Structure-Aware Mesh Decimation

David Salinas, Florent Lafarge, Pierre Alliez

Structure-Aware Mesh Decimation

D. Salinas, F. Lafarge & P. Alliez

January 30, 2015

Abstract

We present a novel approach for the decimation of triangle surface meshes. Our algorithm takes as input a triangle surface mesh and a set of planar proxies detected in a preprocessing analysis step, and structured via an adjacency graph. It then performs greedy mesh decimation through a series of edge collapse, designed to approximate the local mesh geometry as well as the geometry and structure of proxies. Such structure-preserving approach is well suited to planar abstraction, i.e., extreme decimation approximating well the planar parts while filtering out the others. Our experiments on a variety of inputs illustrate the potential of our approach in terms of improved accuracy and preservation of structure.

1 Introduction

With recent advances on automated pipelines for geometric acquisition and processing, it is now routine to generate massive surface meshes of complex, large-scale scenes. This motivates the need for extreme mesh simplification methods aimed at generating meaningful levels of details. Although the automated simplification of surface meshes is a mature research topic with a wide range of methods [19], the extreme simplification of complex meshes has received less interest and remains notoriously difficult for the main following reasons:

- **Structure.** For extreme simplification minimizing a *local* geometric error metric is not sufficient: the structure comes into play and in particular the coarse-scale structures must be detected and preserved.
- **Filtering.** Extreme simplification involves filtering, i.e., "forgetting" small-scale details, and not just minimizing an exhaustive geometric error metric. Extreme simplification thus shares some goals with the process of abstraction which involves filtering among other structure-aware principles.
- **Defects.** Geometric noise and topological defects are present in complex meshes generated by automated pipelines such as dense photogrammetry. Beyond hampering simplification, such defects add further hurdles to the detection of coarse-scale structures. This calls for methods that are resilient to imperfect structure detection.

In this work we focus on the automated, coarse and structure-preserving decimation of surface triangle meshes (not necessarily 2-manifold).

1.1 Related Work

Related work ranges from mesh decimation to abstraction through approximation. We further focus our review on approaches that preserve a notion of structure.

Decimation. Edge collapse is the most common mesh decimation operator and led to very efficient and reliable algorithms [11, 18]. When using a local geometric error metric, the greedy mesh decimation approaches are in general not satisfactory for extreme simplification as the successive erroneous approximations may accumulate. A common example is when small but visually important details are progressively removed from the input scene. In addition, the structures may not be preserved during decimation unless explicitly detected. In general however, the quality of outputs is highly dependent on the quality of the initial structure detection. A wide range of variants have also been proposed to control the global error [3, 4, 7–9, 12, 14, 23, 27] but global error bounds do not imply structure preservation.

Approximation and Remeshing. Mesh optimization approaches [10, 13] are devised to reach a better tradeoff between model fidelity and conciseness. Such approaches are also shown suitable to feature-sensitive remeshing, as vertices automatically migrate onto sharp features when using appropriate error metrics. Feature-sensitive remeshing can also be achieved via resampling after detection [5]. Marinov and Kobbelt [20] proposed an integral error metric designed to derive a subdivision control mesh whose structure is properly adjusted and aligned to the major ge-

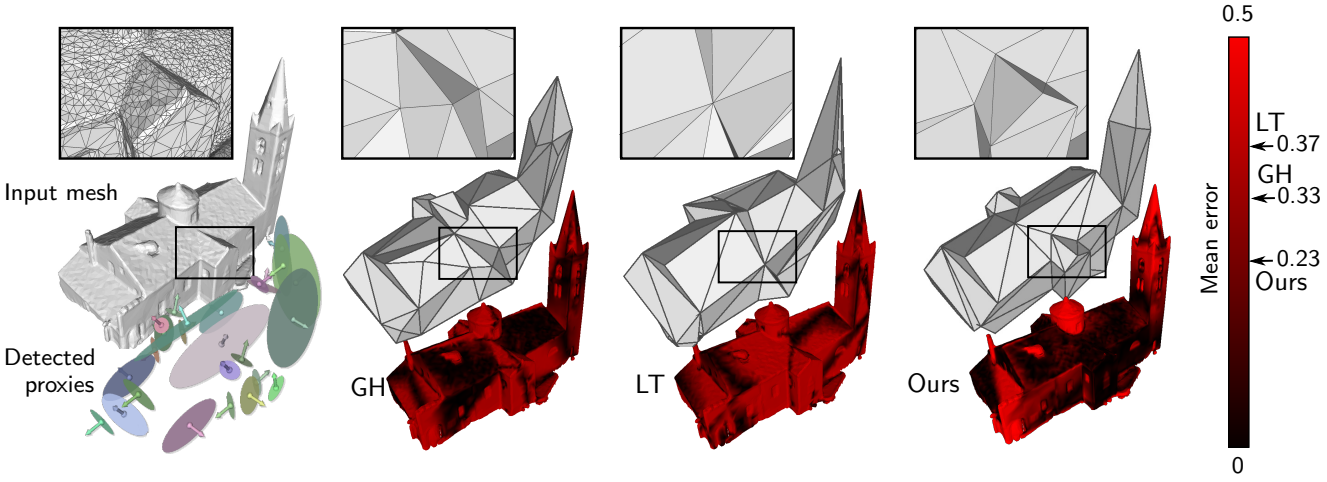

Figure 1: Overview. Our algorithm simplifies dense triangle surface meshes via a structured set of planar proxies (left) that guides the decimation process while preserving the structure. At coarse complexity (here 50 vertices), common mesh decimation approaches (middle) fail to reach low approximation error (see colored meshes) while preserving structure (see closeups). Error curves are depicted by Figure 7.

ometric features. The structure is thus preserved to some extend, only via the error metric.

Abstraction and Structure. Abstraction is the process of computing recognizable visual depictions of known objects via compact descriptions such as a handful of characteristic primitives [21]. Abstraction, which involves filtering and regularization, is especially relevant for the extreme simplification of urban scenes that requires removing small-scale details such as cars, chimneys or plants. Abstraction is also related to structure-aware shape processing [22].

There is no unique or universally accepted definition of structure. It may relate to the graph of sharp features [5], or to the global inter and intra semantic relations among the parts of shape rather than on the local geometry [22]. Structure-aware shape processing usually consists in two steps: structure detection and processing that uses the detected structure information.

In our context the structure is generated by a pre-processing step that detects a set of planar parts - referred to as proxies - and estimates their adjacency via a proximity graph. The structure is thus formed by the decomposition - possibly incomplete - into planar parts as well as by the adjacency relationships between associated proxies. The detection of proxies is performed at a spatial scale induced by a user-specified error tolerance from planarity. The spatial scale may be very large for parts such as facades of buildings. As in general two adjacent proxies meet at a sharp crease, the structure is not unrelated to sharp features but our approach departs from direct feature detection as follows:

- The detection of sharp creases is already an ill-posed problem - see Figure 12 - and our quest for structure-aware geometry processing conflicts with the unavoidable imperfect structure detection when dealing with defect-laden meshes derived from measurement data. In addition, what we need ranges from feature recovery to shape completion [24] through feature regularization [6] as we target extreme simplification. Our experiments show that proceeding by greedy decimation is substantially more robust than performing a one-step projection onto the planar proxies.

- We assume that the structure is contained in the graph of proxies. We then aim at preserving to some extend the graph during decimation but the geometric error term associated to proxies also automatically favors sharp features during decimation.

We choose to consider only planar primitives, omitting cylinders, conics and other primitives as planes are expressive enough to represent most parts of man-made shapes and yield close formula for our error metric. In particular, we observed that roughly 80% of a large-scale urban scene (shown Figure 19) is composed of planar parts.

1.2 Contributions and Overview

Our main contribution is a structure-aware mesh decimation algorithm robust to imperfect detection of coarse-scale structures. It takes as input a surface triangle mesh and a set of planar proxies pre-detected, and generates as output a simplified mesh where coarse-scale structures are

preserved via an error metric and specific rules. Our approach is hybrid in the sense that it combines mesh decimation with the preservation of geometry and structure induced by the proxies:

- The geometry of planar proxies is added to the local error metric used for edge collapses during decimation. This adds to the local scales the larger scales of planar parts, in accordance to the coarse-scale structures induced by the proxies. Such large scales (1) promote abstraction as vertices automatically migrate toward the proxies during decimation, (2) improve resilience to noise and (3) reduce the accumulation of local erroneous approximations during decimation.
- The structure of proxies is preserved during decimation by not performing edge collapse operators that violate a set of structure-preserving rules detailed below.

These two novel technical ingredients are both robust and effective at generating coarse levels of details from complex scenes. We can simplify unstructured parts such as trees in a urban scene while preserving structured parts such as the main facades of buildings. Figure 1 provides an overview of our algorithm.

2 Background

Simplicial complexes. Let $P \subset \mathbb{R}^3$ denote a finite set of points. A simplex is a non-empty simplex $\sigma \subset P$, of dimension its number of elements minus 1. Let τ be a non-empty subset of P. τ is a face of σ if $\tau \subset \sigma$ and a coface of σ if $\tau \supset \sigma$. A simplicial complex K (also referred to as a mesh) is a collection of simplices that contain all faces of its simplices. The set of its simplices of dimension $0/1/2$ is denoted by $\mathrm{V}_K \,/\, \mathrm{E}_K \,/\, \mathrm{T}_K$ and referred to as respectively its vertices, edges and triangles. For a simplex $\sigma \in K$ and a set of simplices S, we denote by $S(\sigma) = \{\tau \in S \mid \tau \cap \sigma \neq \emptyset\}$ the set of simplices intersecting σ. For example, the set of triangles intersecting e is denoted by $\mathrm{T}_K(e)$. When the context is clear we omit K in the notation, for instance, $\mathrm{T}_K(e)$ is simply denoted by $\mathrm{T}(e)$. We also use the notation $v_0 v_1 \dots v_k$ instead of $\{v_0, v_1, \dots, v_k\}$ to denote the simplex with vertices $v_0, v_1, \dots, v_k$. We are primarily dealing with simplicial complexes that approximate 2-dimensional manifolds, but are also dealing with non-manifold meshes. We define a boundary edge as an edge having exactly one triangle in its cofaces. The set of boundary edges is denoted by $\mathrm{E}_{\partial K}$. The point of P associated to a vertex v is denoted by $v = [v_x v_y v_z]$ or $\tilde{v} = [v_x v_y v_z 1]$ for its homogeneous counterpart. The length of an edge e is denoted by $|e|$ and the area of a triangle t is denoted by $|t|$.

Planar proxies. We assume that the input mesh exhibits near-planar parts that can be detected by common shape detection approaches [6, 15, 25]. These near-planar parts are represented by planar proxies. More specifically, a planar proxy φ consists of a set of vertices and a plane $ax+by+cz+d=0$ represented as a vector $\varphi = [a\ b\ c\ d]$, where $n = [a\ b\ c]$ is the unit normal vector to the plane. Figure 5 illustrates a mesh with its detected proxies and Figure 2 depicts a small set of proxies. For each planar proxy φ, the set of simplices $\sigma \in K$ such that all vertices of σ contain φ in their proxies is referred to as the mesh restricted to φ and denoted by $K|_\varphi$ (Figure 2). For every simplex $\sigma \in K$, we denote as $\mathrm{Proxies}(\sigma)$ the set of proxies φ such that $\sigma \in K_\varphi$. In other words, $\mathrm{Proxies}(\sigma)$ is the set of proxies that contain σ in their restricted mesh. Our approach is intended to be general so that a triangle or vertex can be assigned to none or an arbitrary number of proxies. For instance, the set of proxies $\mathrm{Proxies}(t)$ that contain a triangle $t \in \mathrm{T}_K$ may contain 0, 1 or more proxies.

A mesh with its set of proxies is depicted by a colored mesh as shown by Figure 5, where each proxy is assigned a random color. A triangle t of the mesh is depicted in gray when t does not belong to any proxy, that is when $\mathrm{Proxies}(t) = \emptyset$. When t belongs to several proxies the color of t is assigned the color of the proxy whose normal is closest to the normal of t.

Except for Figure 17, planes are always detected by a region-growing approach that we now detail. In a preprocessing step, we estimate the local tangent plane and planarity score at each triangle t by linear least squares fitting of a plane over a point set chosen as a subset of the mesh vertices within a local neighborhood. The latter is set either to the vertices of t in the noise-free case, or to a larger combinatorial neighborhood (all vertices at distance at most 2 in the graph of the mesh from the vertices of t) for noisy data sets (Figures 3 and 17). We then grow iteratively the regions, one region at a time, always seeding from the unconquered triangle with best planarity score. Each region is grown over adjacent triangles, one triangle at a time, while remaining under a normal and distance error tolerance. More specifically, we accept an adjacent triangle when its normal deviates less than the user-specified normal tolerance to the seed plane, and its maximum distance to the seed plane is under the user-specified distance tolerance. The seed plane is not recomputed during growing. After growing a region we reject it if its area is smaller than a user-specified parameter (otherwise every triangle could be detected as a planar proxy). Finally, upon termination we merge the adjacent regions that are near coplanar.

Edge collapse operator. An edge collapse $v_0 v_1 \to v$ is the mesh operator that merges the two vertices v_0 and v_1

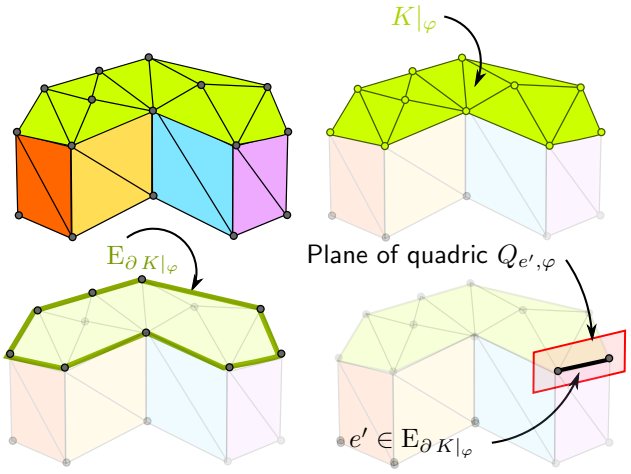

Figure 2: Proxies. Top Left: A mesh and its detected proxies. Top right: mesh restricted to proxy φ with 11 vertices, 18 edges and 11 triangles. Bottom left: nine edges of the boundary of $K|_\varphi$. Bottom right: orthogonal plane passing through $e \in \mathrm{E}_{\partial K|_\varphi}$, used to build a quadric for the proxy boundary - see Section 3.

to a unique vertex v. Mesh decimation algorithms based on edge collapses often proceed as follows: (1) For each edge collapse $v_0v_1 \to v$, define a cost related to an error metric and its corresponding placement strategy devised to find a locally optimal point location for v. (2) Compute an initial prioritized heap of edge collapse operators with increasing cost. (3) Iteratively extract the operator with lowest cost from the heap, compute its optimal location, collapse the associated edge there and update the prioritized heap for edges in the local neighborhood. Section 4 provides details of the cost and optimal location used in our algorithm. Both rely upon a quadric error metric attached to each operator that we now define.

3 Error Quadrics

In the original method from Garland and Heckbert [11] (referred to as GH), the authors associate to each vertex a quadric which represents an approximation of the error between the current and the initial mesh. This quadric is encoded as a 4x4 symmetric matrix, used to compute the sum of squared distances from a point to a set of planes. More specifically, let P be a plane $ax + by + cz + d = 0$ represented as a vector $P = [a\ b\ c\ d]$. We associate to this plane the following quadric:

$$Q_P = PP^T = \begin{pmatrix} a^2 & ab & ac & ad \\ ab & b^2 & bc & bd \\ ac & bc & c^2 & cd \\ ad & bd & cd & d^2 \end{pmatrix}$$

The squared distance of a point v to P can be written $d(v, P) = v^T Q_P v$. An appealing feature of such quadric is that it can be encoded with only 10 coefficients and represent a sum of squared distances from a point v to a list of planes (P_i) as a sum of quadrics:

$$\sum d(v, P_i)^2 = \sum \tilde{v}^T P_i {P_i}^T \tilde{v} = \tilde{v}^T (\sum Q_{P_i}) \tilde{v}.$$

We also use quadrics in our setting but depart from GH in the sense that we optimize simultaneously for several criteria by minimizing the sum of squared distances to (1) the supporting planes of the local mesh triangles, (2) the planes of the local set of proxies where detected, (3) the boundary of proxies and (4) the boundary of the mesh. Each quadric is weighted by an area for scale invariance and lower sensitivity to the initial mesh density.

Inner quadric. For a triangle $t \in \mathrm{T}_K$, we denote as P_t the supporting plane of t and Q_{P_t} its associated quadric. For a planar proxy φ we denote as Q_φ the quadrics defined with the plane of the proxy. Recall that the set of proxies that contain t is denoted as $\mathrm{Proxies}(t)$. Each triangle t of $\mathrm{T}(e)$ is associated to a quadric Q_t:

$$Q_t = \begin{cases} Q_{P_t} & \text{if } \mathrm{Proxies}(t) = \emptyset \\ (1-\lambda) Q_{P_t} + \lambda \sum_{\varphi \in \mathrm{Proxies}(t)} Q_\varphi & \text{otherwise} \end{cases}$$

The inner quadric of an edge e is defined as a weighted sum:

$$Q_{\mathsf{inner}}(e) = \sum_{t \in \mathrm{T}(e)} |t| Q_t.$$

This quadric is used to compute the cost and optimal placement for an edge collapse operator. Parameter λ, referred to as the abstraction parameter, provides a means to trade mesh versus proxy fidelity. For instance when $\lambda = 1$, the vertex is placed at the intersection of proxies when two proxies pass through edge e. When $\lambda = 0$ or e is not associated to any proxy, the quadric only approximates the local error metric to the mesh. The impact of λ on the decimation is illustrated by Figures 15 and 16. Our metric is hybrid in the sense that it relies upon both the local scale of the mesh triangles and the more global scale of proxies. The latter scale yields an increased robustness on defect-laden meshes, see Figure 3.

Boundary quadric. For an edge e' and a plane $R \supset e'$, we denote as $Q_{e',R}$ the quadric associated to the plane orthogonal to R that contains e'. Recall that $\mathrm{E}_{\partial K}$ (resp $\mathrm{E}_{\partial K|_\varphi}$) is the set of edges of K (resp. $K|_\varphi$) that are contained in a unique triangle of K (resp. $K|_\varphi$). For a

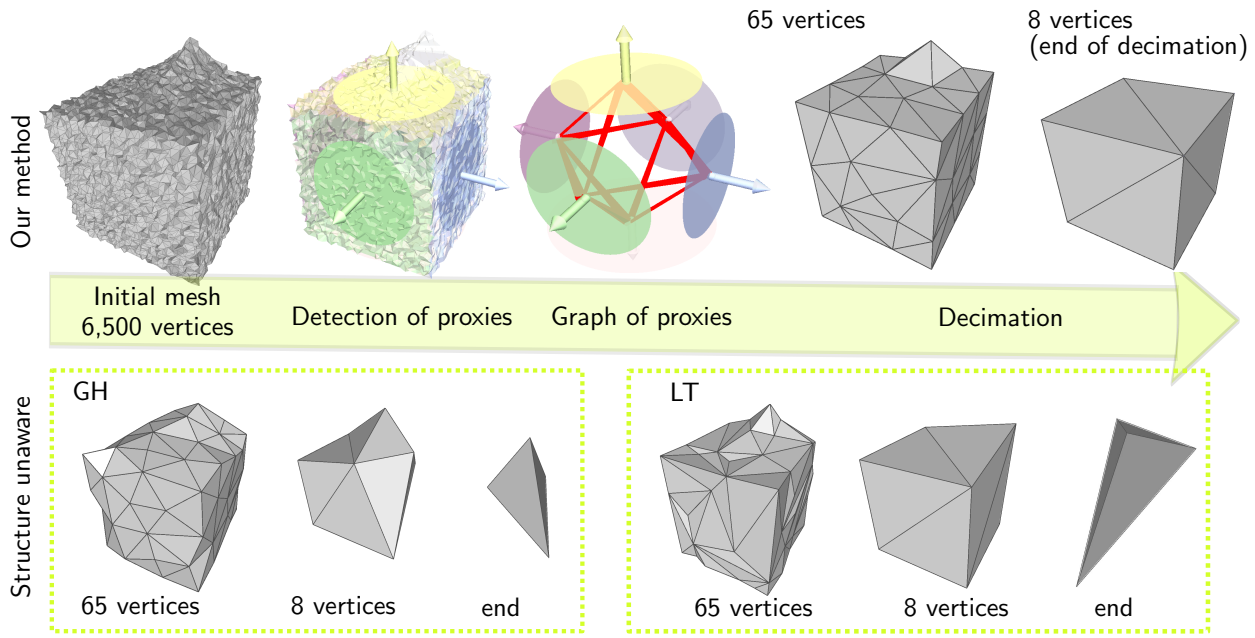

Figure 3: Structure-preserving decimation of a mesh corrupted with uniform noise (magnitude 140% of the average edge length of the mesh). Top: When reaching 8 vertices, no more edges are collapsible thanks to the structure-preserving rules induced by the graph of proxies. Bottom: meshes decimated through the quadric error metric [11] (GH) and the volume-preserving approach [17] (LT). On such defect-laden meshes the proxy-fitting term also yields an increased robustness at coarse decimation levels.

boundary edge $e' \in \mathrm{E}_{\partial K}$ (resp. $e' \in \mathrm{E}_{\partial K|_\varphi}$), we denote as t'_e the only triangle of K (resp. $K|_\varphi$) passing through e'. The boundary quadric of an edge e is then defined as :

$$Q_{\text{bdry}}(e) = \sum_{e' \in \mathrm{E}_{\partial K}} |t'_e| Q_{e',t'_e} + \sum_{\mathrm{E}_{\partial K|_\varphi}} |t'_e| Q_{e',\varphi}.$$

The first term prevents over-simplifying the mesh boundary [11] as without it a planar mesh can be collapsed to a point at no cost. The second term preserves the shape of proxies as a proxy φ contains via its restricted triangulation $K|_\varphi$ more information than just an infinite plane (see Figure 2). We detail next the cost and optimal vertex placement.

4 Cost and Placement

The original GH algorithm [11] consists of computing an initial quadric Q_v for each vertex v, then summing up quadrics at each collapse operator. More specifically, for an edge collapse $e = v_0 v_1 \to v$, the quadric of e (Q_e) is computed by adding the quadrics of its two vertices, the optimal vertex location v being obtained via minimizing the cost $v^T Q_e v$. This approach keeps this way a memory of the decimation through adding quadrics. Nevertheless, it has been observed that memoryless decimation is usually more effective in terms of approximation error [18]. We thus adopt the memoryless approach and recompute Q_e from the current mesh before each collapse operator.

Decompose the quadric matrix Q_e as follows:

$$Q_e = \begin{pmatrix} A & -f \\ -f^T & g \end{pmatrix}$$

Minimizing $v^T Q_e v$ involves solving the linear system $Ax = f$. However, the determinant of A vanishes when the vertices of the star of e lies in only one or two planes. Similarly to Lindstrom [16] we deal with degenerate cases as follows. We compute a singular value decomposition $A = U \Sigma V^T$, then truncate small eigenvalues of Σ and store the result in Σ^+ :

$$\Sigma^+_{ii} = \begin{cases} 1/\Sigma_{ii} & \text{if } \Sigma_{ii}/\Sigma_{11} > \varepsilon \\ 0 & \text{otherwise} \end{cases}$$

where Σ_{11} denotes the largest singular value and ε is a parameter set by default to 10^{-3}. Denoting by $\hat{x}$ the barycenter of the vertices of the star of v_0 and v_1, we set the *SVD solution* of Q_e as:

$$x = \hat{x} + V \Sigma^+ U^T (f - A\hat{x}).$$

Notice that when $\Sigma^{-1} = \Sigma^{+}$, the SVD solution is similar to the one of the linear system, i.e., $x = A^{-1}f$.

We now detail our approach to compute the optimal location for v after the collapse operator $e = v_0v_1 \rightarrow v$. When collapsing the edge e, the set of proxies of $\mathrm{Proxies}(v)$ is defined by $\mathrm{Proxies}(v_0) \cup \mathrm{Proxies}(v_1)$ and quadric of e is defined by:

$$Q_e = (1-\mu)Q_{\mathsf{inner}}(e) + \mu Q_{\mathsf{bdry}}(e),$$

where $\mu \in (0,1)$ is a parameter used to trade boundary for inner simplification. The optimal location for v is obtained through searching for the SVD solution of Q_e, and the cost of collapsing e is set to $v^T Q_e v$. The decimation process may thus be seen as a progressive deformation of the initial mesh onto the set of proxies at a rate controlled by the abstraction parameter λ, see Figure 15.

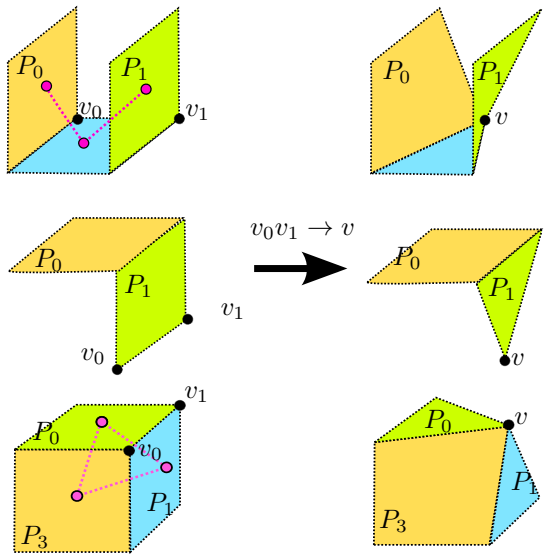

Figure 4: Structure-preserving rules. Three types of structure-altering operators prevented when simulating the edge collapse operator $v_0v_1 \rightarrow v$. Top: collapsing would introduce a new edge P_0P_1 into the graph of proxies (depicted in pink). Middle: collapsing would degenerate proxy P_1. Bottom: collapsing would degenerate the corner vertex between P_0, P_1 and P_3.

5 Structure-preserving

Structure-unaware decimation algorithms often destroy the coarse-scale structures at coarse decimation levels (Figures 13 and 21). Our set of proxies is amenable to a structure-aware decimation algorithm when deriving an undirected proximity graph.

Graph of proxies. From the set of detected proxies we compute a graph $G = (\mathrm{V}_G, \mathrm{E}_{G,\alpha})$ where each proxy is represented as a vertex of V_G. The edges $\mathrm{E}_{G,\alpha}$ of G consist of pairs of proxies that have a distance from each other lower than α. More specifically, the distance between two proxies is approximated as the minimum Euclidean distance between the two point sets of proxies. A clique $\sigma = \{P_1, \ldots, P_n\}$ of a graph G is a subset of vertices of G such that all pairs of vertices of σ are connected by edges of G.

Consider for the moment an abstracted version of a dense, defect-free input scene where all proxies are correctly detected and all points lie within their proxy's plane. The ideal graph of proxies coincides with the proximity graph such that, e.g., a cube would be structured by a dual octahedron. Although on real-world data such perfect graph detection is hopeless, the proximity graph provides enough information about the structure to devise three structure-preserving rules: respectively graph, proxy and corner preservation, as illustrated by Figure 4.

Graph preservation. Two parts that are not connected in the graph should not be connected during decimation. We accept an edge collapse $v_0v_1 \rightarrow v$ only if all pairs of proxies in $P_v = P_{v_0} \cup P_{v_1}$ are edges of the initial edges of G. This condition may be seen as a *meta* link condition that prevents undesirable operations such as joining two walls of an urban scene that were initially disjoint.

Proxy preservation. When performing a collapse operator the resulting vertex receives the union of proxies of the two edge vertices. Thus, proxies are always witnessed by at least one vertex. However, a proxy may degenerate into a single vertex or edge during decimation. To prevent such degeneracy we reject a collapse operator when the number of vertices forming this proxy after collapse drops below a user-specified parameter. Although 3 are sufficient to guarantee a valid dimension, we set this parameter to 4 in all experiments carried out on urban scenes.

Corner preservation. Corners are visually very noticeable on scenes made up of large planar parts (Figure 13). To prevent loosing corners or letting the vertices migrate far away from corners, we must detect and preserve them during decimation. Inferring corner is however non trivial even from the graph of proxies for two main reasons: (1) initially, corners may not be represented as vertices having at least 3 proxies, as depicted by Figure 5 (left). (2) corners may be at the junction of more than three proxies, as depicted by Figure 6 (left). All intersections of 3 of the proxies are in general distinct as depicted by Figure 6

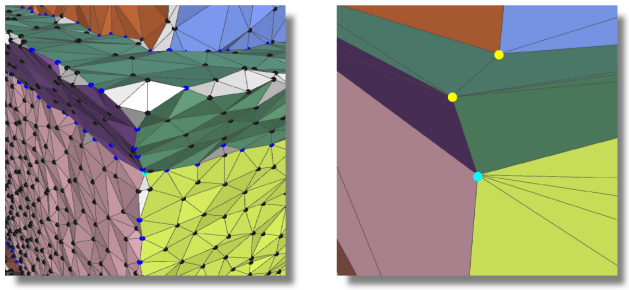

Figure 5: Witnessing corners. Left: initial mesh and number of proxies assigned to vertices. Notice how only one of the three corners is initially witnessed after initial detection. Right: number of proxies assigned to vertices after decimation. Vertices with 1, 2, 3 and 4 proxies are depicted respectively as black, blue, yellow and cyan dots.

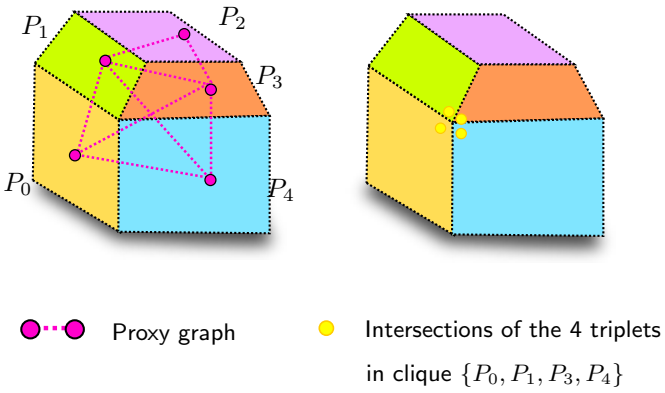

Figure 6: Multiple corners. The graph of proxies has a clique σ with size larger than 3. When the planes are not exactly aligned, intersecting all triplets included in σ yields $\binom{|\sigma|}{3}$ distinct points.

(right), and we must find a single optimal point location for this cluster of intersections.

One solution is to perform data completion [24], through a graph cut formulation applied to a discretized space. However, as our goal differ from reconstruction we favor an approach that takes advantage of the scale-space traversal performed during decimation.

Let us denote by $|\sigma|$ the size of a clique defined by its number of vertices. A clique σ is maximal if there is no clique of G that is distinct from σ and contains σ. Observe that when the proxy graph G is ideal, i.e., captures all inferred proxy edges, every scene corner is represented by a maximal clique of G of size at least 3. Corners are likely to attract vertices during the decimation process as they coincide with the "best" location to minimize the error metric locally. However, best herein depends on the current scale and vertices close to corners may be moved further during decimation. We would like instead to snap them to corners. We explain next how we infer corners during the decimation and how points are chosen to represent such corners.

Let σ be a maximal clique of G with size $|\sigma| \geq 3$. We call a vertex v a witness of σ if v contains at least three proxies of σ and denote as $\mathrm{Witness}(\sigma)$ the set of witnesses of σ. We also denote as $\mathrm{Corners}(G)$ the set of witnessed maximal cliques of G of size strictly greater than or equal to 3. For instance, in Figure 5, only one clique is initially witnessed (left) whereas three cliques are witnessed after decimation (right), and thus $\mathrm{Corners}(G)$ has respectively one and three elements. We infer corners with the set $\mathrm{Corners}(G)$ (hence corners may be inferred during the decimation as $\mathrm{Corners}(G)$ may grow during the decimation). The first time a clique σ becomes witnessed (i.e. becomes a new element of $\mathrm{Corners}(G)$), we compute a representing point p_σ as follows. We pick one vertex v in $\mathrm{Witness}(\sigma)$ and set the point p_σ to the closest intersection from v of all the $\binom{|\sigma|}{3}$ triplets of planes included in σ. Finally, to prevent vertices from migrating far away from inferred corners, we reject an edge collapse operator when the distance of a point p_σ to its closest witness vertex increases after a collapse.

Note that albeit we preserve proxies and corners that correspond respectively to vertices and maximal cliques of the proxy graph, we do not add analog structure-preserving rules for sharp creases that correspond to edges in the graph. The reason is that (1) removing small crease edges is sometimes desirable, see e.g. edge P_0P_3 depicted by Figure 6. Adding a rule for longer sharp creases would thus require defining a parameter, and (2) most crease edges are already preserved even at coarse approximation with our set of rules and our metric. Other structure-preserving rules devised to preserve other features such as boundary corners may be added as well but we found in experiment that adding more rules prevent decimation from reaching coarse levels of details. We thus opted for a set of structure-preserving rules that is both small enough to allow coarse decimation and meaningful to preserve most salient structures.

6 Experiments

Our method is implemented in C++. As mesh data structure we use the recent skeleton-blockers (SB) data-structure [2]. Our main reason for using such data structure is that it can represent any simplicial complex (not

necessarily manifold) and is efficient at performing edge collapse operators. We compare our approach to the volume-preserving decimation approach (LT) [18] implemented in the CGAL library, to the GH approach [11] implemented with our SB data structure, and to the Variational Shape Approximation (VSA) approach [10]. Our method does not require the input to be manifold but the LT implementation of CGAL does. We thus convert the input mesh to a 2-manifold mesh for comparison with LT. Errors between approximations and input meshes are computed as the symmetric Hausdorff distance or symmetric mean distance. The symmetric Hausdorff distance between two spaces X and Y is defined as $d_{\mathrm{H}}(X,Y) = \max\{\sup_{x\in X} d(x,Y), \sup_{y\in Y} d(y,X)\}$ and is computed by dense point sampling of X and Y. The mean distance has a similar definition except that distances are averaged instead of taking the maximum. Denote by X' and Y' the discretization of X and Y, the mean distance between X and Y is approximated by $\max\{\frac{1}{|X'|}\Sigma_{x\in X'} d(x,Y), \frac{1}{|Y'|}\Sigma_{y\in Y'} d(y,X)\}$.

All timings are measured on an Intel i7 2.70GHz processor with 16GB RAM. Our method is evaluated on various designed and real world models with size ranging from thousands to millions of vertices. Errors and timing are shown in Table 1 which shows that our sequential implementation decimates around 2K vertices per second.

Parameters. We set the abstraction parameter $\lambda = 0.8$ in all experiments except for Figure 15 where we evaluate the impact of this parameter on the decimation. The boundary parameter μ is set to 0.8. In all experiments shown the planar proxies are detected using a region-growing approach [15]. The link condition to preserve the 2-manifold property is not enforced for models with non trivial topology as we wish to remove small holes and handles during decimation. Mesh inversion however is prevented by checking that the normals to all triangles vertex-incident to the collapsed edge do not vary too much after collapse. More specifically, for every triangle around a vertex about to be collapsed, we check that the angle between the normal of this triangle and the corresponding triangle obtained after the collapse (if any) is lower than 150 degrees. However, as we take arbitrary simplicial complexes as input that may not be 2-manifold nor orientable, we perform the normal-inversion test only when the mesh is locally orientable.

Designed models. We start by performing a few sanity checks on ground truth models designed in Trimble Sketchup, re-meshed uniformly and corrupted with uniform noise. When proxies are well detected, our algorithm stops as expected to coarse and accurate approximations

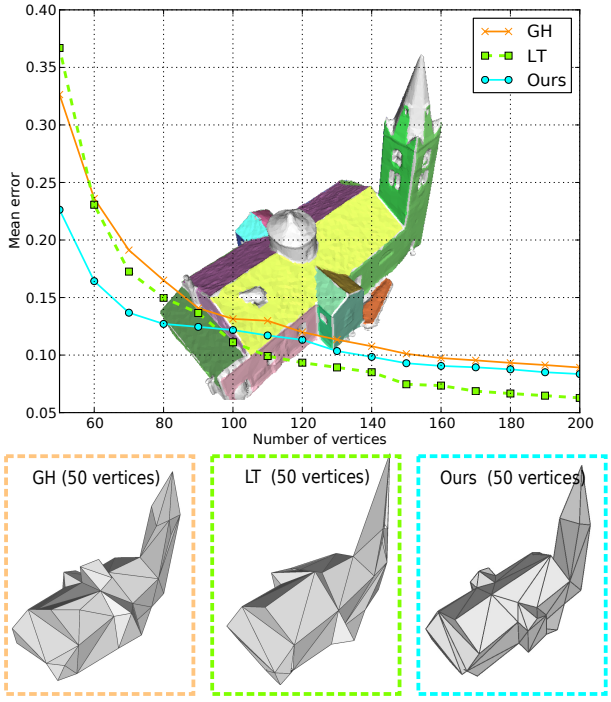

Figure 7: Comparison. We plot the mean geometric error against mesh complexity while decimating the church building shown Figure 1 with Garland-Heckbert (GH), Lindstrom-Turk (LT) and our method. Our structure preserving rules and metric generates more error at fine complexities due to abstraction, but lower error at coarse complexities by preserving main structural elements. Note that the fast increase in the error after 60 vertices is due to the filtration of the small tower on the middle of the roof. The latter does not belong to any proxy, hence is not detected as a structural element.

(Figures 3, 10 and 11). When the proxy detection step is imperfect on meshes with ill-defined sharp features, the latter are often recovered during decimation via the proxy term in the quadrics, and preserved through the structure-preserving rules. Furthermore, the latter rules provide a stopping criterion in the sense that no edge is collapsible upon termination of the decimation. In the examples shown the final mesh is very coarse, visually coherent and close to the minimal complex required to represent the model or scene without loosing coarse structures. Structure-unaware methods would require defining an error-based stopping criterion through a trial-and-error visual inspection process. We found in experiments that specifying a tolerance error for proxy detection makes it easier to inspect the quality of proxy detection before decimation. Figure 11 (right) illustrates that when the error tolerance for proxy detection is larger, the mesh converges

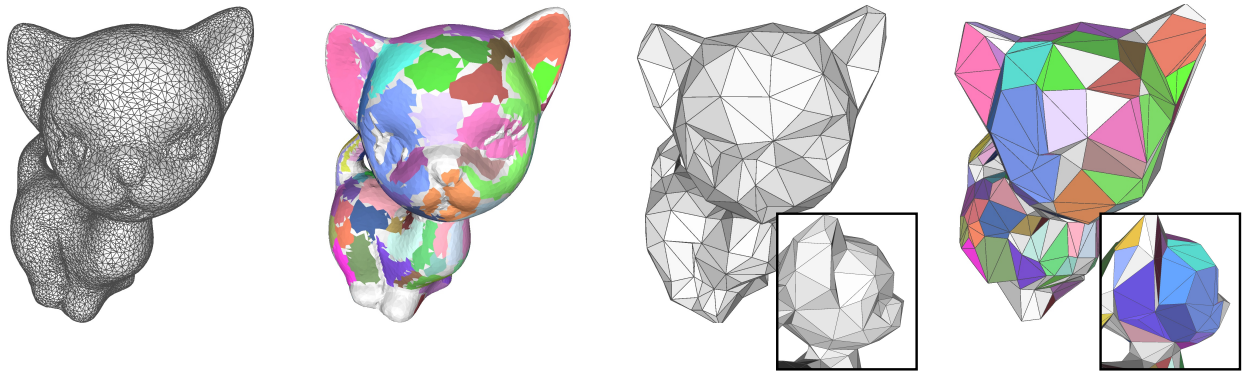

Figure 8: Kitten. This non-developable surface challenges our piecewise planar approximation. From left to right, input mesh (10,000 vertices), initial planar proxies, mesh simplified with Garland-Heckbert and with our method for a similar complexity (228 vertices). The right mesh is the output when decimation stops because of the structure-presering rules. In this case, our algorithm generates an extra mean-error compared to Garland-Heckbert (0.35 vs 0.23) because our error metric favors close fitting to the initial planes (see close-ups).

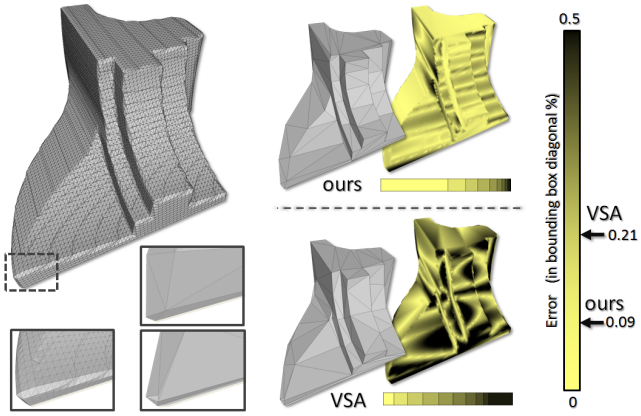

Figure 9: Comparison with VSA. A model of a Fandisk reconstructed with a voxel-based method (available from AIM@Shape repository) is decimated to 77 vertices with our method (top right) and VSA (bottom right). Our method yields a lower mean error than VSA.

to a coarser approximation. This error tolerance is our mean to select the desired level of details for the final approximation. The noise robustness of our algorithm for three different proxy detection approaches is illustrated by Figure 17. The approaches are: (1) region growing on the mesh (used by default in all other experiments), (2) region growing based on the mesh vertices only (adjacency relationships are derived from a k-nearest neighbor graph instead of from the triangulation) and (3) RANSAC [25].

Real world models. Results for real-world case are shown by Figures 1, 9, 18, 15, 20 and 21. On defect-laden meshes generated via a structure-from-motion reconstruction pipeline the detection of proxies is substantially harder (Figure 14). Our results show that even when the detection of proxies is imperfect we generate meaningful decimated meshes in terms of both visual coherence (Figure 20 and 21) and objective error metric (Figures 7 and 16). For the sake of comparison with an abstraction process driven by the detected proxies, we depict in Figure 21 a mesh generated by one-step projection of vertices onto the plane of their closest proxy, where detected. Our experiments revealed that this approach is too sensitive to the quality of proxy detection and performs poorly at recovering clean sharp features, even when followed by decimation. In other words, our experiments revealed that it is substantially more robust to traversal the scales progressively with a dual error metric relating to both the current mesh and the proxies. Robustness to different proxy detection and graph parameters is illustrated by Figure 18. Detecting more proxies provides a means to lower the approximation error. We can vary the graph proximity parameter α in a relatively large interval without affecting too much the result.

Limitations. Our decimation approach is both automated and efficient as for other greedy methods. Nevertheless, it may not generate perfectly abstracted models starting from dense defect-laden meshes: the final output meshes may, e.g., not have all their faces perfectly aligned - albeit close - with abstracted planes. We also observed in

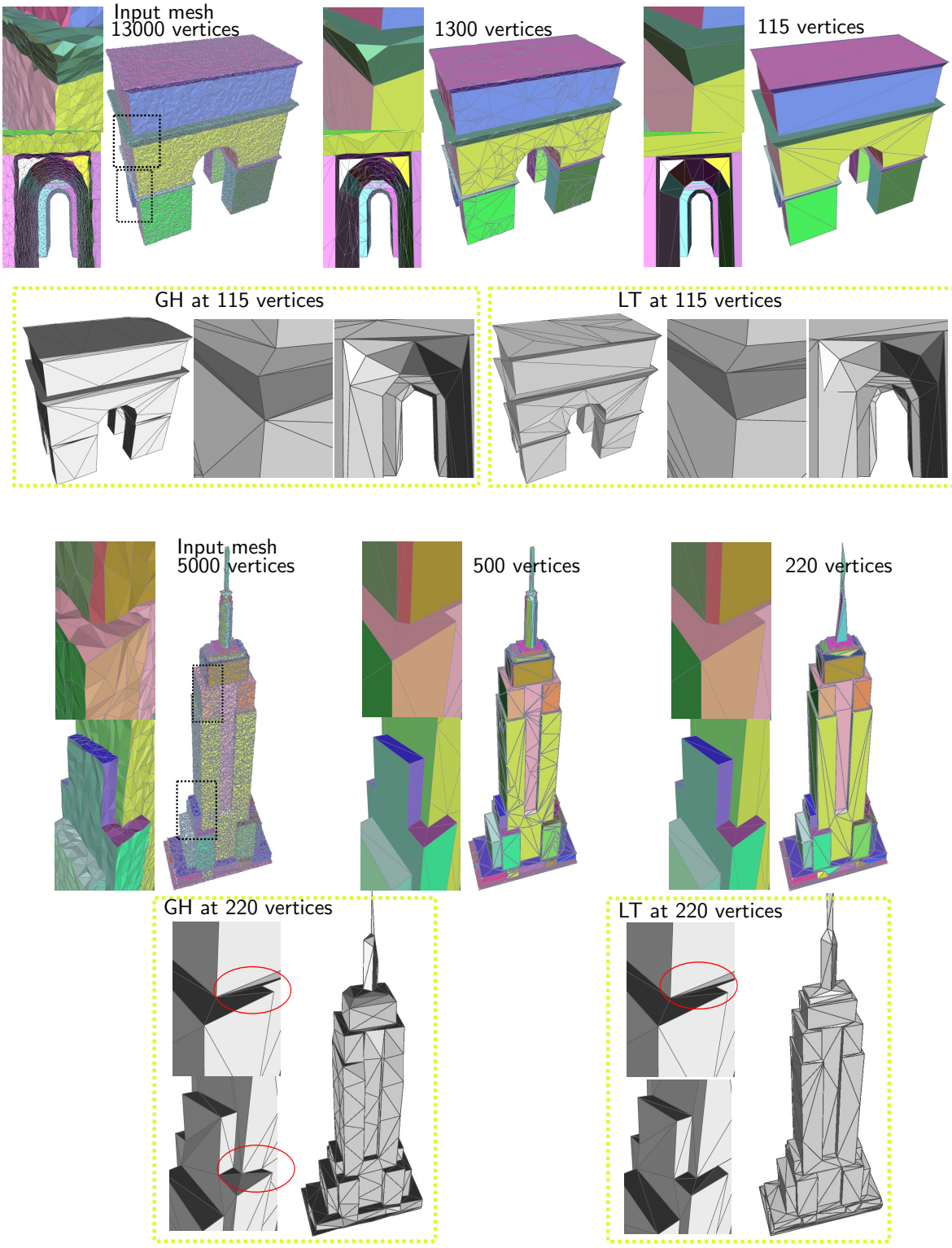

Figure 10: Triumphal Arch (top) and Empire State Building (bottom). Both models have been designed in Trimble Sketchup, re-meshed uniformly and corrupted with uniform noise (respective magnitude 30% and 15% of the average edge length of the mesh). From left to right: input mesh with detected planar proxies, mesh decimated with 10% vertices and final mesh obtained when no more edge is collapsible due to structure-preserving rules.

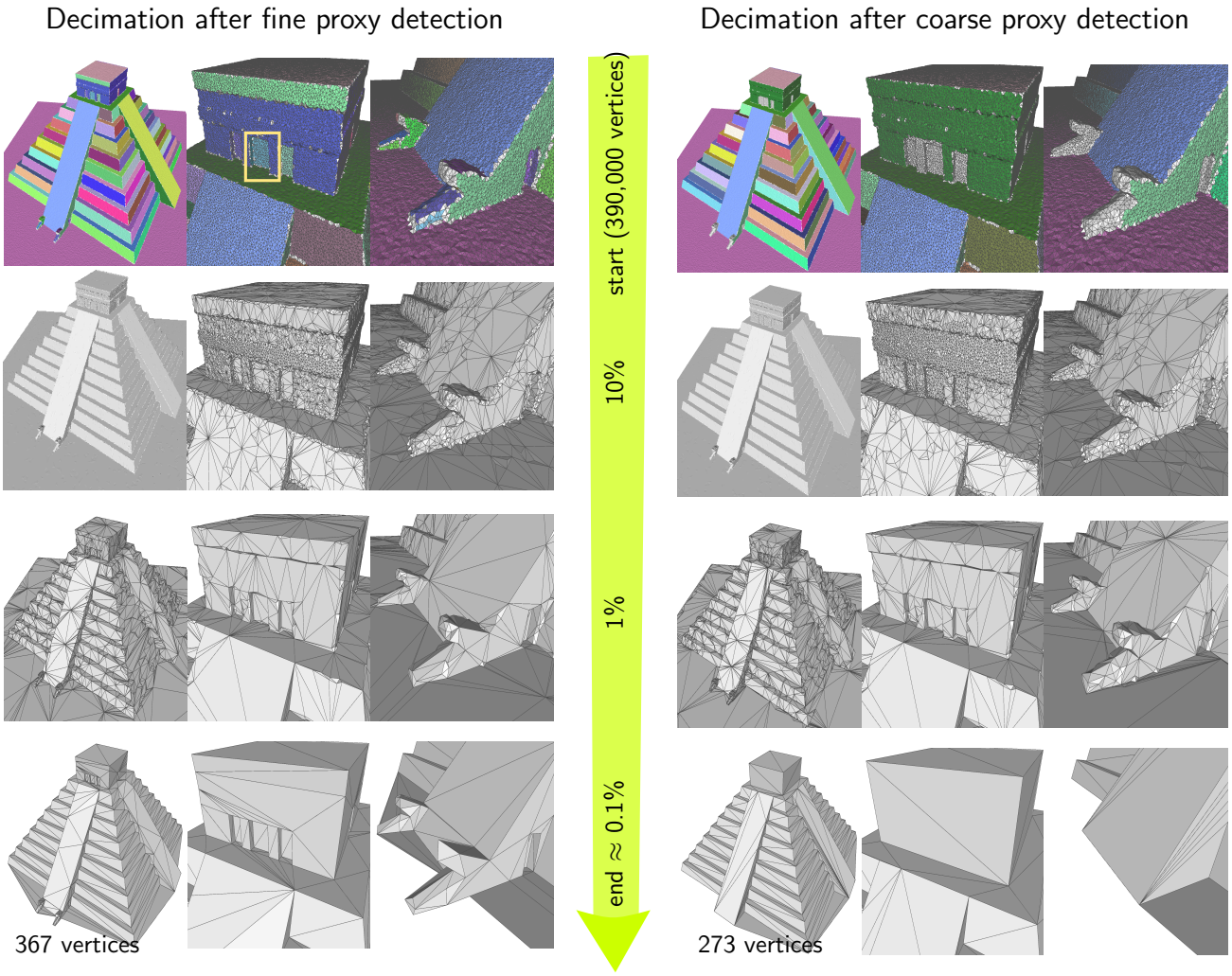

Figure 11: Chichen Itza. The model (390K vertices) has been designed in Trimble Sketchup, re-meshed uniformly and corrupted with uniform noise (magnitude 15% of the average edge length). Left: the initial fine proxy detection step captures the doors and some details of the entrance statue. Right: a coarser proxy detection step captures only 5 planes on the top of the pyramid. In both cases the structures are well preserved during decimation even at the end when no edge is collapsible anymore. Results obtained via other methods for this model are illustrated by Figure 13, and a different closeup on the door is depicted by Figure 12.

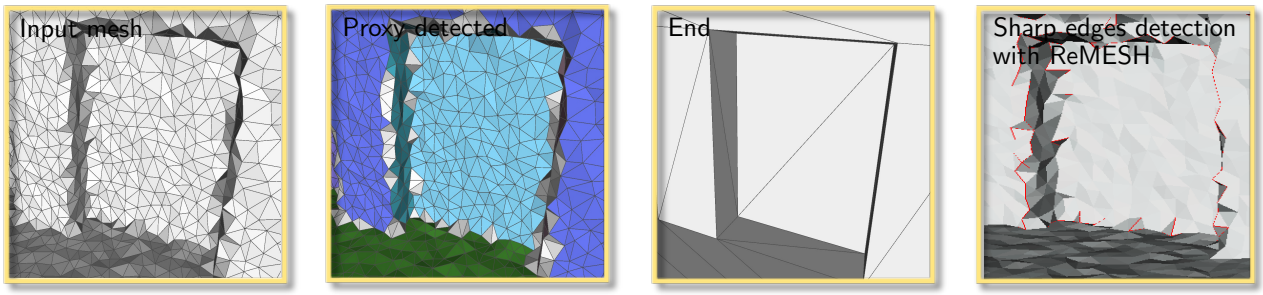

Figure 12: Chichen Itza (closeup). Closeup on a door of model depicted by Figure 11 and decimation with a small tolerance error for proxy detection. While sharp features such as crease edges and corners are ill-defined in the input mesh, they are recovered during decimation via the error metric to proxies, then preserved through structure-preserving rules.

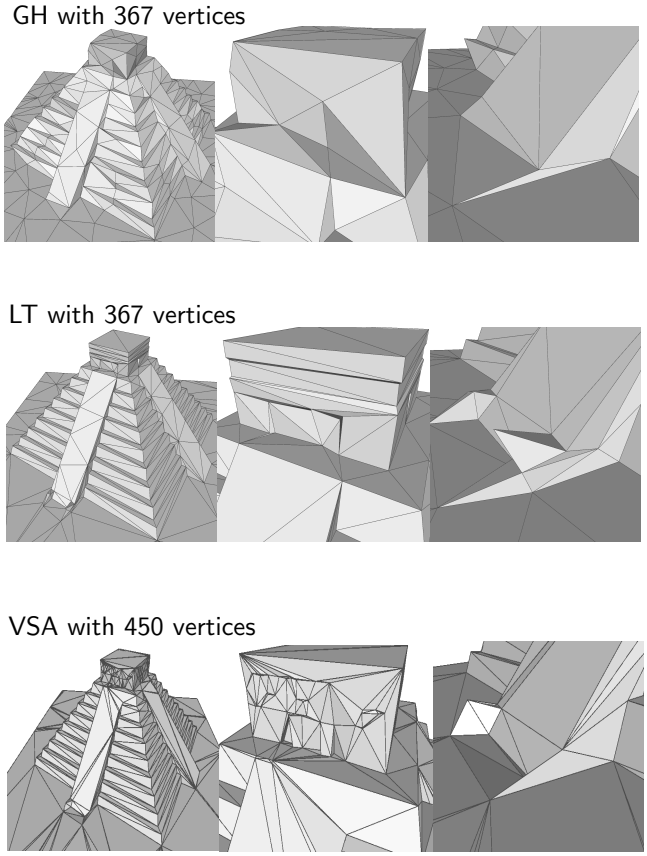

Figure 13: Comparison. Outputs of VSA, GH and LT when decimating the Chichen Itza model at coarse resolution. Unlike our method, structure parts such as stairs or facade details are already altered at 367 vertices.

experiments that using a richer set of mesh optimization operators such as edge flip, edge split or global vertex relocation would further improve the quality of results. However, proving termination of such pliant approach would be an issue and the algorithm complexity would substantially increase. Another weakness of our approach is the use of a fixed set of proxies after detection via a single tolerance error. Performing a simplification of the proxies in tandem with the mesh decimation seems a natural direction to explore, but this would contradict the current approach where proxies are used to avoid error accumulation and improve resilience to defects. Finally, planar proxies are not adapted to approximate free-form nor non-developable surfaces as shown by Figure 8.

7 Conclusion

We introduced a mesh simplification approach that combines mesh decimation with fidelity to a set of planar proxies detected and structured in an adjacency graph during a pre-processing step. The common error to the decimated mesh is augmented by an error metric to the proxies and boundaries of planar parts delineated by proxies. This error-to-proxy as well as structure-preserving rules derived from the graph of proxies are used to recover and preserve the coarse-scale structures. Our approach is particularly well suited to the robust simplification of defect-laden meshes generated by automated reconstruction pipelines applied to physical scenes composed of man-made objects.

Our experiments show that the decimated meshes provide meaningful abstractions, and an abstraction parameter provides a means to trade mesh versus proxy fidelity. Compared to direct mesh abstraction through, e.g., projection onto proxies, the scale-space traversal operated by the mesh decimation, combined with the aforementioned

combined metric, reveals substantially more robust at recovering the sharp features.

Further work includes the simplification of proxies during decimation, the use of domain-specific semantic rules and a generalization to non-planar proxies.

Acknowledgments. The authors wish to thank Acute3D and InterAtlas for providing the urban meshes of Paris. This work is partially funded by an ERC Starting Grant "Robust Geometry Processing" (257474).

References

[1] Acute3d, 2010.

[2] D. Attali, A. Lieutier, and D. Salinas. Efficient data structure for representing and simplifying simplicial complexes in high dimensions. *International Journal of Computational Geometry and Applications (IJCGA)*, 22(4):279–303, 2012.

[3] H. Borouchaki and P.J. Frey. Simplification of surface mesh using hausdorff envelope. *Computer Methods in Applied Mechanics and Engineering*, 194(48-49):4864 – 4884, 2005.

[4] Mario Botsch, David Bommes, Christoph Vogel, and Leif Kobbelt. GPU-based tolerance volumes for mesh processing. In *Pacific Conference on Computer Graphics and Applications*, pages 237–243. IEEE Computer Society, 2004.

[5] Mario Botsch and Leif Kobbelt. Resampling feature and blend regions in polygonal meshes for surface anti-aliasing. *Computer Graphics Forum*, 20(3):402–410, 2001.

[6] Alexandre Boulch, Martin de La Gorce, and Renaud Marlet. Piecewise-Planar 3D Reconstruction with Edge and Corner Regularization. *Computer Graphics Forum*, 33(5):55–64, 2014.

[7] A. Ciampalini, Paolo Cignoni, Claudio Montani, and Roberto Scopigno. Multiresolution decimation based on global error. *The Visual Computer*, 13(5):228–246, 1997.

[8] J. Cohen, A. Varshney, D. Manocha, G. Turk, H. Weber, P. Agarwal, F. Brooks Jr., and W. Wright. Simplification envelopes. *Proceedings of SIGGRAPH*, pages 119–128, 1996.

[9] Jonathan Cohen, Dinesh Manocha, and Marc Olano. Successive mappings: An approach to polygonal mesh simplification with guaranteed error bounds. *International Journal of Computational Geometry and Applications*, 13(1):61–96, 2003.

[10] David Cohen-Steiner, Pierre Alliez, and Mathieu Desbrun. Variational shape approximation. *ACM Trans. Graph.*, 23(3):905–914, August 2004.

[11] Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '97, pages 209–216, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[12] André Guéziec. Surface simplification inside a tolerance volume. Technical Report 20440, IBM, 1996.

[13] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Mesh optimization. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH, pages 19–26, 1993.

[14] Alan D. Kalvin and Russel H. Taylor. Superfaces: Polygonal mesh simplification with bounded error. *IEEE Computer Graphics and Appl.*, 16(3), May 1996.

[15] Florent Lafarge and Clment Mallet. Creating large-scale city models from 3d-point clouds: A robust approach with hybrid representation. *International Journal of Computer Vision*, 99(1):69–85, 2012.

[16] Peter Lindstrom. Out-of-core simplification of large polygonal models. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 259–262, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[17] Peter Lindstrom and Greg Turk. Fast and memory efficient polygonal simplification. In *Proceedings of the Conference on Visualization '98*, VIS '98, pages 279–286, Los Alamitos, CA, USA, 1998. IEEE Computer Society Press.

[18] Peter Lindstrom and Greg Turk. Evaluation of memoryless simplification. *IEEE Transactions on Visualization and Computer Graphics*, 5(2):98–115, April 1999.

[19] David Luebke, Benjamin Watson, Jonathan D. Cohen, Martin Reddy, and Amitabh Varshney. *Level of Detail for 3D Graphics*. Morgan Kaufmann Editions, 2002.

[20] Martin Marinov and Leif Kobbelt. Automatic generation of structure preserving multiresolution models. *Comput. Graph. Forum*, 24(3):479–486, 2005.

[21] Ravish Mehra, Qingnan Zhou, Jeremy Long, Alla Sheffer, Amy Gooch, and Niloy J. Mitra. Abstraction of man-made shapes. In *ACM SIGGRAPH Asia 2009 Papers*, SIGGRAPH Asia '09, pages 137:1–137:10, New York, NY, USA, 2009. ACM.

[22] Niloy Mitra, Michael Wand, Hao (Richard) Zhang, Daniel Cohen-Or, Vladimir Kim, and Qi-Xing Huang. Structure-aware shape processing. In *SIGGRAPH Asia 2013 Courses*, SA '13, pages 1:1–1:20, New York, NY, USA, 2013. ACM.

[23] E. Ovreiu, J. G. Riveros Reyes, S. Valette, and R. Prost. Mesh simplification using a two-sided error minimization. In *International Conference on Image, Vision and Computing (ICIVC 2012)*, pages 26–30, 2012.

[24] Ruwen Schnabel, Patrick Degener, and Reinhard Klein. Completion and reconstruction with primitive shapes. *Computer Graphics Forum (Proc. of Eurographics)*, 28(2):503–512, March 2009.

[25] Ruwen Schnabel, Roland Wahl, and Reinhard Klein. Efficient RANSAC for point-cloud shape detection. *Computer Graphics Forum*, 26(2):214–226, June 2007.

[26] H-H. Vu, R. Keriven, P. Labatut, and J.-P Pons. Towards high-resolution large-scale multi-view stereo. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, Miami, Jun 2009.

[27] Steve Zelinka and Michael Garland. Permission grids: practical, error-bounded simplification. *ACM Transactions on Graphics*, 21(2):207–229, April 2002.

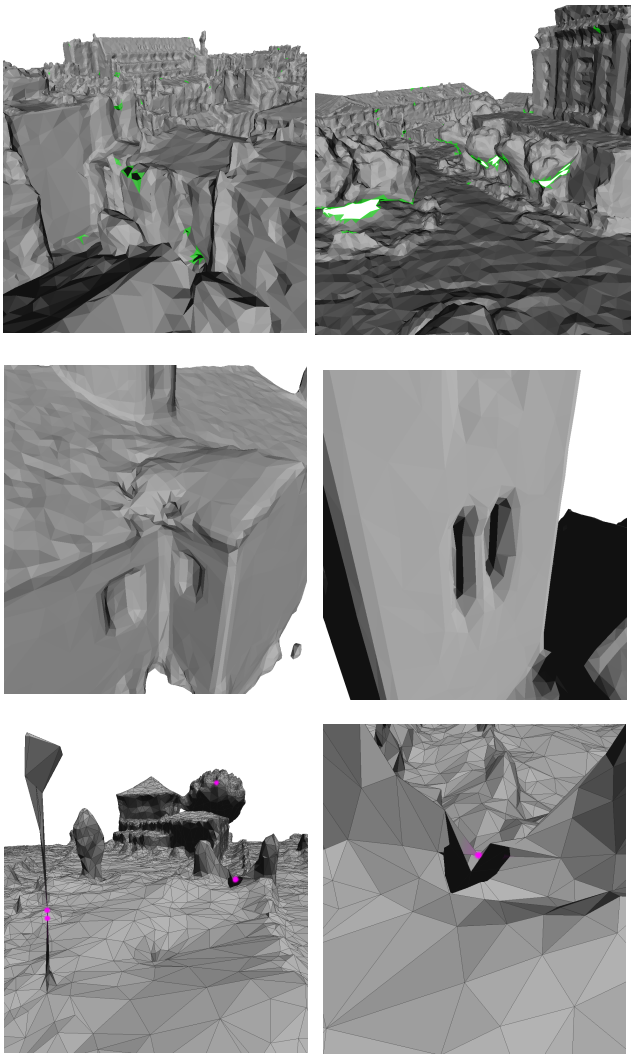

Figure 14: Robustness. Defects such as non-manifoldness, spurious handles or holes often arise in outputs of automated urban modeling pipelines. Top: holes due to occlusions. Middle: spurious handles. Bottom: non-manifold vertices.

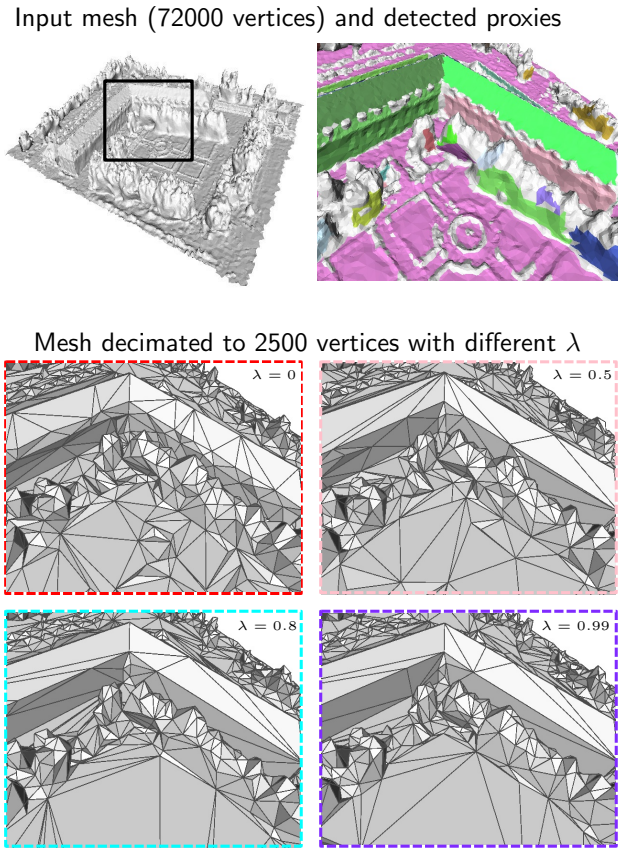

Figure 15: Impact of the abstraction parameter λ on the decimation. Top: input mesh and detected proxies. Bottom: meshes decimated with four different values for λ parameter. While the complexity of the output mesh is identical, larger values for λ imply faster removal of details on proxies during decimation and faster abstraction as evidenced by the straighter walls for high λ and the curves depicted by Figure 16.

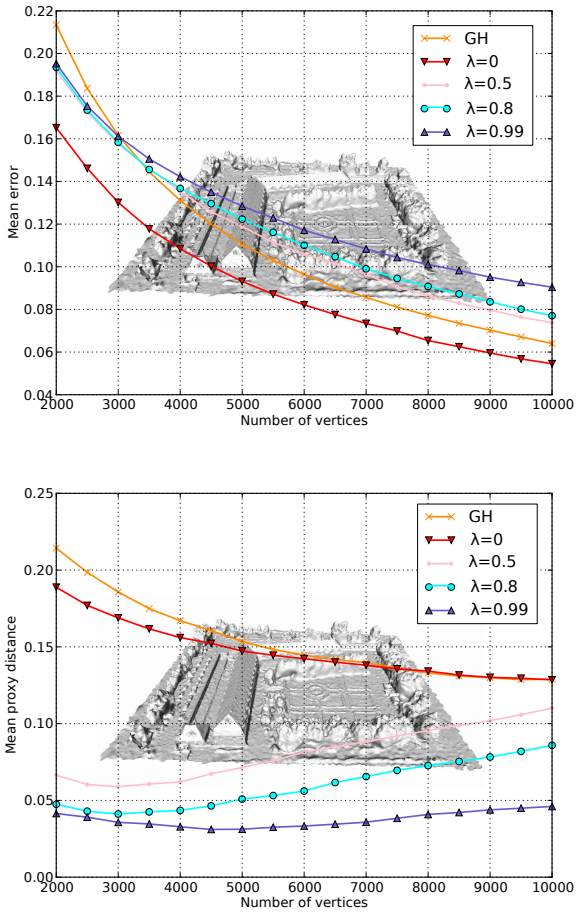

Figure 16: Complexity-distortion. We plot the mean error between the initial and approximated mesh (top) and the mean proxy distance (bottom) while decimating the mesh depicted by Figure 15, with four distinct values for the abstraction parameter λ. The mean proxy distance is computed as the average distance from vertices to their associated proxy. Higher values for λ yield lower proxy error but higher error for the approximation compared to the original mesh.

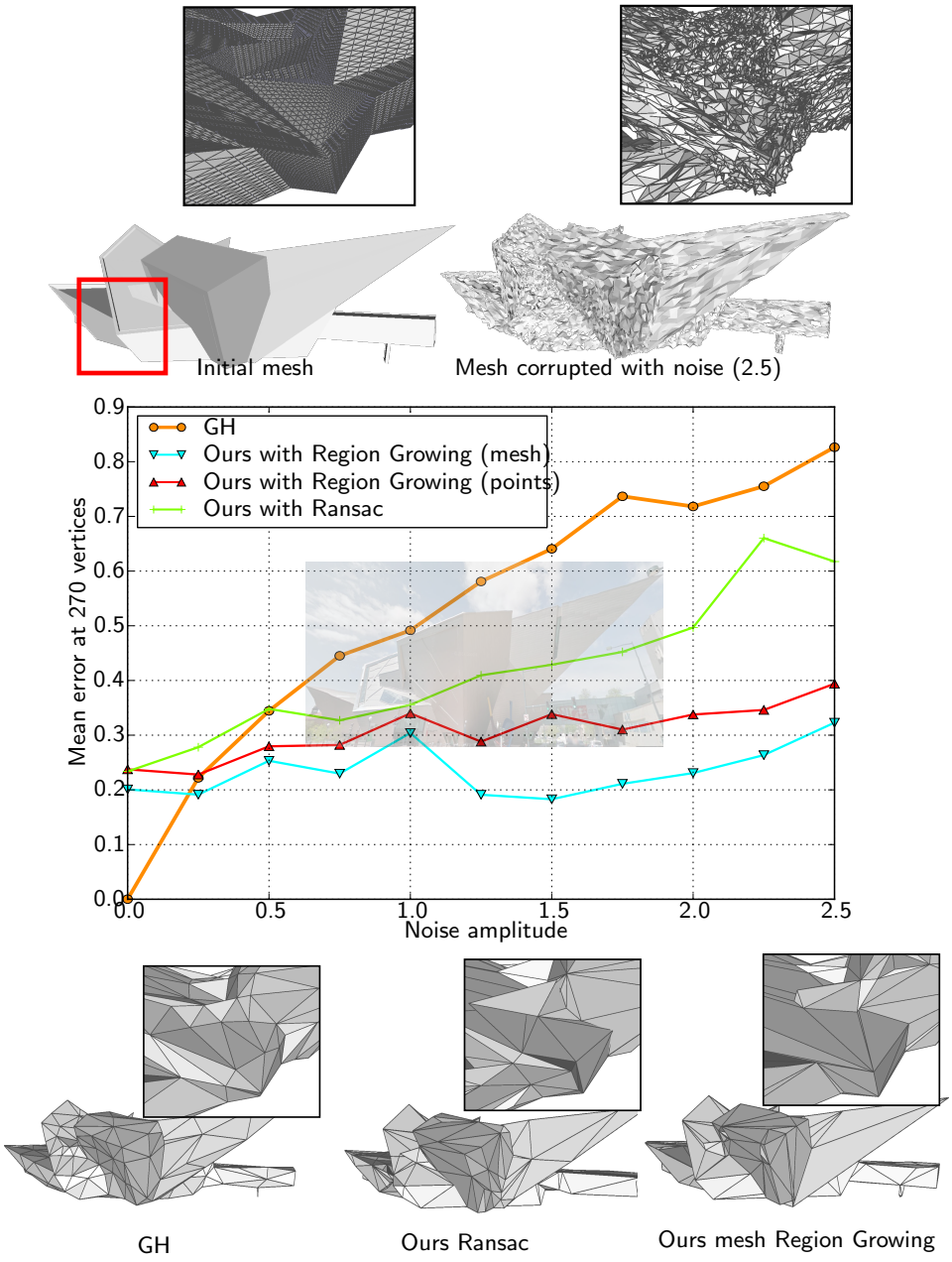

Figure 17: Noise robustness for three planar detection methods. A dense mesh of the Denver museum (upper left) is decimated after being corrupted by various levels of uniform noise. The curves plot the mean error against noise, measured after the input mesh model (40,000 vertices) is decimated to 270 vertices with Garland-Heckbert (GH) and our method. We combine our approach with (1) planar proxies detected via RANSAC, region-growing based on the mesh triangles and region-growing based on the mesh vertices (with a 10-nearest neighbors graph). The planar proxies are detected with the same parameters for all noise values. When the mesh is corrupted with noise, our algorithm with region growing yields a nearly uniform error in this range and lower than Garland-Heckbert. However, in absence of noise, our method generates an extra error mainly due to abstraction. RANSAC yields lower results that region-growing in this case, which is expected since this method is primarily designed to be outlier robust.

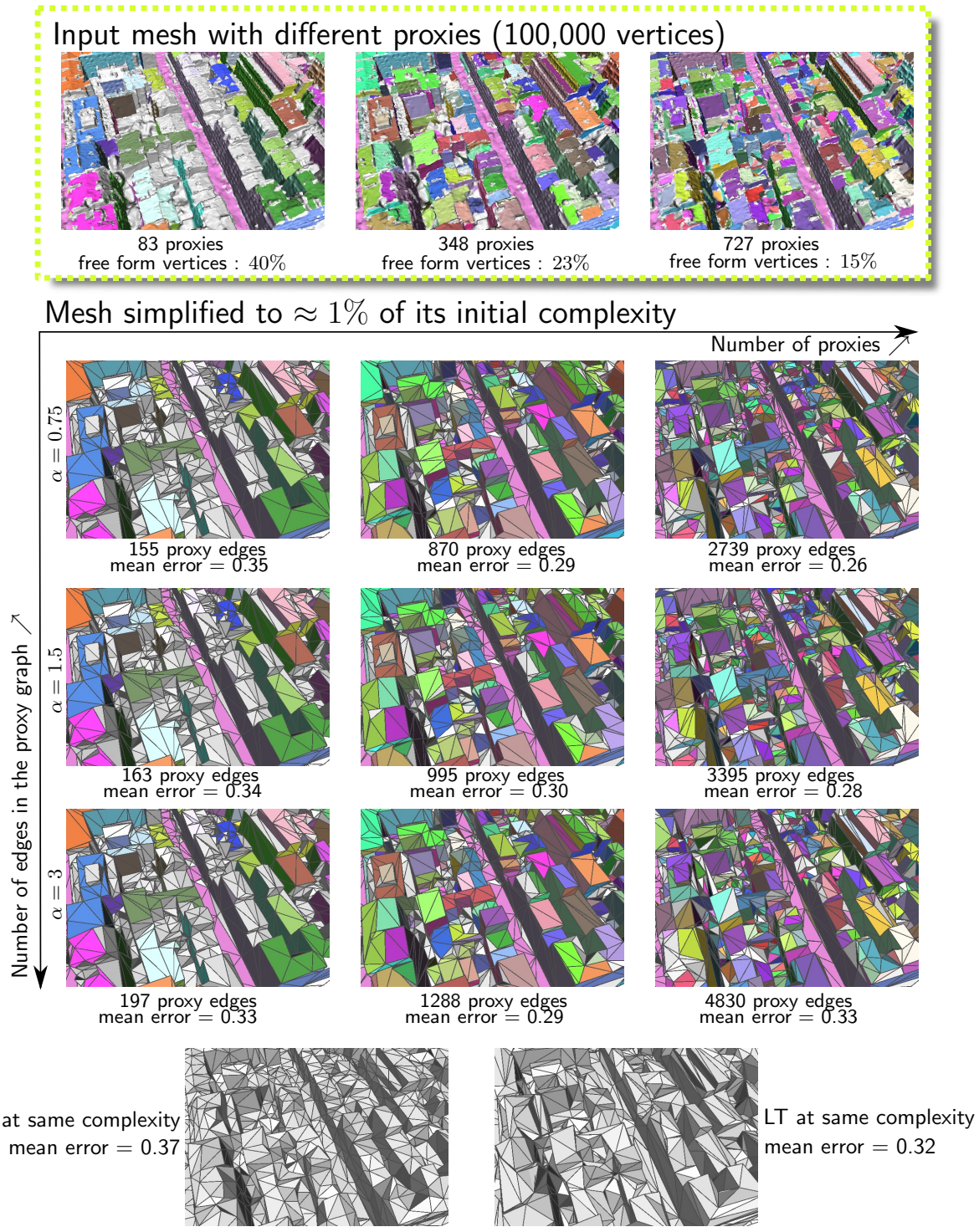

Figure 18: Parameters. We illustrate the impact of the number of proxies and graph distance threshold parameter α on the decimation. The input model is a part of the mesh shown Figure 19. Top line: different number of proxies detected. Decimated meshes are then shown by increasing number of proxies (from left to right) and increasing α from bottom to top. Observe that the quality of results (visual and objective) is stable with respect to α parameter and the number of pre-detected proxies.

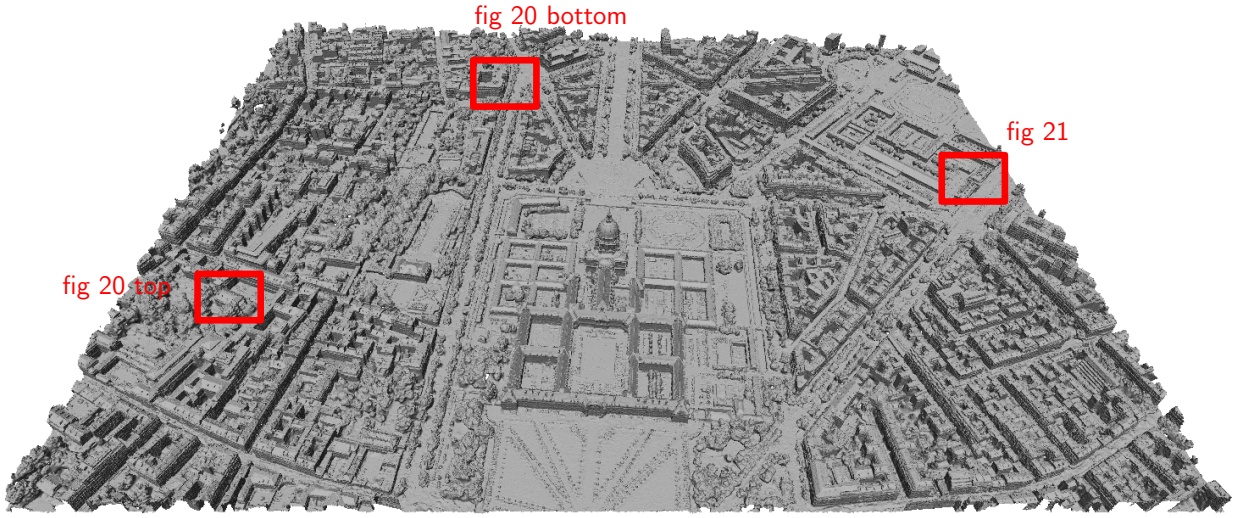

Figure 19: Paris. Input mesh generated through multi-view stereo reconstruction [1, 26]. The mesh, composed of 12 tiles, covers 1km^2 of Paris and comprises 6M vertices. Decimation results are depicted by Figure 20 and compared to one-step projection by Figure 21.

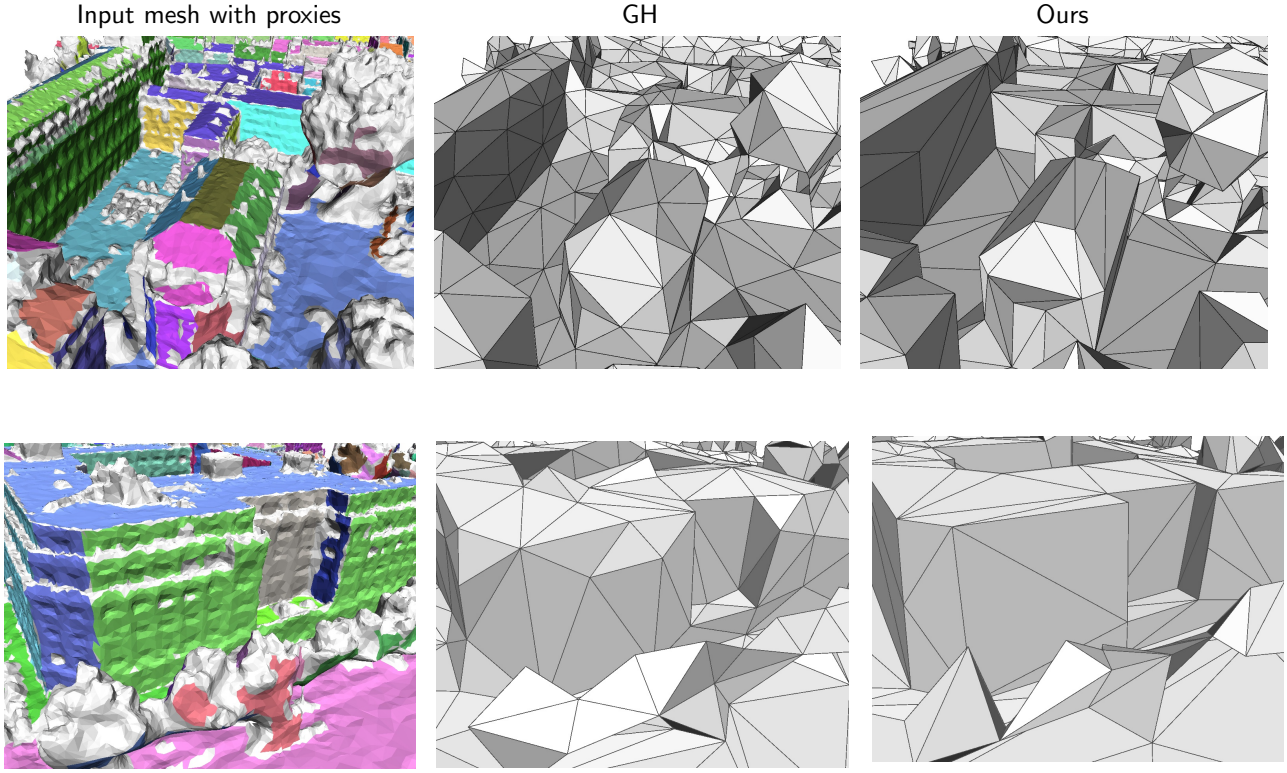

Figure 20: Comparison on Paris. The input mesh is depicted by Figure 19. From left to right: input mesh with color attributes related to the proxies, mesh decimated with Garland Heckbert (GH) and with our approach (complexities are identical). Adding an error metric relating to the scale of proxies tends to reinforce parallelism and co-planarity which further increases visual coherency.

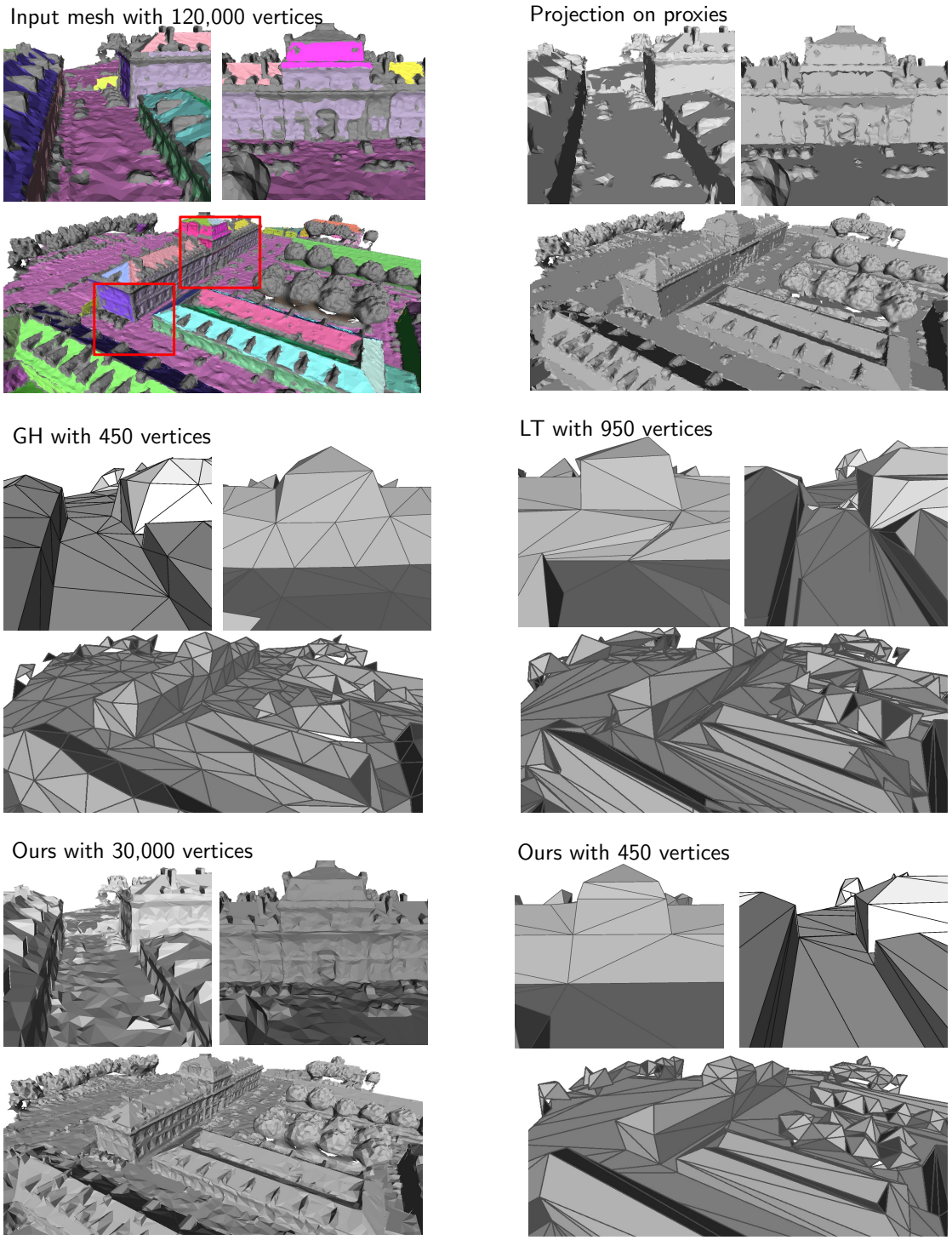

Figure 21: One-step projection and comparison on Paris. Top left: input mesh with colors associated to planar proxies. Top right: mesh with vertices projected in one step onto their proxies. Middle left: mesh decimated via Garland-Heckert (GH) to 450 vertices. Middle right: mesh decimated via Lindstrom-Turk (LT) to 950 vertices. Bottom: mesh decimated via our approach at 30,000 and 450 vertices. At coarse complexity our method compares well to previous work. The GH approach tends to preserve uniform density while ours favors the placement of vertices at corners together, the parallelism and the co-planarity. The LT approach generates high errors at very coarse complexity.

Input mesh	#vertices	#simplified vertices	#proxies	Decimation time (s)	Mean error	Method
Fandisk	$12,000$	81	41	2.9	0.0079	LT
				0.94	0.0064	GH
				10	0.0076	VSA
				5.3	0.0060	Ours
Chichen Itza	$392,000$	450	173	85	0.032	LT
				44	0.064	GH
				600	0.28	VSA
				250	0.034	Ours
Triumphal arch	$13,631$	115	70	2.9	0.11	LT
				0.65	0.33	GH
				5.3	0.042	Ours
Church	9,301	50	25	1.6	0.42	LT
				0.7	0.33	GH
				3.0	0.23	Ours
Paris	$5,927,613$	$30,742$	7,392	702	0.59	GH
				$2,990$	0.52	Ours

Table 1: Performance. We list the number of vertices of the input mesh, number of vertices of the decimated mesh, number of proxies detected, total decimation time and mean error. Results of VSA and LT are not shown for the last cases as they could not reach such low complexity.