1 make it impossible to open gaps that leave cortex unattributed, to create overlap that assigns cortex to multiple regions, or to introduce adjacencies not present in the source data. Only boundaries that are already in the Voronoi diagram are smoothed. The one failure mode this does not reach is region collapse, which destroys rather than fabricates structure, and is addressed by choice of algorithm rather than through this guarantee (Section 3.4). Second, its fidelity cost is measured across a range of settings, so that the deviation simplification adds is known and bounded, even though the baseline it is measured against is itself an approximation.

7.2 Reproducibility

Five of the evaluated atlases are distributed with FreeSurfer 7.4.1 (Destrieux, DK, DKT, Yeo7, Yeo17) and pinned by that version. The Schaefer100 (7-Network) parcellation [22] was obtained separately from the CBIG repository[2] as the files `{lh,rh}.Schaefer2018_100Parcels_7Networks_order.annot` in `.../Schaefer2018_LocalGlobal/Parcellations/FreeSurfer5.3/fsaverage/label/`. These are applicable to FreeSurfer 7.4.1 because `fsaverage`'s vertex indexing is fixed across FreeSurfer versions. All inputs are standardised public templates; replicators must obtain them themselves.

The full source code, including the scripts that produce the underlying data for every figure and table, will be made publicly available at `https://github.com/fvdveen/parcellation_boundaries` on completion of the project.

The stochastic elements have been discussed in Section 4.3 and only affect the colour of the rendered geometry; with no colour-specific path enabled in any renderer they have no impact on rendering time.

Every atlas-algorithm-parameter combination provided bit-identical output across repeated runs; no run-to-run variation was observed at any stage. The simplification stage is fully deterministic given a fixed toolchain; its priority queues impose a total ordering on vertex removal, and ties are broken using first-index selection. The one reproducibility risk lies before simplification: the construction of the shared-arc diagram can depend on library-internal tie resolution. This was stable across runs but is resolved by the library rather than through an explicit rule; pinning it is left to the released implementation.

The cost section reports wall-clock times on a single fixed machine (Appendix F) and will not be reproducible in absolute terms. Per-face costs and mean rendering time all depend on hardware, operating system,

[2] `https://github.com/ThomasYeoLab/CBIG`, commit `d1454a611f7de10a3b36665e6fbb3fb6c770d140`.

and load. The shape of the cost model should remain invariant across machines as this rests on counts and relative slopes and not absolute timings.

7.3 Integrity

All claims made in this work are bounded to the population they measure. The algorithms are on equal footing; the evidence does not support declaring one the single best, so none is. Likewise the cost model is reported for the three evaluated renderers only. Two of these are proof-of-concept renderers not at feature parity with NBT, and are used to probe the cost model's structure. They show the headroom available in NBT's renderer, but the speedup they show is not authoritative, since they omit features that NBT's renderer implements.

On DK and Schaefer100 the simplification-filtering comparison cannot be fully resolved, in both cases because the evaluated filtering grid does not cover the comparison range densely enough. On DK the two frontiers stay close where they overlap, and the filtering grid is sampled sparsely across that range, so no consistent lead can be established. On Schaefer100 the filtering data points fall within a narrow band of face counts, leaving too little shared range to compare over; both coarser and finer settings would need evaluating. In both cases the comparison is reported as open rather than forcing a verdict, with the denser grid left to future work.

The mean-max-HD metric is computed over only the faces surviving simplification and does not see collapse. As Section 4.1 established, any resulting bias can only flatter the reported scores, and not conceal a deviation larger than already reported.

7.4 Use of AI Tools

Generative AI tools were used in two capacities during the preparation of this thesis. Claude Opus 4.8 assisted with writing, helping to refine phrasing, structure, clarity, grammar, and spelling. For programming Claude Code was used to generate boilerplate code, improve readability, and draft portions of the analysis scripts and notebooks used to calculate the metrics. All code was reviewed, tested, and verified by the author. The results were independently verified and understood by the author. The author further used these tools to brainstorm and test ideas. All resulting decisions, findings, and interpretations are the author's own work, and they take full responsibility for the content of this thesis.

References

[1] Franz Aurenhammer. Voronoi diagrams–a survey of a fundamental geometric data structure. *ACM computing surveys (CSUR)*, 23(3):345–405, 1991.

[2] Boris Burle, Laure Spieser, Clémence Roger, Laurence Casini, Thierry Hasbroucq, and Franck Vidal. Spatial and temporal resolutions of eeg: Is it really black and white? a scalp current density view. *International Journal of Psychophysiology*, 97(3):210–220, 2015.

[3] Mark De Berg. *Computational geometry: algorithms and applications*. Springer Science & Business Media, 2000.

[4] Mark de Berg, Marc van Kreveld, and Stefan Schirra. Topologically correct subdivision simplification using the bandwidth criterion. *Cartography and Geographic Information Systems*, 25(4):243–257, 1998.

[5] Rahul S Desikan, Florent Ségonne, Bruce Fischl, Brian T Quinn, Bradford C Dickerson, Deborah Blacker, Randy L Buckner, Anders M Dale, R Paul Maguire, Bradley T Hyman, et al. An automated labeling system for subdividing the human cerebral cortex on mri scans into gyral based regions of interest. *Neuroimage*, 31(3):968–980, 2006.

[6] Christophe Destrieux, Bruce Fischl, Anders Dale, and Eric Halgren. Automatic parcellation of human cortical gyri and sulci using standard anatomical nomenclature. *Neuroimage*, 53(1):1–15, 2010.

[7] David H Douglas and Thomas K Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: the international journal for geographic information and geovisualization*, 10(2):112–122, 1973.

[8] Herbert Edelsbrunner, David Kirkpatrick, and Raimund Seidel. On the shape of a set of points in the plane. *IEEE Transactions on information theory*, 29(4):551–559, 1983.

[9] Regina Estkowski and Joseph SB Mitchell. Simplifying a polygonal subdivision while keeping it simple. In *Proceedings of the seventeenth annual symposium on Computational geometry*, pages 40–49, 2001.

[10] Bruce Fischl. Freesurfer. *Neuroimage*, 62(2):774–781, 2012.

[11] Bruce Fischl, Martin I Sereno, Roger BH Tootell, and Anders M Dale. High-resolution intersubject averaging and a coordinate system for the cortical surface. *Human brain mapping*, 8(4):272–284, 1999.

[12] Mauricio G Gruppi, Salles VG Magalhães, Marcus Vinícius Alvim Andrade, W Randolph Franklin, and Wenli Li. An efficient and topologically correct map generalization heuristic. In *ICEIS (1)*, pages 516–525, 2015.

[13] Xuejun Hao, Dongrong Xu, Ravi Bansal, Jun Liu, and Bradley S Peterson. An improved representation of regional boundaries on parcellated morphological surfaces. *Computerized Medical Imaging and Graphics*, 35(3):206–219, 2011.

[14] Richard Hardstone, Simon-Shlomo Poil, Giuseppina Schiavone, Rick Jansen, Vadim V Nikulin, Huibert D Mansvelder, and Klaus Linkenkaer-Hansen. Detrended fluctuation analysis: a scale-free view on neuronal oscillations. *Frontiers in physiology*, 3:23105, 2012.

[15] Hiroshi Imai and Masao Iri. Computational-geometric methods for polygonal approximations of a curve. *Computer Vision, Graphics, and Image Processing*, 36(1):31–41, 1986.

[16] Arno Klein and Jason Tourville. 101 labeled brain images and a consistent human cortical labeling protocol. *Frontiers in neuroscience*, 6:33392, 2012.

[17] Christoph M Michel and Denis Brunet. Eeg source imaging: a practical review of the analysis steps. *Frontiers in neurology*, 10:325, 2019.

[18] David E. Muller and Franco P. Preparata. Finding the intersection of two convex polyhedra. *Theoretical Computer Science*, 7(2):217–236, 1978.

[19] Faisal Mushtaq, Dominik Welke, Anne Gallagher, Yuri G Pavlov, Layla Kouara, Jorge Bosch-Bayard, Jasper JF Van Den Bosch, Mahnaz Arvaneh, Amy R Bland, Maximilien Chaumon, et al. One hundred years of eeg for brain and behaviour research. *Nature human behaviour*, 8(8):1437–1443, 2024.

[20] Urs Ramer. An iterative procedure for the polygonal approximation of plane curves. *Computer graphics and image processing*, 1(3):244–256, 1972.

[21] Alan Saalfeld. Topologically consistent line simplification with the douglas-peucker algorithm. *Cartography and Geographic Information Science*, 26(1):7–18, 1999.

[22] Alexander Schaefer, Ru Kong, Evan M Gordon, Timothy O Laumann, Xi-Nian Zuo, Avram J Holmes, Simon B Eickhoff, and BT Thomas Yeo. Local-global parcellation of the human cerebral cortex from intrinsic functional connectivity mri. *Cerebral cortex*, 28(9):3095–3114, 2018.

[23] M Visvalingam and JD Whyatt. Line generalisation by repeated elimination of points. *The Cartographic Journal*, 30(1):46–51, 1993.

[24] Robin Weiler, Marina Diachenko, Erika L Juarez-Martinez, Arthur-Ervin Avramiea, Peter Bloem, and Klaus Linkenkaer-Hansen. Robin's viewer: Using deep-learning predictions to assist eeg annotation. *Frontiers in Neuroinformatics*, 16:1025847, 2023.

[25] BT Thomas Yeo, Fenna M Krienen, Jorge Sepulcre, Mert R Sabuncu, Danial Lashkari, Marisa Hollinshead, Joshua L Roffman, Jordan W Smoller, Lilla Zöllei, Jonathan R Polimeni, et al. The organization of the human cerebral cortex estimated by intrinsic functional connectivity. *Journal of neurophysiology*, 2011.

A Evaluated Simplification Parameters

Table 2: Simplification algorithm parameters used during evaluation

Algorithm	**Parameters**
RDP	0.05, 0.06, 0.07, 0.085, 0.1, 0.12, 0.15, 0.175, 0.2, 0.25, 0.35, 0.5, 1.0, 2.0, 16.0, 10000
VW	0.005, 0.007, 0.01, 0.015, 0.02, 0.025, 0.03, 0.04, 0.05, 0.1, 0.15, 0.25, 0.5, 1.0, 2.0, 5.0, 16.0, 10000
Saalfeld	0.05, 0.06, 0.07, 0.085, 0.1, 0.12, 0.15, 0.175, 0.2, 0.25, 0.35, 0.5, 1.0, 2.0, 16.0, 18.0, 20.0, 22.0, 24.0, 10000
de Berg et al.	0.05, 0.06, 0.07, 0.085, 0.1, 0.12, 0.15, 0.2, 0.25, 0.35, 0.5, 0.75, 1.0, 1.2, 1.3, 1.4, 1.5, 10000
TopoVW	0.5, 0.6, 0.7, 0.8, 0.9, 0.9125, 0.925, 0.9375, 0.95, 0.96, 0.975, 0.9775, 0.98, 0.9875, 1

B IoU Against Vertex Retention

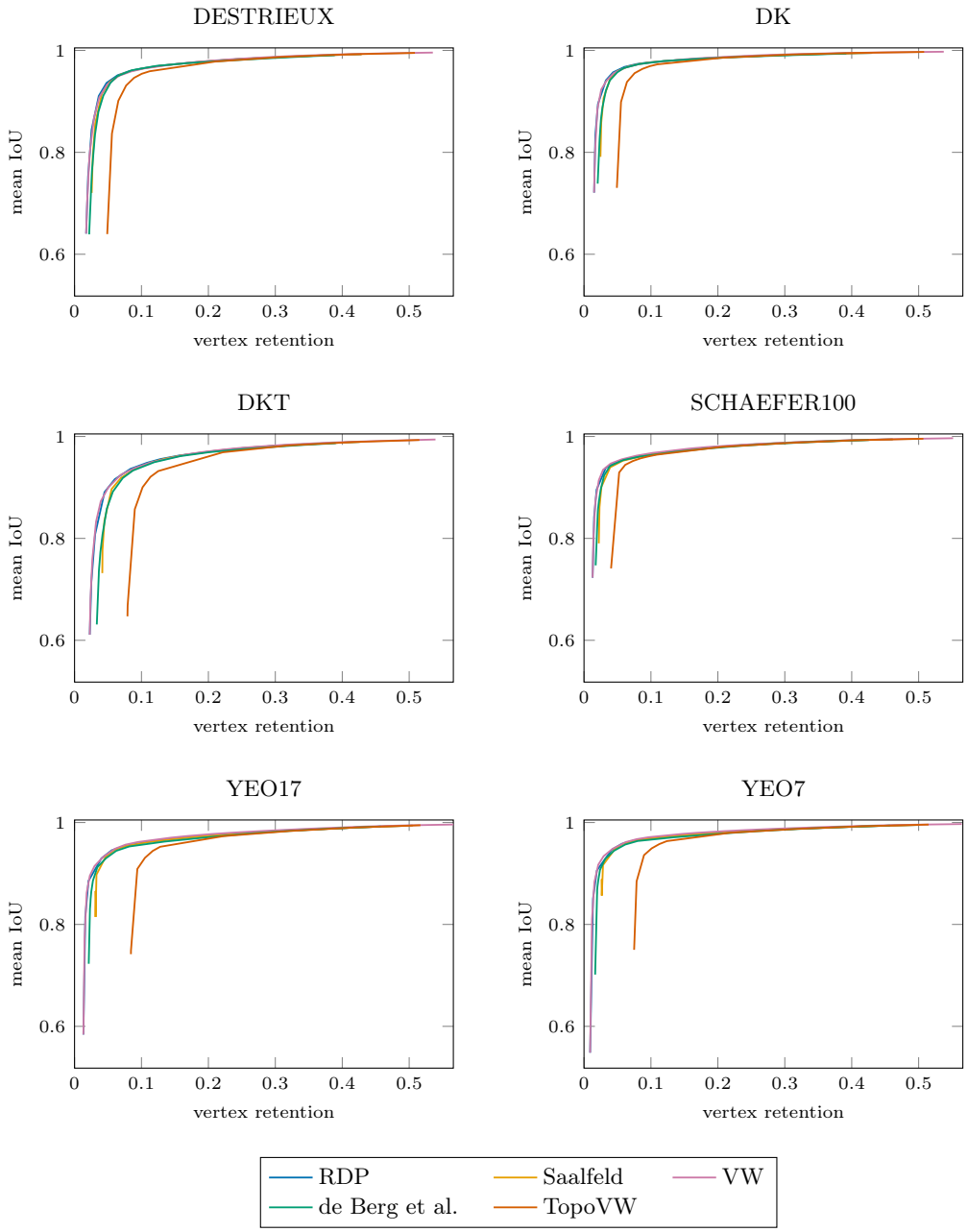

Figure 9: Mean IoU (higher is better) versus vertex retention for all six atlases (unfiltered setting; hemispheres combined). Each curve extends leftward to the lowest vertex retention attained for that algorithm within the swept parameters; the algorithms reach different minima, TopoVW the highest. Supports Figure 6.

C Mean Per-Component Hausdorff Against Vertex Retention

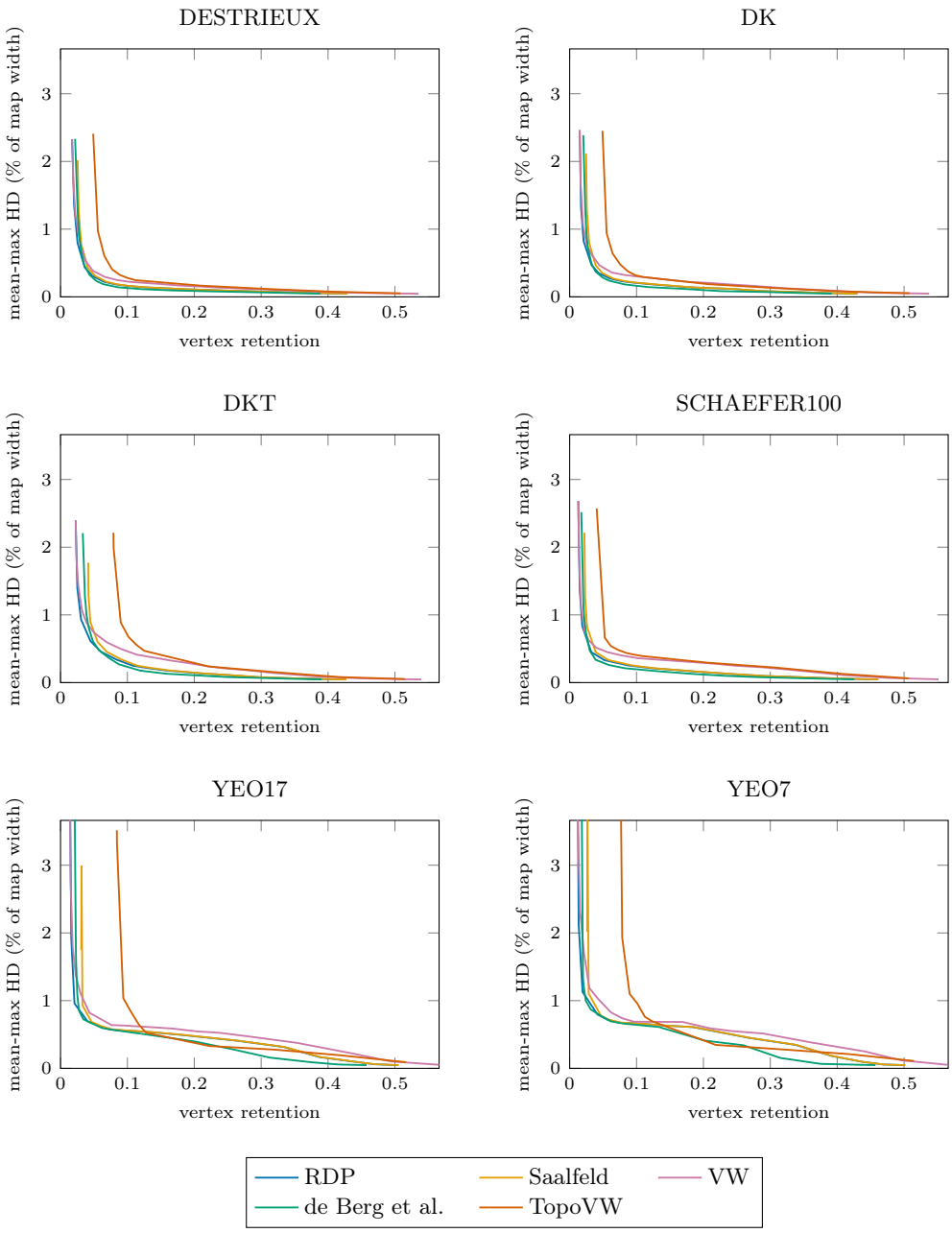

Figure 10: Per-component mean of Hausdorff distance versus vertex retention, as a fraction of characteristic map width, for all six atlases (lower is better, unfiltered setting; hemispheres combined). Each curve extends leftward to the lowest vertex retention attained for that algorithm within the swept parameters; the algorithms reach different minima, TopoVW the highest. Supports Figure 6.

D Collapsed Face Count Against Vertex Retention

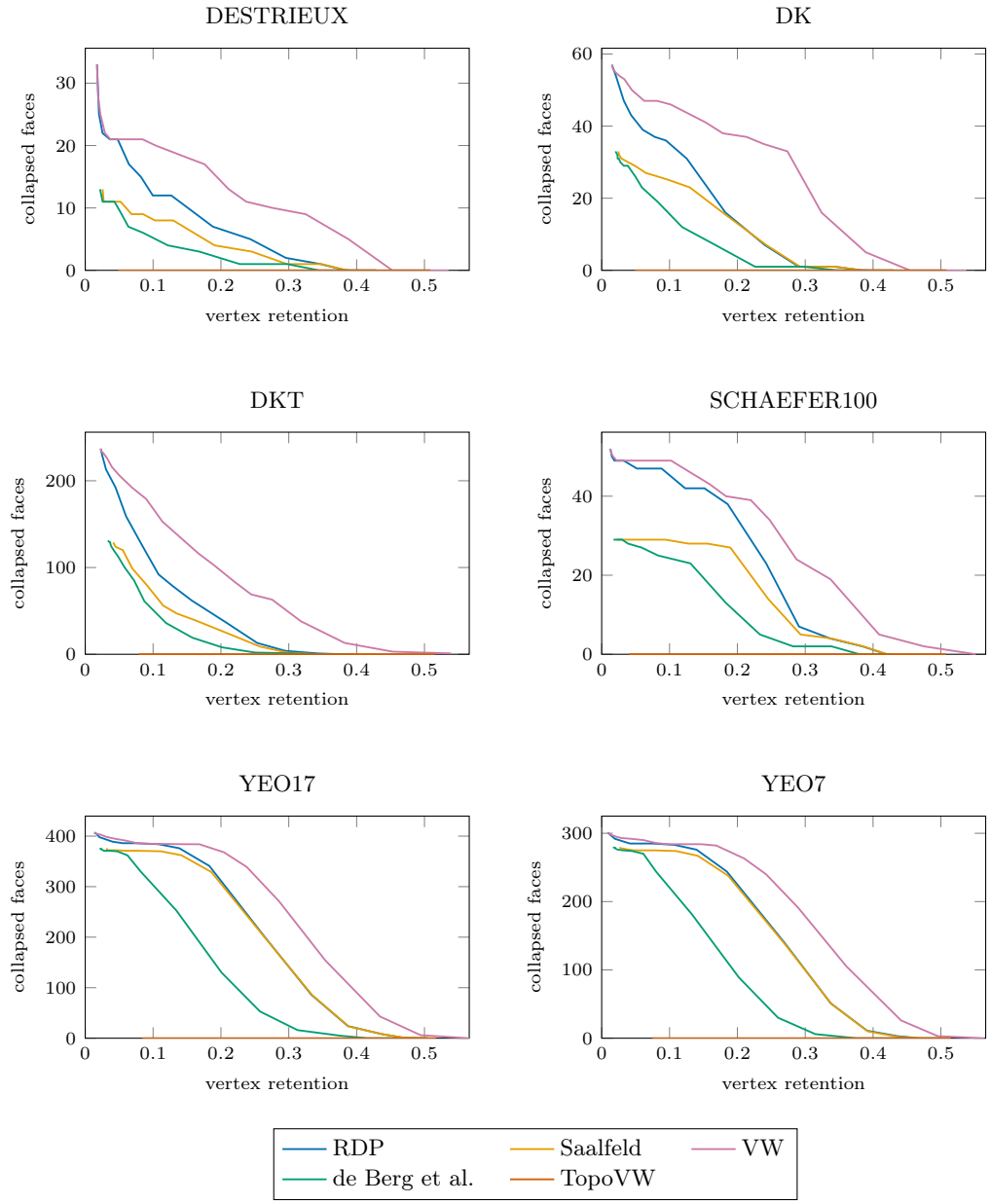

Figure 11: Number of collapsed faces versus vertex retention for all six atlases (unfiltered settings; hemispheres combined). TopoVW is always zero; the reference algorithms accumulate collapsed faces under aggressive simplification. y-axis scales differ across atlases. Supports Figure 7.

E Collapsed Area Against Vertex Retention

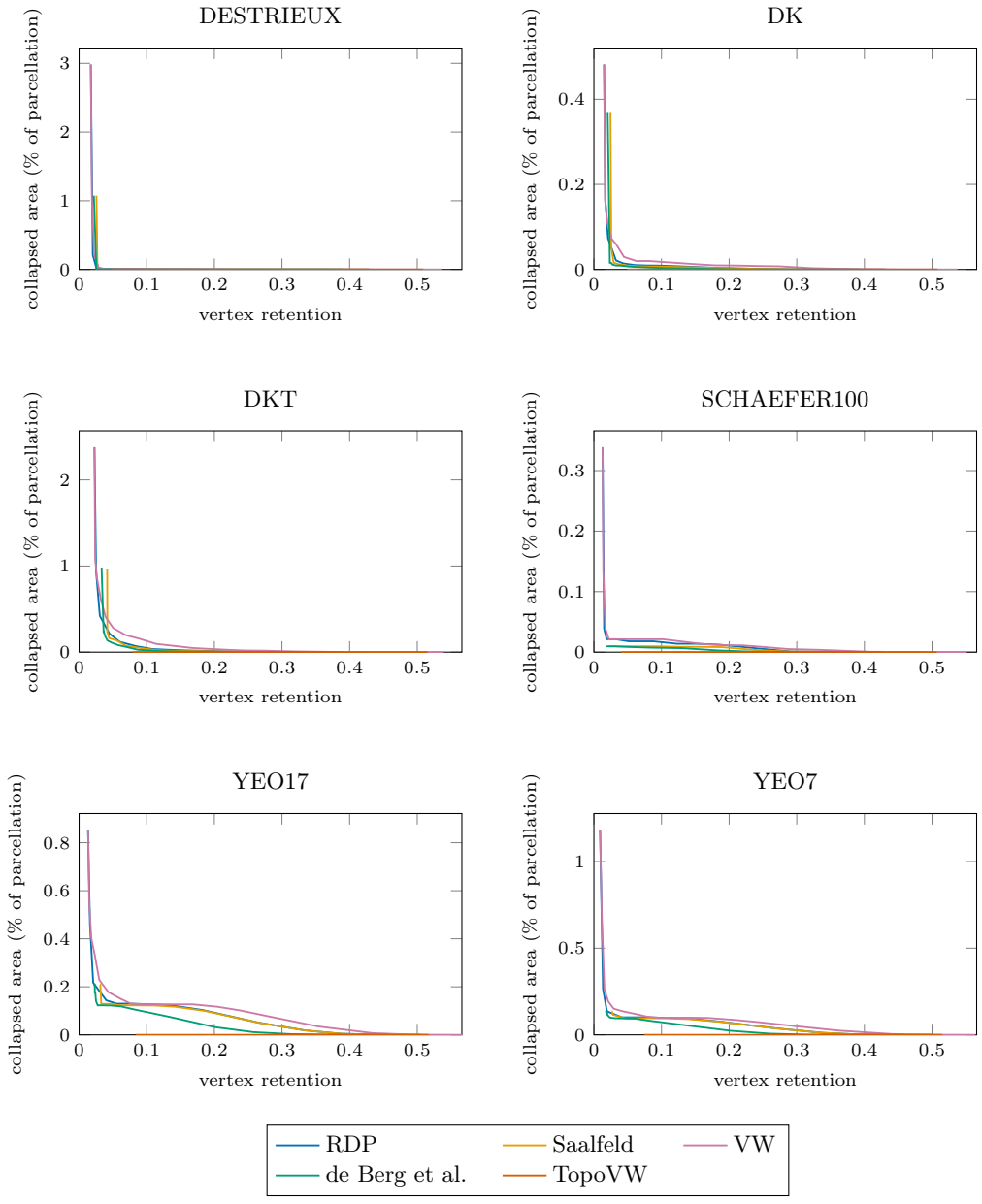

Figure 12: Total area of collapsed faces as a percentage of the original parcellation area versus vertex retention for all six atlases (unfiltered settings; hemispheres combined). y-axis scales differ across atlases. TopoVW is always zero. Supports Figure 7.

F Per-Renderer Cost-Model Regression

Table 3: Full per-renderer cost-model regressions (linear scale), including the pooled (combined) regime. R^2 columns are for the vertices-only, faces-only and joint models; the partial columns give the fraction of otherwise-unexplained variance added by faces given vertices ($f \mid v$) and vertices given faces ($v \mid f$). Face-only slope is the face coefficient of the single-predictor model; joint coef. is the face coefficient after adding vertices. Brackets are 95% confidence intervals on each face coefficient, from the same OLS fits. n is the number of test cases (parcellations).

Renderer	Regime	n	R^2(v)	R^2(f)	R^2(both)	Partial ($f \mid v$)	Partial ($v \mid f$)	Face-only slope	95% CI	Joint coef.	95% CI
nbt	simplification	528	0.31	0.96	0.96	0.95	0.13	20.952	[20.579, 21.324]	20.020	[19.617, 20.422]
	filtering	270	0.71	0.96	0.97	0.89	0.19	28.630	[27.946, 29.314]	25.254	[24.211, 26.297]
	combined	798	0.04	0.90	0.96	0.96	0.61	20.221	[19.753, 20.689]	20.441	[20.148, 20.735]
fast	simplification	528	0.31	0.99	0.99	0.99	0.26	6.835	[6.767, 6.903]	6.594	[6.526, 6.662]
	filtering	270	0.68	0.98	0.98	0.95	0.10	8.511	[8.374, 8.648]	8.021	[7.800, 8.241]
	combined	798	0.03	0.94	0.99	0.99	0.80	6.531	[6.412, 6.650]	6.596	[6.542, 6.649]
svg	simplification	528	0.28	0.97	0.97	0.96	0.04	1.598	[1.575, 1.621]	1.564	[1.538, 1.591]
	filtering	270	0.66	0.99	0.99	0.97	0.02	1.593	[1.573, 1.614]	1.560	[1.525, 1.594]
	combined	798	0.00	0.98	0.98	0.98	0.00	1.607	[1.590, 1.625]	1.607	[1.590, 1.624]

G Evaluated Filter Parameter

Table 4: Filtering parameter grid. The filtering frontier is the envelope over all $9 \times 5 = 45$ combinations.

Parameter	Values
Minimum face area	0, 0.25, 0.5, 1, 2, 5, 10, 20 and 50
Minimum points per face	1, 2, 3, 5 and 10

H Best Achieved IoU for Simplification and Filtering

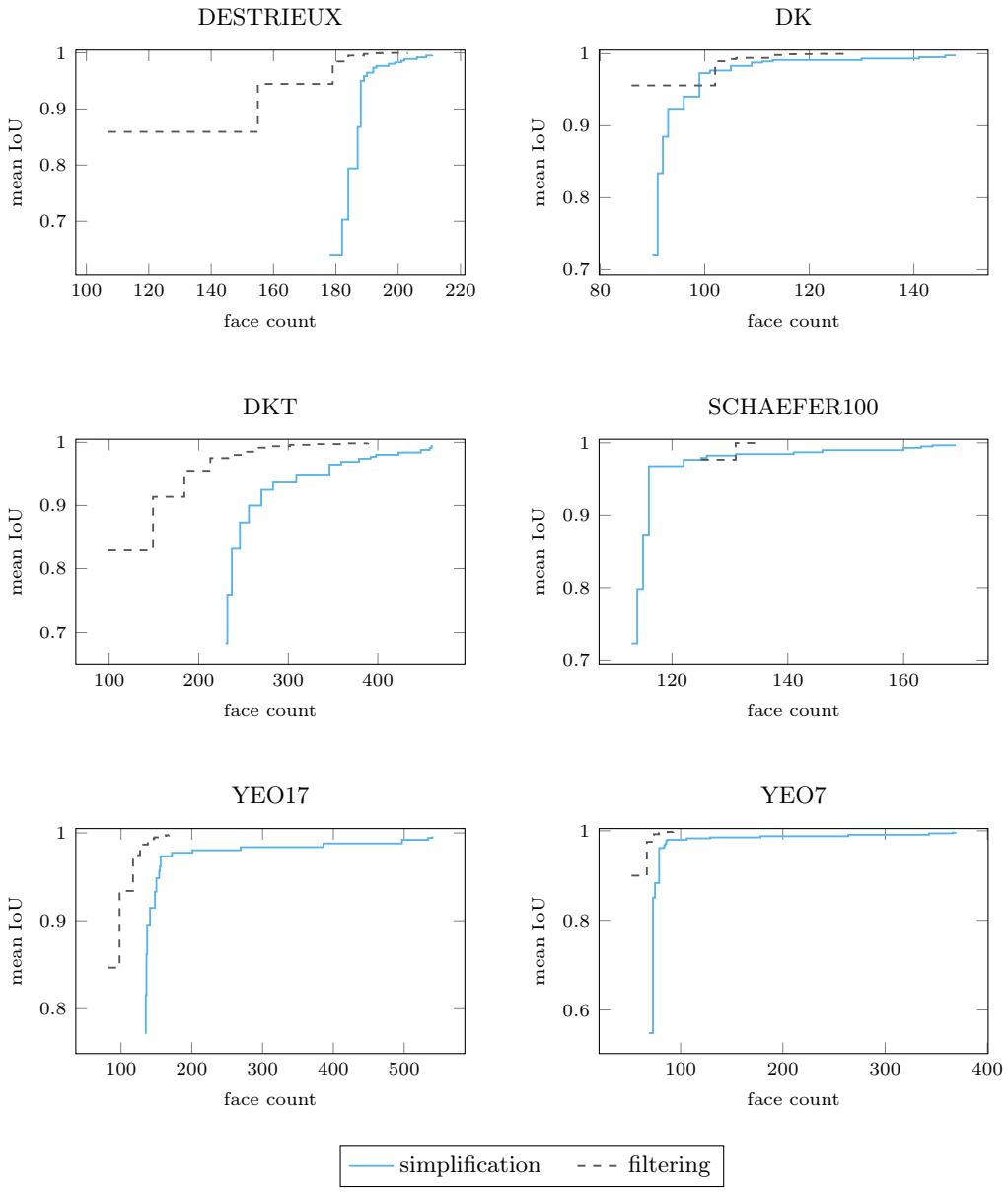

Figure 13: Best-achieved-value front (mean IoU vs face count) for all six atlases. Simplification algorithms (solid) and filtering (dashed, gray). y-axis upper limit is 1; lower limit scales to the data. Supports Figure 8.

I Best Achieved One-Sided Hausdorff for Simplification and Filtering

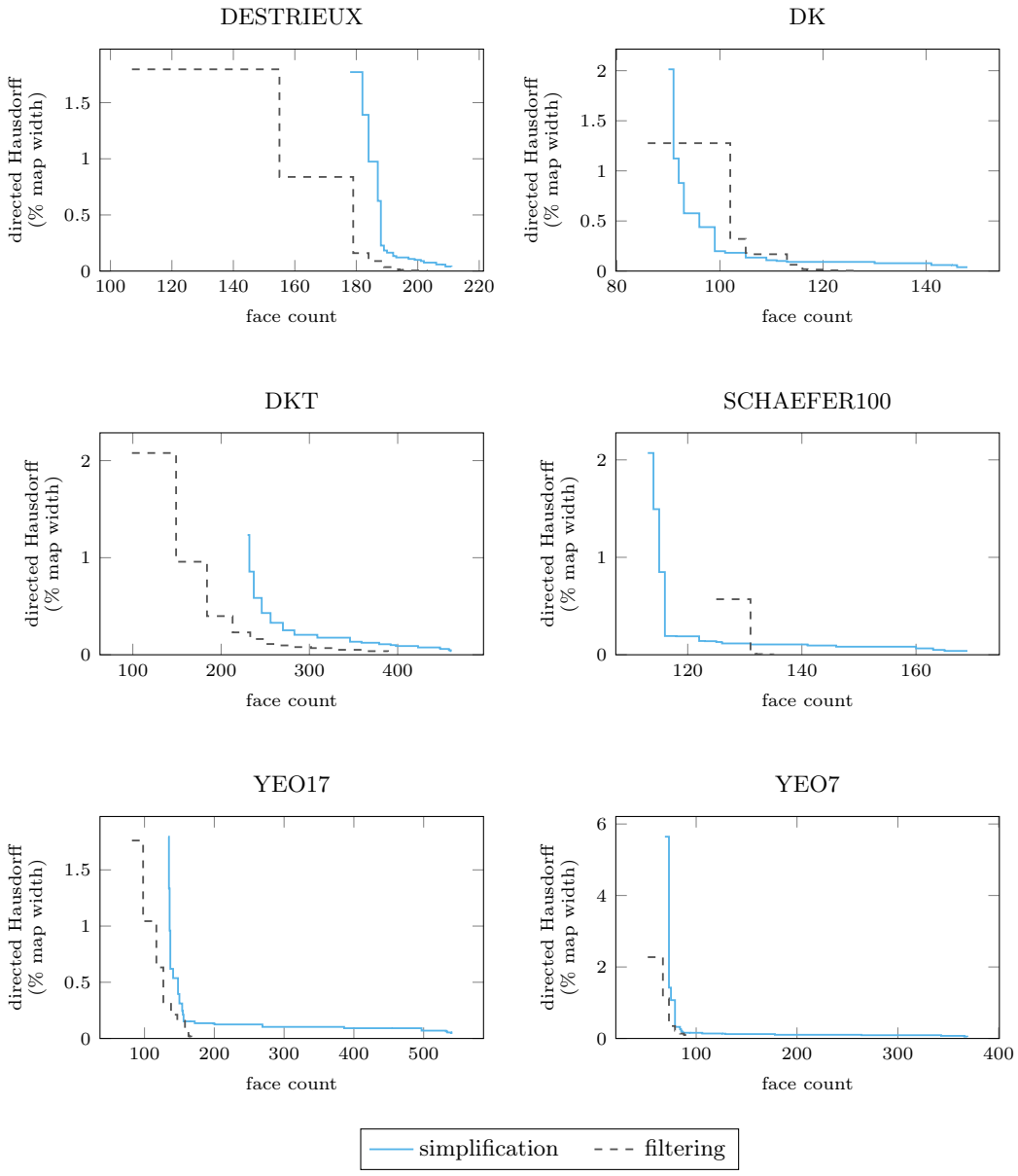

Figure 14: Best-achieved-value front (directed Hausdorff vs face count, as % of characteristic map width) for all six atlases. Simplification algorithms (solid) and filtering (dashed, gray). y-axis lower limit is 0; upper limit scales to the data. Supports Figure 8.