LOD Generation for Urban Scenes
Yannick Verdie, Florent Lafarge, Pierre Alliez

To cite this version:
Yannick Verdie, Florent Lafarge, Pierre Alliez. LOD Generation for Urban Scenes. ACM Transactions on Graphics, 2015, 34 (3), pp.15. hal-01113078

HAL Id: hal-01113078
https://inria.hal.science/hal-01113078
Submitted on 4 Feb 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
LOD Generation for Urban Scenes

YANNICK VERDIE, FLORENT LAFARGE and PIERRE ALLIEZ
Inria Sophia Antipolis - Méditerranée

We introduce a novel approach that reconstructs 3D urban scenes in the form of levels of detail (LODs). Starting from raw data sets such as surface meshes generated by multi-view stereo systems, our algorithm proceeds in three main steps: classification, abstraction and reconstruction. From geometric attributes and a set of semantic rules combined with a Markov random field, we classify the scene into four meaningful classes. The abstraction step detects and regularizes planar structures on buildings, fits icons on trees, roofs and facades, and performs filtering and simplification for LOD generation. The abstracted data are then provided as input to the reconstruction step which generates watertight buildings through a min-cut formulation on a set of 3D arrangements. Our experiments on complex buildings and large scale urban scenes show that our approach generates meaningful LODs while being robust and scalable. By combining semantic segmentation and abstraction it also outperforms general mesh approximation approaches at preserving urban structures.

Categories and Subject Descriptors: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling

Additional Key Words and Phrases: urban reconstruction, levels of detail, abstraction, iconization, Markov random field, min-cut formulation, arrangement of planes.

1. INTRODUCTION

The quest for automated modeling of large scale urban scenes has received an increasing interest in recent years. A first class of approaches apply procedural modeling from grammatical rules and a fair amount of user interaction to generate detailed 3D models that are highly semantized [Vanegas et al. 2010]. Another class of approaches, referred to as urban reconstruction and focus of the present work, aims at the automated generation of accurate 3D models from physical measurements [Musialski et al. 2013].

The availability of massive airborne data sets at the scale of entire cities has stimulated research on automated methods for urban reconstruction. The quality of the reconstruction may be evaluated through visual inspection, faithfulness to the ground truth when available, or complexity/distortion tradeoff. While LIDAR scans have mostly been used during the last decade, recent advances on fully automated multi-view stereo (MVS) workflows [Acute3D 2014; Autodesk 2014; Pix4D 2014] allow the generation of complex surface triangle meshes, enriched with high resolution textures.

Airborne LIDAR scans exhibit high potential to reconstruct non-vertical elements such as roofs, but often fail to go beyond the 2.5D representation of urban environments. MVS meshes yield real 3D and unprecedented amount of details on vertical components such as facades. Surfaces derived from MVS workflows offer novel opportunities to generate LODs that are controllable via intuitive parameters, and meaningful for applications such as interactive navigation, urban planning, computational engineering and video games. Meaningful herein relates to LODs that are coherent across the entire scene, allow for incremental refinement, and provide some level of abstraction. This explains our motivation to go beyond simplification through semantic- and structure-aware reconstruction with LODs (see Fig.1). Note however that MVS meshes are in general less accurate than points generated by LIDAR acquisition systems at similar resolution. In particular, such meshes contain many geometric and topological defects (Fig.2) that require increased robustness for extracting semantic and structural information.

1.1 Related Work

Our review of previous work covers the four main facets of our problem statement: reconstruction, abstraction, LOD generation and semantic segmentation specific to urban scenes.

Urban reconstruction. In our setup reconstruction amounts to turn the raw input data into LODs of a 3D urban scene composed of watertight buildings in an environment composed of ground and trees. This problem being ill-posed, the state-of-the art ranges from interactive [Arikan et al. 2013] to automated [Poullis and You 2009] through semi-automated approaches [Sinha et al. 2008]. A possible taxonomy of the literature is to distinguish between two types of input data: depth maps and LIDAR point sets.

Depth maps are commonly generated from MVS images. The approaches proposed for generating compact 3D-models of buildings from depth maps proceed, e.g., in 2D through space partitioning [Zebedin et al. 2008] or in 3D through assemblies of cuboids generated by Monte Carlo sampling [Lafarge et al. 2010]. Some
approaches combine both ground-based and aerial data to generate more complete representations of urban scenes [Frueh and Zakhor 2003]. In addition to being often hampered with high noise, a major limitation of depth maps is that they prevent distinguishing buildings from high vegetation. LIDAR point set data became popular from the mid-2000 mostly for their accuracy, despite the fact that they are geometrically less structured than depth maps and do not contain any radiometric information. Urban LIDAR data stimulated a series of work mainly focused on parsing building components and extracting building contours.

For LIDAR as well as for depth map data, a popular methodology consists of relying upon 3D planar primitives for roofs and facades [Poullis and You 2009; Lafarge and Mallet 2012], with advances on parsing planes [Toshev et al. 2010] and discovering global regularities among planes [Zhou and Neumann 2012]. The Manhattan World assumption [Coughlan and Yuille 2000] constrains planes to follow only three orthogonal directions. This assumption reduces the solution space to explore as well as the geometry of 3D models [Vanegas et al. 2012]. Both airborne LIDAR scans and depth maps only provide 2.5D representations that prevent modeling the geometry of vertical components.

Some approaches address the urban reconstruction problem by inverse procedural modeling. Based on a grammar and related semantic rules, forward procedural modeling has no equivalent in terms of control over geometric complexity, structure and semantic [Bao et al. 2013]. Impressive results are obtained at the street-view level [Teboul et al. 2010; Martinovic et al. 2012; Riemenschneider et al. 2012]. However, inverse procedural modeling applied to airborne measurement data is still an open problem: state-of-the-art approaches rely on simple grammars and require assumptions such as axis-aligned geometry that do not match our objective [Vanegas et al. 2010].

Abstraction. One step toward an improved control over complexity and structure is a process referred to as abstraction. The latter creates recognizable visual depictions of known objects through compact descriptions involving a handful of characteristic primitives such as curves [Mehra et al. 2009], icons or solids. Abstraction thus goes well beyond shape simplification [Garland and Heckbert 1997] and mesh repairing [Ju 2004] as involves filtering, smoothing, and reinforcing the regular structures. Abstraction is also related to the problem of structure discovery of a scene [Mitra et al. 2013] or of an entire collection of objects [Yumer and Kara 2012]. Abstraction perfectly matches our objective to generate compact descriptions in the form of LODs.

LOD Generation. Managing LODs is a common topic in geometric modeling [Luebke et al. 2002], but is less common in urban reconstruction [Arefi et al. 2008]. General mesh simplification or approximation approaches are effective but often merge objects of different classes (e.g., a tree and a roof) and fragment structural features such as the boundary of a roof. Most of these approaches rely on a pure geometric error metric and are thus oblivious to semantics and structure of urban scenes. Some error metrics are more feature-preserving than others, which indirectly helps preserve the structure, but the structure itself is scale-dependent and hence can hardly be decoupled from semantic labels specific to urban LODs.

Semantic segmentation. Techniques for attempting to bridge the semantic gap have been proposed [Falcidieno and Spagnuolo 1998]. They range from annotation to learning, with the usual dilemmas between interactive vs automated approaches, and supervised vs unsupervised learning. The automated segmentation of surface meshes into parts, too general for our setup, has been well explored [Attene et al. 2006; Shamir 2008; Chen et al. 2009], with clustering or learning [Kalogerakis et al. 2010] as favored methodology. The classification of airborne LIDAR urban scans is also an active research topic [Rottensteiner et al. 2012], with three common classes: building, vegetation and ground. Existing approaches are however not directly applicable to our problem as our input data contain more noise and no additional properties such as echo number or signal magnitude which help distinguishing the classes of interest. Lin et al. [2013] decompose the elements of residential buildings through supervised learning, with however no abstraction nor LODs. In addition, our setup differs as targets unsupervised learning and dense urban scenes with global regularities.

In summary, and in spite of the variety of methods currently available to address each facet of our problem individually, there is a dire need for an automated reconstruction and LOD genera-
1.2 Positioning and Contributions

In our framework the input data are raw triangle surface meshes, typically generated from multi-view stereo workflows. These meshes are not required to be manifold or watertight. The main objects of interest are buildings and trees, and the structure to discover corresponds to the LODs defined by CityGML, [Groger and Plumer 2012], see Fig. 3. A given LOD describes both the lower LODs and additional details enriching its structure and geometry. Generating LODs is a crucial advantage when targeting a large range of urban applications.

An important methodological choice when designing a complete urban reconstruction pipeline is the way to associate semantics and geometry. The recent approaches [Haene et al. 2013; Lin et al. 2013] that deal with semantics and geometry simultaneously yield elegant formulations but are not scalable. We instead follow a sequential approach (semantics-then-geometry) similar to [Poullis and You 2009; Lafarge and Mallet 2012]. The underlying idea consists in first extracting semantic classes via classification so that (i) the subsequent geometric reconstruction is adapted to each class of interest, and (ii) computational complexity is reasonable even at city scale. Note that existing approaches designed for airborne LIDAR [Poullis and You 2009; Lafarge and Mallet 2012] are not directly applicable to our input data because (i) MVS meshes require high robustness to deal with geometric and topological defects, and (ii) 2.5D shape arrangement processes cannot generate multiple coherent LODs with real 3D.

Our pipeline proceeds with three main steps: semantic-based segmentation of input meshes (§2), abstraction of urban objects via icon extraction and filtering of planar proxies (§3), and reconstruction at four different LODs (§4), see Fig.4.

Our main contributions are as follows:

—A fully automated reconstruction pipeline that departs from existing work by the ability to (i) generate multiple coherent LODs, and (ii) take as input raw surface meshes generated by multi-view stereo workflows;

—Three new technical ingredients that yield robustness to input mesh defects, scalability and efficiency: (i) a feature-preserving Markov Random Field used for classification, (ii) a greedy process for the global regularization of planes with a hierarchical organization of the canonical geometric relationships, and (iii) a min-cut formulation applied to a discrete approximation of a 3D planar arrangement for robust reconstruction.

2. CLASSIFICATION

The classification step relies on a Markov Random Field (MRF) in order to distinguish between four classes of urban objects: ground, tree, facade and roof. As the classification is unsupervised we rely solely on geometric attributes using the following rationale: (i) ground is characterized by locally planar surfaces located below the other classes, (ii) trees have curved surfaces, (iii) facades are vertical surfaces adjacent to roofs and (iv) roof are mostly composed of piecewise-planar surfaces.

2.1 Superfacet Clustering

As the input meshes are very dense, classifying each triangle facet through the MRF would lead to unpractical computation times. In a pre-processing step we thus over-segment the input mesh into superfacades: sets of connected triangle facets, similar in spirit to the notion of superpixels used for image analysis. Superfacades are obtained by clustering, through region growing, the triangle facets with similar shape operator matrices. More specifically, we estimate the shape operator matrix [Cohen-Steiner and Morvan 2003] for each triangle facet on a local spherical mesh neighborhood of radius $R_m$, and compare during clustering these matrices via the Frobenius norm. Growing is effective when this distance remains inferior to to a limit value $d_c$. Fig.5(left) depicts how this clustering procedure identifies the nearly planar components and preserves the sharp features.

2.2 Geometric Attributes

Three geometric attributes are computed for each triangle facet $f_i$ of the input mesh:

—The elevation attribute $a_e$ is defined as a function of the relative height ($z$ coordinate) of the triangle facet centroid, denoted by $z_i$:

$$a_e(f_i) = \sqrt{\frac{z_i - z_{\min}}{z_{\max} - z_{\min}}},$$

where $(z_{\min}; z_{\max})$ denotes the height range of all triangle facet centroids located within a local spatial neighborhood. The square root ensures that small values of relative height get a larger elevation attribute. The size of the neighborhood, set by default to 40 yards, must be sufficiently large to meet ground components and sufficiently small to gain resilience to hilly environments.

—The planarity attribute $a_p$ denotes the planarity of the superfacet containing $f_i$, derived from the so-called surface variation [Pauly...
et al. 2002]:
\[ a_h(f_i) = 1 - \frac{3 \lambda_0}{\lambda_0 + \lambda_1 + \lambda_2}, \] (2)
where \( \lambda_0 \) denotes the minimum eigenvalues of the covariance matrix computed in closed form over all triangle facets of the superfacet containing \( f_i \). Each eigenvalue measures the variance of the superfacet along the corresponding eigenvector. The variation measures how much the superfacet deviates from the local tangent plane: the planarity is thus 1 for a perfectly planar superfacet, and 0 for an isotropic superfacet with three identical eigenvalues.

—The **horizontality** attribute \( a_h \) measures the deviation of the unit normal \( n_i \) to triangle facet \( f_i \) with respect to the vertical axis:
\[ a_h(f_i) = |n_i \cdot n_z|, \] (3)
where \( n_z \) denotes a unit vector along the Z coordinate axis.

From these geometric attributes defined for each triangle facet, all taking values within \([0, 1]\), we compute the geometric attribute for each superfacet as the area-weighted sum of the geometric attributes of its triangle facets. We compute similarly the normals of superfacets. Figure 5 illustrates the superfacet clustering and geometric attributes on a part of a urban scene.

### 2.3 Markov Random Field

From the geometric attributes computed per superfacet, a Markov Random Field is used to label each superfacet with one of the four classes: \( \{ \text{ground, tree, facade, roof}\} \). The defects of the raw input mesh require a regularized global optimization process offered by MRF which adds contextual as well as spatial consistency to the classification. More specifically, we use a MRF with pairwise superfacet interactions. The quality of a label configuration \( I \) is measured by energy \( U \):
\[ U(I) = \sum_{i \in S} D_i(l_i) + \gamma \sum_{(i,j) \in E} V_{ij}(l_i, l_j) \] (4)
where \( D_i \) and \( V_{ij} \) denote the unary data term and propagation constraints respectively, balanced by parameter \( \gamma > 0 \). \( S \) denotes the set of superfacets. \( E \) denotes all pairs of adjacent superfacets, two superfacets being adjacent if they share at least one edge in the input mesh. The data term combines the above-described attributes weighted by the area \( A_i \) of the superfacet \( i \):
\[ D_i(l_i) = A_i \times \begin{cases} 1 - a_p \cdot a_h \cdot \pi & \text{if } l_i = \text{ground} \\ 1 - a_p \cdot a_h & \text{if } l_i = \text{tree} \\ 1 - a_p \cdot a_h \cdot a_e & \text{if } l_i = \text{facade} \\ 1 - a_p \cdot a_h \cdot a_e & \text{if } l_i = \text{roof} \end{cases} \] (5)
where \( \pi = 1 - a \). The pairwise interaction \( V_{ij} \) between two adjacent superfacets \( i \) and \( j \) favors label smoothness away from sharp creases:
\[ V_{ij}(l_i, l_j) = C_{ij} \cdot w_{ij} \cdot 1_{[l_i \neq l_j]}, \] (6)
where \( 1_{[l_i \neq l_j]} \) denotes the characteristic function, and \( C_{ij} \) denotes the length of the interface between superfacets \( i \) and \( j \) (sum of interface edge lengths). Weight \( w_{ij} \) is introduced to lower the label propagation over sharp creases that often appear when two classes meet (e.g., for trees adjacent to facades, see Fig. 6). \( w_{ij} \) is defined as the angle cosine between the estimated normals of two superfacets. As the unary data term and pairwise potential are weighted by the superfacet areas and interface lengths, this energy formulation behaves similarly to a triangle facet-based energy with grouping constraints. An approximate solution to this energy minimization problem is solved through the \( \alpha - \beta \) swap algorithm [Boykov et al. 2001].

### 2.4 Semantic Rules

The aforementioned geometric rationale alone is not sufficient to solve the ill-posed classification problem. Two types of errors frequently occur when dealing with complex urban scenes: (i) roof superstructures such as chimneys or dormer-widows may be wrongly labeled as tree, these elements being too small and irregular to be considered locally planar, and (ii) vertical components of large trees may be labeled as facade. We thus add the following semantic rules:

—**Rule 1.** superfacets labeled as tree and adjacent to only superfacets labeled as roof are re-labeled roof. This rule relies on the common assumption that large trees are not located on top of roofs.

—**Rule 2.** superfacets labeled as facade and adjacent to superfacets labeled as tree and ground are turned to tree.

---

Fig. 5. Superfacet clustering and geometric attributes used for classification. Superfacet clustering produces nearly planar patches that preserve sharp creases and reduce the complexity of the classification problem. The three considered geometric attributes provide distinct and complementary information to classify the urban object of interest.
3. ABSTRACTION

From the superfacets classified in §2 the abstraction step creates compact descriptions involving characteristic icons and planar proxies. Compact descriptions are obtained by geometric and structure simplification through planar shape approximation and regularization, iconization and LOD-based filtering. Remind that superfacets are classified into four classes: ground, tree, facade and roof, the latter containing roof superstructures. A key idea behind our approach is to specialize the abstraction and reconstruction steps to these classes as well as to the LODs, such that, e.g., trees are represented by icons instead of attempting to reconstruct them with planar proxies. In addition, regularization is our means to improve scalability and robustness of the reconstruction step.

The classes are abstracted as follows:

—Ground is represented by a 2D Delaunay triangulation lifted in 3D with a natural neighbor interpolation of the maximum elevation attribute of the input mesh superfacets labeled as ground (see §2).

—Facades and roofs are approximated by a set of planar proxies with reinforced regularities, these proxies being used as input to the final watertight reconstruction step (§4). We restrict ourselves to planar proxies as planar surfaces cover on average 80% of urban areas, and are amenable to effective abstraction and reconstruction for LOD generation.

—Roof superstructures, facade components and trees are abstracted through iconization on distinct depth maps. We do not approximate roof superstructures and facade components with planar proxies due to the limited resolution of airborne MVS meshes (typically as only a few triangle facets for a chimney or a window). For a similar reason there is no specific superstructure class, and, e.g., chimneys are re-labeled from tree to roof during classification (see §2). Tree icons are parametric shapes invariant by rotation along Z-axis, similar to [Verdie and Lafarge 2014].

The icons and regularized proxies are then filtered through LOD generation, before reconstruction.

3.1 Planar Proxies

We first identify a set of nearly planar superfacets by selecting the ones labeled as roof or facade, with a high planarity attribute $a_p$ and a minimum large area ($w$ impose $a_p > t_p$ and area $A$ larger than $A_{\text{min}}$, where $t_p$ and $A_{\text{min}}$ are two model parameters). For each near-planar superfacet we compute its least-squares fitting plane, referred to as superfacet proxy. We then improve globally the regularity of these proxies by altering their orientation and position so as to reinforce their canonical geometric relationships. Departing from other approaches such as GlobFit [Li et al. 2011] or LIDAR-specific algorithms [Zhou and Neumann 2012], we adopt a detection-then-regularization strategy with a single iteration in order to favor scalability and low running times. For instance, less than one second is required to regularize proxies shown in Fig.9 instead of ten minutes for GlobFit. The main idea is to organize the geometric relationships hierarchically then regularize the proxies.
in one step via a greedy process. In addition, we introduce a novel Z-symmetry geometric relationship relevant for abstracting building roofs.

**Geometric relationships.** Denote by $P_1$ and $P_2$, two proxies with respective unit normals $n_1$ and $n_2$, and centroids $c_1$ and $c_2$. We define four canonical relationships under an orientation tolerance $\epsilon$ and an Euclidean distance tolerance $d$:

- **Parallelism.** $P_1$ and $P_2$ are $\epsilon$-parallel if $|n_1 \cdot n_2| \geq 1 - \epsilon$;
- **Orthogonality.** $P_1$ and $P_2$ are $\epsilon$-orthogonal if $|n_1 \cdot n_2| \leq \epsilon$;
- **Z-symmetry.** $P_1$ and $P_2$ are $\epsilon$-Z-symmetric if $|n_1 \cdot n_1| - |n_2 \cdot n_2| \leq \epsilon$, where $n_z$ is the unit vector along the vertical axis;
- **Coplanarity.** $P_1$ and $P_2$ are $\epsilon$-coplanar if they are $\epsilon$-parallel and $|d_\perp(c_1, P_2) + d_\perp(c_2, P_1)| < 2d$, where $d_\perp(c, P)$ denotes the orthogonal distance between point $c$ and proxy $P$.

The first three relationships are related to the proxy orientations, and coplanarity is a specific instance of parallelism with an additional relative positioning constraint. The notion of Z-symmetry matches the common assumption that connected components of roofs tend to share similar slope values. For our urban context the rotational symmetry [Li et al. 2011; Zhou and Neumann 2012] is too general. In addition, Z-symmetry is detected with linear operations while rotational symmetry involves a quadratic complexity.

**Detection of regularities.** Global regularities are detected via a hierarchical decomposition. Parallelism relationships form the reference layer - referred to as layer 1 - of this hierarchy: we cluster the proxies which are $\epsilon$-parallel into parallel clusters, and compute the average orientation of each cluster. The upper layer - referred to as layer 2 - is formed by detecting orthogonality and Z-symmetry relationships among the parallel clusters. An orthogonality graph is constructed with one node per parallel cluster, and one edge between two nodes that are $\epsilon$-orthogonal. We also cluster the parallel clusters which are $\epsilon$-Z-symmetric into Z-symmetric groups, and compute the average angle of each group with respect to the Z-axis. A lower layer - referred to as layer 0 - is finally created via the coplanarity relationship: we decompose each parallel cluster into sets of proxies that are $\epsilon$-coplanar, and compute the average centroid for each set of coplanar proxies. Note that all aforementioned averaging processes are weighted by the area of each proxy, the area of each proxy being defined by the total area of its associated superfacets. Such hierarchical organization is efficient as we avoid performing a costly pairwise analysis of the different relationships among proxies. Fig.9 illustrates the hierarchical organization used to detect regularities.

The regularization step operates on clusters of proxies, and comprises two steps: re-orientation and re-positioning.

**Re-orientation.** Parallel clusters are re-oriented through propagating geometric relationships into the orthogonality graph. Denote respectively by source and target node a pair of nodes altered by the propagation. The initial orientation of the target node is altered by constraining its normal to match the relationships of (i) orthogonality with respect to the source node, and (ii) Z-symmetry if the node has been clustered into a Z-symmetric group. We distinguish between three cases:

- **Case A:** both relationships are active. There is in general a unique orientation that satisfies both relationships. It may occur that no solution exists due to contradicting relationships: the target node is then not re-oriented.
- **Case B:** only the orthogonality relationship is active. We re-orient the target node in accordance to the orientation orthogonal to the source node in the hierarchy, that best aligns to its initial orientation.
- **Case C:** only the Z-symmetry relationship is active. This occurs when a node has no parent in the hierarchy, i.e., when it is the root node of the orthogonality graph. We re-orient the target node in accordance to the orientation satisfying the Z-symmetry constraint that best aligns to its initial orientation.

These three cases are illustrated by Fig.10 with a representation on the unit sphere. To prevent from large deviations, we perform a re-orientation only when the dot product between the initial and the re-oriented normals is lower than $1 - \epsilon$. The greedy propagation in the orthogonality graph proceeds from large to small nodes, the size of a node being defined by the total area of its proxies. Such loop-free propagation tends to reduce contradictions between relationships and gives more confidence to the larger nodes.

**Re-positioning.** For each set of coplanar proxies, the centroid of each proxy is translated along the line supporting the proxy normal.
so that the average centroid of the coplanar set is contained within the proxy plane.

### 3.2 Iconization

The iconization step is devised to abstract trees, roof superstructures and facade elements. For these three types of icons, we adopt a two-step strategy illustrated in Fig.11. Elements of interest are first located from depth maps, and then abstracted by 3D icons that are fitted to the input mesh. Icons of trees are used at the four LODs, whereas roof superstructure and facade icons are only used at LOD3.

For trees we first construct a depth map by rasterizing in an image the input mesh in the XY coordinate plane and taking as height value the maximum elevation attribute restricted to superfacets labeled as tree. We greedily extract the local maxima of this map by a watershed algorithm in order to locate the center of each tree icon in the XY plane, and fit the best half ellipsoid to the map while keeping the center fixed, similar to [Lafarge and Mallet 2012].

For roof superstructures, which mainly correspond to chimneys, dormer-windows and small roof extensions, we construct a depth map by rasterizing in an image the input mesh in the XY coordinate plane and taking as height value the maximum elevation attribute restricted to superfacets labeled as roof. We then classify it as either a door or a window. A grid element is classified as a door when it is located at the bottom row of the grid and contains a minimum average depth value (set to 10cm by default). All other occupied elements are classified as windows.

### 3.3 LOD Generation

The LOD generation step proceeds by filtering the regularized proxies and abstracting the icons, in accordance to the urban LODs used by CityGML:

- **LOD0**: the ground mesh is not used as the representation is planar. Trees are depicted as discs computed as vertical projection of tree icons, and buildings are depicted by 2D regions bounded by polylines computed only from the abstracted proxies labeled as facade using a 2D instance of the min-cut formulation (§4). Superstructures are omitted.

- **LOD1**: ground mesh, enriched with vertical cylinders for trees and a LOD0-building elevated in 3D with horizontal proxies as roofs whose height is defined as the median of associated superfacet heights.

- **LOD2**: ground mesh enriched with tree icons and buildings reconstructed (§4) with all proxies to generate piecewise-planar roofs.

- **LOD3**: LOD2 enriched with roof superstructures and facade elements.

Transactions on Graphics, Vol. VV, No. N, Article XXX, Publication date: Month YYYY.
4. RECONSTRUCTION

The final reconstruction step turns the proxies regularized and filtered in previous step into watertight buildings. For each connected component of buildings identified in §2, a 3D arrangement of planes provides us with a means to assemble the planar proxies into well-behaved surfaces; watertight and free of self-intersection. When combined with global regularization, LOD filtering and min-cut, it furthermore generates lightweight polygon meshes that preserve the structural components of the scene at the chosen LOD, and completes missing parts of the scene in a plausible manner.

4.1 Discrete 3D Arrangements

Even when restricting it to each building component, computing the complete, exact arrangement leads to very high computational complexity (we experimented with scenes containing hundreds of building components, each containing on average hundred planes). Previous work based on arrangements attempted to reduce complexity by restricting the arrangements to axis-aligned planes [Purukawa et al. 2009], creating multi-layers of 2D arrangements of lines [Oesau et al. 2014] or computing a two-level hierarchy made up of a rectilinear volumetric grid combined with a convex polyhedral cell decomposition [Chauve et al. 2010]. The approaches are either too restrictive or algorithmically too complex, exceeding half an hour when dealing with only few hundred planes. Observing that only a very small subset of the faces of the arrangement contribute to the output after solving for a min-cut surface, we postpone the exact geometric computation operations to the final surface extraction step after min-cut solve. We rely instead on a transient discrete approximation of the arrangement so as to avoid the compute-intensive exact geometric operations required to insert each plane into the arrangement.

For each subset of the input mesh associated to a building component we first compute an object-oriented bounding box $B$. We then sample uniformly $B$ by placing anchors points at all corners of a uniform grid aligned to $B$. Each of these anchors is enriched with two attributes:

—A binary flag that specifies whether the anchor is estimated to be inside or outside the inferred building. This flag is guessed by casting rays and counting the intersection parity of these rays against the input mesh. If the number of rays with odd (resp. even) intersections is higher, an inside (resp. outside) flag is assigned to the anchor. Five rays have shown sufficient in all experiments: four towards the upper corners of $B$ and one towards the barycenter of these corners. For input meshes highly corrupted with spurious holes, a more global approach [Zhou et al. 2008] would be better suited.

—An integer that denotes the index of the cell of the planar arrangement containing the anchor.

Instead of computing the exact geometry of the arrangement cells the approximate arrangement of planes is constructed solely via an arrangement binary tree and the anchor cell indices. More specifically, we greedily insert the $6$ planes of $B$, then the planar proxies, while refining the binary tree in which each node refers to a cell. The anchor cell indices are updated after each plane insertion: this operation corresponds to the insertion of a new layer of nodes into the binary tree.

The anchors are also used to compute via quadratures approximate geometric attributes (volume of cells, area of facets) that are required by the subsequent min-cut formulation. Fig.12 depicts a set of anchors at work for surface reconstruction.

4.2 Min-Cut Formulation

For each arrangement a min-cut formulation is used to find an inside/outside labeling of the cells, the output surface being defined as the interface facets between inside and outside. Consider a graph $(C,F)$ where $C = \{c_1, \ldots, c_n\}$ denotes the nodes relating to the cells induced by the space partition, and $F = \{f_1, \ldots, f_m\}$ denotes edges relating to the facets separating all pairs of adjacent cells. A cut in the graph consists of separating the cells $C$ into two disjoint sets $C_{in}$ and $C_{out}$. The edges between $C_{in}$ and $C_{out}$ correspond to a set of facets forming a surface $S \subset F$.

In order to quantize the quality of the solution, i.e., the surface $S$ induced by the cut $(C_{in}, C_{out})$, we introduce the following cost function $C$:

$$ C(S) = \sum_{c_k \in C_{out}} V_{c_k} \, g(c_k) + \sum_{c_k \in C_{in}} V_{c_k} \, (1 - g(c_k)) + \beta \sum_{f_i \in S} A_{f_i}, $$

(7)

where $V_{c_k}$ denotes the volume of cell $c_k$, $g(c_k)$ denotes the function estimating the label likelihood of cell $c_k$ with respect to the ratio of its inside/outside anchors, and $A_{f_i}$ denotes the discrete area of facet $f_i$. The first two terms of the cost function $C$ are data terms whereas the third term weighted by parameter $\beta \geq 0$ acts as a regularization term in order to favor solutions with small area. The optimal cut minimizing the cost $C(S)$ is found via the max-flow algorithm [Boykov and Kolmogorov 2004].

Function $g(c_k)$, defined in the interval $[0,1]$, quantizes the coherence of assigning label inside to cell $c_k$ with ratio $r_{in}$ of inside anchors contained in $c_k$:

$$ g(c_k) = \left(2r_{in} - 1 \right) \times \frac{\left|2r_{in} - 1\right| + 1}{2}, $$

(8)
Choosing \( g \), as illustrated by Fig. 13.

\[ g(r_{an}) = \begin{cases} 
\alpha & \text{if } r_{an} > 0 \\
0 & \text{otherwise}
\end{cases} \]

where \( \alpha \) denotes a parameter tuning the data sensitivity of function \( g \), as illustrated by Fig.13.

The optimal cut corresponds to a subset of facets separating the inside and outside cells, as depicted by Fig.12. The final geometry of these interface facets is then computed with exact arithmetic by intersecting the set of corresponding planes from the binary tree. By construction each interface facet is thus a planar convex polygon. For LOD0 and LOD1 we create a 2D instance of such discrete arrangement and min-cut formulation by sampling a single horizontal layer of anchors. In some rare cases a cell is free of anchors. Such a cell, referred to as an empty cell, is not taken into account in the cost function \( C \) as its discrete volume is null. This situation may arise when the cell is thin compared to the anchor spacing. The anchor spacing must also be not too short in order to reach practical performance and memory consumption. Empty cells are rare in practice thanks to the plane regularization process which limits the number of thin cells, in particular when we constrain the anchor spacing to be lower than the Euclidean distance tolerance \( d \), as illustrated by Fig. 14. In all experiments shown the default value for the anchor spacing is set to 0.5 yards.

5. RESULTS

Our algorithm is implemented in C++ using the CGAL library, an \( \alpha - \beta \) swap and a max-flow library [Boykov and Kolmogorov 2004; Boykov et al. 2001]. All timings are measured on an Intel Core i7 clocked at 2GHz. We experiment with real-world meshes generated by state-of-the-art multi-view stereo workflows (Acute3D Smart3DCapture and Autodesk 123DCatch) as well as with defect-free meshes used to evaluate robustness and accuracy. The parameters of the algorithm and their default values are summarized in Tab. I. The number of parameters is large, but this is the price to pay for such a complete system combining semantic segmentation, abstraction and LOD reconstruction of urban scenes in an unsupervised manner. Note also that the default values of these parameters, used for all shown experiments, are stable on a large range of inputs. Fig.15 illustrates our algorithm at work on various types of urban scenes ranging from residential houses to dense urban blocks through architectural buildings. The reconstructions match our initial goal to generate meaningful levels of details: the semantic labels are recovered, the structures are preserved and the details are coherent across the scene.

Scalability and Performances. Our pipeline digests input meshes with several million triangle facets. On average a block of buildings is fully processed in around 30 seconds for LOD1 and 3 minutes for LOD2. Fig.16 depicts a variable density urban scene covering 1km square of Paris with 235 building components, 3.3K roofs and 1.3K trees. For this complex model the total running time is less than 20 minutes for LOD1 and around 2 hours for LOD2 (175K facets), with a sequential implementation of the plane arrangement per block. The LOD-reconstructions are the most time-consuming operations, in particular the LOD2-reconstruction where the plane arrangement is performed in 3D with all the proxies. LOD3 is not shown because superstructures are not observable at this scale. Tab.II lists some models and associated

Transactions on Graphics, Vol. VV, No. N, Article XXX, Publication date: Month YYYY.
Fig. 15. Reconstruction and LOD Generation. First row: on this simple residential scene all facades and roofs are well classified and the Z-symmetry relationships between the two types of roof (2 and 4 slopes) enables abstraction. Second row: on this dense urban component each roof is simple but all roofs form a complex arrangement as the buildings have been built at different times with little coherence. Third row: on this architectural building both Z-symmetry and orthogonal relationships cooperate to abstract the central part of the roof. Fourth row: this building contains complex and thin roof superstructures. Despite a limited accuracy of the input MVS mesh our method recovers the main facades and roofs, and most superstructures.

Table II. Running times and output complexity. The time required for reconstruction LOD2 and LOD3, as well as for extracting LOD0 and LOD1 are similar as the time required for iconization and height filtering is negligible. The complexity refers only to the number of polygon facets of the building models at LOD2, the trees and superstructures being omitted.

<table>
<thead>
<tr>
<th>Input mesh</th>
<th>Classification</th>
<th>Planar proxies</th>
<th>Iconization</th>
<th>LOD0 reconstruction</th>
<th>LOD1 reconstruction</th>
<th>LOD2 reconstruction</th>
<th>Model complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Church (59K triangle facets, Fig.17)</td>
<td>5s</td>
<td>1.5s</td>
<td>2s</td>
<td>41s</td>
<td>198s</td>
<td>190 facets</td>
<td></td>
</tr>
<tr>
<td>Building block (170K triangle facets, Fig. 15, 2nd row)</td>
<td>7s</td>
<td>1.1s</td>
<td>1.1s</td>
<td>21s</td>
<td>137s</td>
<td>456 facets</td>
<td></td>
</tr>
<tr>
<td>Invalides district (11M triangle facets, Fig. 16)</td>
<td>55s</td>
<td>95s</td>
<td>36s</td>
<td>17min</td>
<td>112min</td>
<td>175K facets</td>
<td></td>
</tr>
</tbody>
</table>

numbers.

Robustness. Cases that challenge the robustness of our algorithms include input meshes with insufficient density and defects such as noise, holes and overlaps. Fig.15(fourth row) shows that small scale roofs may not be reconstructed in LOD2 but are recovered in LOD3 as roof superstructures. Imperfect input data often lead to over- or under-detected planar proxies. We observe that over-detection is often compensated by the proxy regularization procedure that merges nearly-coplanar proxies. Under-detection however leads to very few proxies as observed on free-form architectural buildings, and hence to an overly abstracted reconstruction. Nevertheless, in the worst case where no proxies are detected for a building component, the output LOD is abstracted as its bounding box. Data that challenge the classification step include merged objects such as a tree touching a facade, and clutter elements such as cars or hedges digested by the four classes of interest. The regularization term of the energy
together with the semantic rules improve spatial consistency and reduce the number of classification errors. Fig. 18 evaluates the robustness of the proxy detection and abstraction on an input mesh with variable scale features, noise and smoothed features. As shown by Fig. 19, the pipeline is to some extent robust to missing data (small holes) in the input mesh. In particular, the arrangement of planes provides a means to reconstruct sharp creases and corners even when data are missing near the intersection of proxies. When holes are too large, the algorithm may fail detecting some planar proxies that are important to infer the correct structure.

**Accuracy and abstraction.** Fig. 17 evaluates the accuracy of the reconstructed LODs against the input meshes, albeit our approach is designed to provide a tradeoff between faithfulness to input data and structure-aware abstraction. The comparisons against two mesh approximation approaches [Garland and Heckbert 1997; Cohen-Steiner et al. 2004], referred to as QEM and VSA respectively, show comparable approximation errors, better resilience to holes and topological artifacts of the input mesh.


Fig. 17. Geometric accuracy and structure-awareness. We compare the LODs to two mesh approximation algorithms by measuring the Hausdorff distance (color scale from yellow to black) to the input mesh. The complexity of the LOD2, QEM [Garland and Heckbert 1997] and VSA [Cohen-Steiner et al. 2004] models is identical (190 facets). LOD2 without plane regularization has a lower root mean square error (RMS) than LOD2 with planar regularization but is less abstracted and consumes more time to reconstruct. In terms of structure-awareness, thin components such as the church towers are correctly preserved in the different LODs, which is not the case for mesh approximation algorithms. In addition, QEM and VSA do not fill the holes contained in the input mesh (see front right facade).

Fig. 18. Robustness. Left: “defect-free” input mesh colored by superfacades and its planar proxies (middle). Our reconstruction algorithm (applied here without classification to evaluate only the proxy detection and abstraction steps) recovers most features (bottom). Notice the curved area reconstructed by planar polygons. Middle left: when noise is added the small scale features are filtered out and the vault is overly simplified. Right: when fed with the output of the Poisson reconstruction method the behavior of the algorithm is similar to the one on the smoothed mesh (middle right).

Input data. Deciding upon the best type of input data for urban reconstruction is a recurrent dilemma [Leberl et al. 2010]. While most existing approaches take point clouds as input, in particular LIDAR scans, we argue that dense meshes generated by MVS workflows offer significant advantages. Contrary to point clouds, MVS meshes have richer geometric and topological description derived from photo-consistency principles. To our knowledge, none of the existing surface reconstruction methods from point clouds are able to combine semantization and photo-consistency in order to generate abstracted LODs. Fig.20 describes a qualitative comparison of our approach against three specialized urban reconstruction methods from LIDAR scans based on planimetric arrangement [Lafarge and Mallet 2012], point set structuring [Lafarge and Alliez 2013], and global 2.5D regularization [Zhou and Neumann 2012].

Limitations. We limited the classification to four common classes of urban objects. At first glance such a low class number may appear restrictive in terms of semantics, but these four classes match CityGML and the requirements of several application needs. They provide a satisfactory tradeoff between robustness and quality of the reconstruction (We found only few errors during visual inspection of the large scale scene at LOD1 and LOD2 against the
Fig. 20. Comparisons with urban reconstruction methods. Starting from a low-density airborne LIDAR point cloud, we generate a dense mesh using a standard elevation grid used as input to our algorithm. The two methods specialized to airborne LIDAR (planimetric arrangement [Lafarge and Mallet 2012] and global 2.5D regularization [Zhou and Neumann 2012]) as well as point set structuring [Lafarge and Alliez 2013] also provide low-complexity and structure-aware representations. However, our approach generates 3D models with higher level of abstraction through regularization, see the distribution of output normals where only one (resp. four) non-horizontal direction are required at LOD1 (resp. LOD2). As highlighted by close-ups (colored by normal directions), we recover non-trivial adjacency of roof sections with improved accuracy.

airborne tiled image, see Fig. 16). This choice hampers the reconstruction of less common urban structures such as bridges or elevated roads. Our 3-step pipeline is however amenable to inserting additional classes with new labels in the MRF-based classification, and new objects to abstract and reconstruct. The combination of large buildings and irregular non-flat ground can yield classification errors: e.g., sharp creases of the ground surface labeled as building, or large buildings labeled as ground. The use of planar proxies is also a limitation when dealing with freeform architecture buildings such as the dome of Les Invalides depicted by Fig. 16.

At first glance our approach may be seen as a complex assembly: iconization on depth maps for trees, superstructures and facade elements, as well as 3D arrangements of primitives for buildings. A closer look however reveals that our methodological choices are specialized to the scale, structure and semantic of data. In addition, they are matching the limitations and constraints of real-world measurement data: despite recent advances on airborne acquisition the resolution of MVS meshes is too limited to handle roof superstructures with a process similar to the one applied on buildings. Methods specialized to high resolution depth maps reveal more appropriate for accurate faithful reconstruction of superstructures and facade elements.

6. CONCLUSIONS

Our work on LOD generation for urban scenes provides an automated framework to generate semantic-aware LODs from raw meshes. The four LODs generated are both meaningful and refine-able thanks to a coherent design of the abstraction and reconstruction steps for the LODs (e.g., roof superstructures of LOD3 are reconstructed from height maps derived from the roofs computed in LOD2, and LOD0 is reconstructed by instantiating a 2D min-cut algorithm applied to a subset of abstracted proxies used in other LODs).

Our initial goal to devise a fully automated pipeline translates into a unsupervised classification method relying solely on geometric attributes and semantic rules. The classification is performing well with a small set of local geometric attributes and global solve. Our approach is shown to exhibit robustness to defect-laden meshes through regularized optimizations combined with 3D arrangements which generate well-behaved surfaces by construction. Our initial goal to devise a scalable workflow is also met by exploiting both the classification and abstraction steps in order to instantiate one reconstruction process per building or per tree component, and to reduce the combinatorial complexity of the 3D arrangements.

As future work we wish to devise a photo-consistent framework by exploiting in all steps of our approach the color attributes of the multi-view stereo images. We will also investigate the fusion of airborne and ground-based measurements in order to reconstruct facade elements such as doors or windows with more details.

Acknowledgments

We wish to thank Acute3D, Autodesk, Interatlas, French mapping agency, Qian-Yi Zhou and Henrik Zimmer for providing datasets and materials for comparisons. This work is supported by an ERC Starting Grant “Robust Geometry Processing” (257474).

REFERENCES

Transactions on Graphics, Vol. VV, No. N, Article XXX, Publication date: Month YYYY.


