Planar Shape Detection of Structural Scales

Hao Fang Florent Lafarge
Inria
firstname.lastname@inria.fr

Mathieu Desbrun
Caltech
mathieu@cms.caltech.edu

Abstract

Interpreting 3D data such as point clouds or surface meshes depends heavily on the scale of observation. Yet, existing algorithms for shape detection rely on trial-and-error parameter tunings to output configurations representative of a structural scale. We present a framework to automatically extract a set of representations that capture the shape and structure of man-made objects at different key abstraction levels. A shape-collapsing process first generates a fine-to-coarse sequence of shape representations by exploiting local planarity. This sequence is then analyzed to identify significant geometric variations between successive representations through a supervised energy minimization. Our framework is flexible enough to learn how to detect both existing structural formalisms such as the CityGML Levels Of Details, and expert-specified levels of abstraction. Experiments on different input data and classes of man-made objects, as well as comparisons with existing shape detection methods, illustrate the strengths of our approach in terms of efficiency and flexibility.

1. Introduction

Shape detection from raw 3D data is a long-standing problem whose goal consists in turning a large amount of geometric data into a higher level representation based on simple geometric shapes. Instead of reasoning at the scale of 3D atomic elements such as points, triangular facets or voxels, it is often more appealing to directly handle larger geometric shapes in order to both reduce the algorithmic complexity and analyze objects with a higher representation level. Most common geometric shapes include lines, planes and quadrics. In this work, we focus on planar shapes due to their relevance to man-made environments [17].

Shape detection is typically used as a prior step in a large variety of vision-related tasks ranging from surface reconstruction [2, 5, 29, 37, 20] to object recognition [4, 22] and data registration [7, 38]. Existing algorithms typically require two user-specified parameters: (i) *a fitting tolerance* ϵ that specifies the maximal distance of a datum to its associated geometric shape, and (ii) *a minimal shape size* σ that specifies how large a group of samples must be to be considered as a geometric primitive—typically, a number of inliers when dealing with point clouds, or a minimum area for meshes. Finding parameter values that produce desirable results often involves fastidious manual labor: surprisingly, the incidence of these two parameters on shape detection has not been formally studied in the literature.

In this work, we propose an efficient exploration of this (ϵ, σ) space of geometric abstractions to find the *structural scales* of an input geometry, i.e., the few simplified representations that are truly meaningful to capture the structure of man-made objects. From a progressive planarity-driven coarsening of the input data, we demonstrate that we can reliably detect structural scales whose characteristics are learned from training sets of different types of objects such as buildings, house furniture, or cars.

2. Related work

We first review prior work in three related aspects of our goal: shape detection, scale-space exploration, and classification of 3D data.

Shape detection. The automated detection of geometric shapes from 3D measurement data is an instance of the general problem of fitting mathematical models to data. Among many successful approaches, region growing based approaches [26] are very efficient when input data is relatively clean. They proceed by growing a local shape hypothesis in a spatial neighborhood of a seed point while fitting is observed. In presence of outliers, RANSAC-based algorithms such as [27] typically performs best by iteratively constructing many shape hypotheses from a few samples, verifying them against the input data, and selecting the shapes with the highest numbers of inliers. Accumulation space methods [6, 12, 1] operate through voting in the parameter space of the shapes. Gaussian sphere mapping is a common choice for 3D data but requires a good estimation of normals. More recently, various research efforts have tried to both detect and regularize shapes according to geometric relationships such as parallelism or orthogonality [30, 23]. This can also be solved as a labeling problem by turning shape detection into shape selection among a finite set of candidates [10, 25, 19]. Although these methods are popular in 3D vision, obtaining a representation that adequately describes the input shape often requires time-intensive parameter tuning.

Scale-space exploration. Scale-space theory is an interesting framework for representing nD signals at multiple scales, in particular for the analysis of images [15] and 3D data [24, 14, 18]. Such methods rely on the detection of geometric variations in a spatial neighborhood whose size is controlled by a continuous scale parameter. These methods, which have been designed to apply to free-form objects with continuously differentiable surfaces, are however ill suited to man-made objects that typically involve piecewise parametric surfaces. For such objects, existing works use filtering procedures to select shapes at different structural levels, usually called *levels of details* (LOD [16]). Shape selection typically relies on simple user-defined area and orientation rules [17, 32]. In particular, there is no scale parameter to characterize, detect or learn structural variations.

3D object recognition. Detecting structural scales is closely related to 3D object recognition. Traditional methods exploit geometric descriptors to locally characterize data distribution, e.g., through corner and edge detectors [21], scatter matrix [31] or spin images [11]. These 3D descriptors are usually embedded into a learning procedure to recognize, for instance, urban objects from LIDAR scans [8]. With the emergence of deep learning, recent methods exploit multi-layered voxel-based representations [35] to strongly simplify the shape of objects. Deep learning techniques are efficient for recognition tasks, but are so far less attractive for addressing reconstruction issues. Moreover, these recognition methods operate on raw geometric data of free-form objects, and cannot be easily extended to shape representations of man-made objects.

3. Motivation and overview

The motivation behind our work is to explore the (ϵ, σ) space of shape approximation for a given input 3D scene, where ϵ quantifies the geometric tolerance to data and σ defines the minimum number of inliers: its geometric relevance to the issue of shape and scale detection has been repeatedly confirmed (see, e.g., [26]). Yet, it may appear at first sight that finding meaningful abstractions of input shapes by exploring this (ϵ, σ) space is simply intractable: even a greedy search through discrete sampling is unlikely to find the few key structural scales that we seek. We observe, however, that for a vast range of 3D objects (including man-made shapes), the meaningful structural scales are likely to be well captured along the (bottom-left to top-right) diagonal of the parameter space (ϵ, σ) as illustrated in Figure 1. This property has an important practical con-

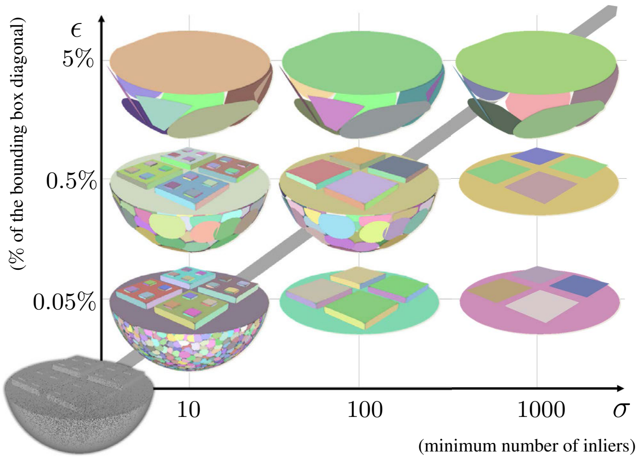

Figure 1. *Influence of shape detection parameters.* A point sampled object partially piecewise-planar (bottom left) is turned into a set of planar elements by region growing [26] given a fitting tolerance ϵ and a minimal shape size σ. Increasing σ for a fixed ϵ progressively removes the smallest planar elements. Simplifications that are most representative of a key structural scale are located along the bottom-left to top-right diagonal: above (resp., below), planar regions (resp., free form parts) disappear too fast.

sequence: we can turn this two-parameter exploration task into a simple 1D exploration along this diagonal—a far more tractable task.

We are left with two issues to address: (i) how to sample efficiently the shape configurations along the parameter space diagonal which are likely to cross the different structural scales, and (ii) how to detect structural scales robustly.

To address (i), we propose a shape-collapsing procedure described in Section 4 that merges progressively pairs of planar shapes from an initial configuration with both low ϵ and σ, i.e., a configuration at the bottom left of the parameter space of Figure 1. Since merging two planar shapes cannot decrease the maximal distance to an inlier or the minimum shape size, repeated shape merging will generate a sequence of shape representations near the diagonal of the parameter space, as illustrated in Figure 2. Such a procedure is very efficient, and returns a fine discretization of abstractions roughly along the diagonal of our two-parameter space: starting from n planar shapes, we produce a sequence of n shape configurations called a *trajectory* in the parameter space.

As structural scales correspond to arbitrary levels of abstraction, solving (ii) by tracking and quantifying the geometric changes along this diagonal is not a reliable approach to detect them. Instead, we adopt an efficient strategy detailed in Section 5 that consists in learning the geometric characteristics of structural scales from a training set. The latter is typically created by a manual assignment of structural scales to the configurations of trajectories obtained by our shape collapsing procedure on a few test datasets. This training strategy offers the advantage to be fast compared to

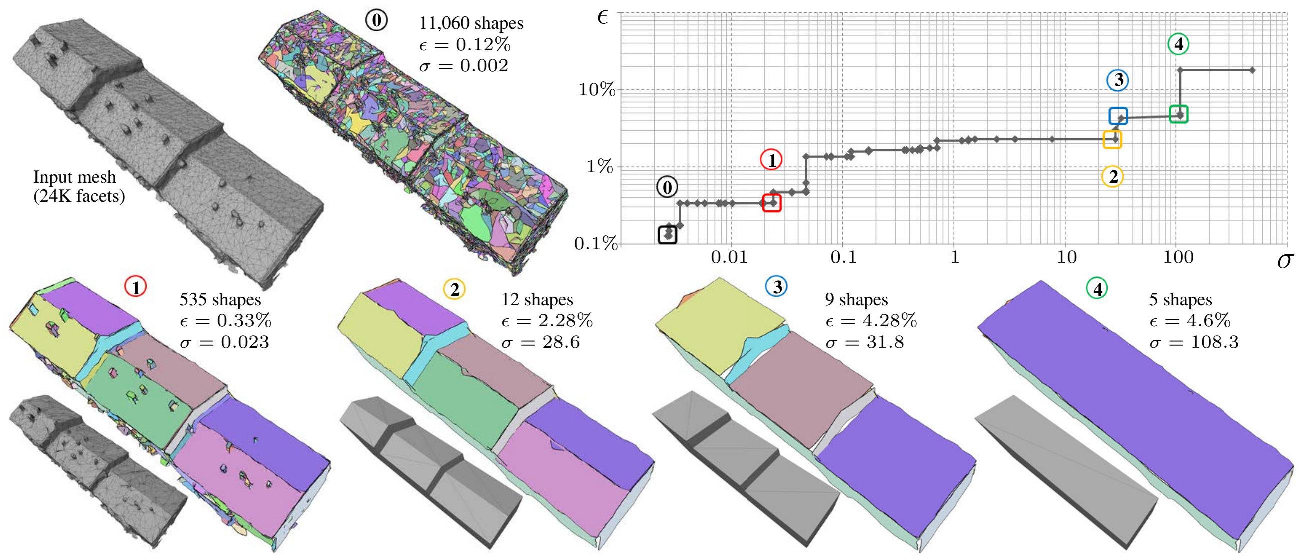

Figure 2. *Overview.* Starting from 3D data (here a dense mesh generated by MultiView Stereo, top left), our algorithm produces a set of high-level representations with planar primitives (representations 1–4) describing the object at different representative structural scales (bottom). By progressively merging planar regions of an initial state (representation 0), one creates a sequence of representations whose further analysis allows for the extraction of a few structurally relevant representations (top right). Such shape representations can be used, for instance, as input for piecewise-planar reconstruction [5] (see grey compact meshes). Note that each shape is displayed as a colored polygon computed as the α-shape of its inliers projected onto the shape; we use this visualization of inliers in all following figures.

a greedy exploration of the 2D parameter space, and consistent with the way planar shapes are sampled during the testing.

4. Shape collapsing

Our shape-collapsing process iteratively merges two planar shapes from a current shape abstraction. This approach relies on two key ingredients: a merging operator specifying how to create a new planar shape from two existing ones, and a priority policy that orders the shape pairs to merge.

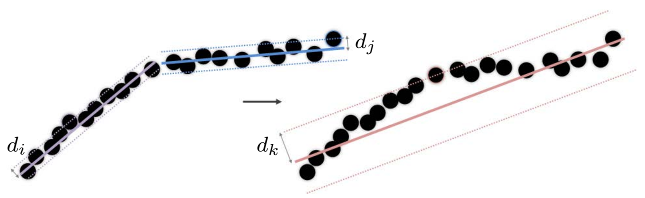

Figure 3. *Merging operator.* Two adjacent shapes i and j are merged into the shape k that minimizes the Euclidean distance to their joint sets of inliers. If d_i denotes the distance between shape i and its furthest inlier, note that $d_k \geq \max(d_i, d_j)$.

Initialization. We start by extracting an initial configuration of planar shapes from input data, be it a 3D point cloud or a surface mesh. A region growing algorithm [26] is used with low parameter values, typically $\epsilon = 0.05\%$ of the bounding box diagonal, and $\sigma = 10$ inliers. As preprocessing, we compute an adjacency graph between the detected shapes based on spatial proximity: for surface meshes, two planar shapes are considered as adjacent if at least a pair of their respective inlier facets shares a common edge in the input mesh; for a point cloud instead, two shapes are adjacent if at least a pair of their respective inlier points are mutual neighbors in the k-nearest neighbor graph of the input points (we use $k = 20$ in all our experiments).

Merging operator. This operator is applied on the edges of the adjacency graph. It merges two adjacent planar elements into the planar shape that minimizes the Euclidean distance to their joint sets of inliers, as illustrated in Figure 3. The optimal planar shape is trivially found via Principal Component Analysis.

Priority policy. In order to choose the next pair of planar shapes to merge, a weight is assigned to each edge of the adjacency graph. Merging is then performed on the edge with the lowest weight. Different metrics can be considered for specifying the weights, e.g., deviation of the normal vectors of the two planes, or area of the smallest of the two shapes. After an experimental evaluation of several metrics, we chose the Euclidean distance between input points to planar shapes as it offers the best compromise between accuracy and performance. In particular, this choice limits drifts during shape collapsing because it relies on a direct measurement to input data. Formally, we define the weight w_{ij} between planar shapes i and j as

$$w_{ij} = \sqrt{\frac{1}{\sigma_i + \sigma_j} \sum_{p_k \in I_{ij}} d(p_k, P)^2} \tag{1}$$

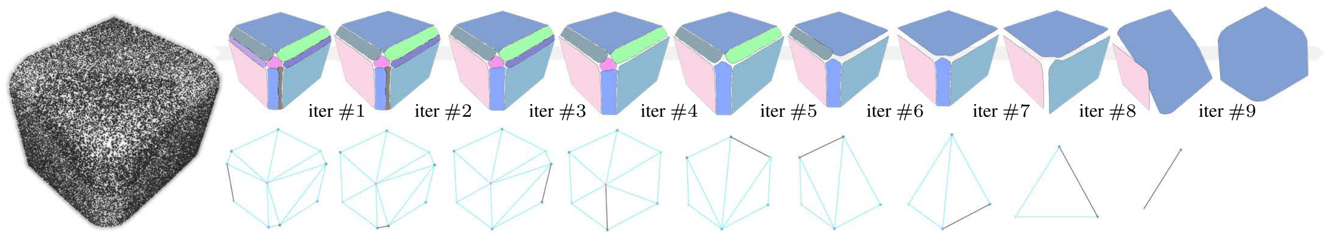

Figure 4. *Shape collapsing.* Iteratively merging adjacent planar elements creates a sequence of shape representations, some of which being structurally representative, e.g., representations obtained after iterations #4 and #7 (top). At each iteration, the black edge in the adjacency graph (bottom) indicates the edge with the lowest weight, i.e. the next edge to be collapsed.

where σ_i is the size of shape i, I_{ij} is the joint set of inliers from shapes i and j, and P is the optimal planar shape computed by the merging operator. At each iteration, we choose the pair of shapes with the lowest weight as the candidates to be merged. After merging two shapes, the adjacency graph as well as the weights are updated. Note that this update is local as only edges with the planar shapes adjacent to the two merged shapes are impacted. Figure 4 illustrates this procedure.

5. Detection of structural scales

Given a roughly-diagonal trajectory in parameter space, our goal is now to detect structural scales by analyzing the geometric evolution of the shape representations along the trajectory. For an object with simple structure, the problem can be solved in a unsupervised manner by detecting strong geometric variations between two successive piecewise-planar representations. However, in mosts cases, structural scales are levels of abstraction that cannot be reliably detected without learning from training samples. We thus formulate the detection of structural scales as a supervised labeling problem by assigning a structural scale to each shape configuration of the trajectory.

Feature vector. We define a feature vector in order to characterize a configuration of planar shapes from a geometric point of view. Four different geometric descriptors are used:

- *Centroid distance* that computes the Euclidean distance between the barycenters of two adjacent shapes;
- *Normal alignment* measuring $|\mathbf{n}_i \cdot \mathbf{n}_j|$ between the normals $\mathbf{n}_i$ and $\mathbf{n}_j$ of two adjacent shapes;
- *Area variation* that computes $1 - |\sigma_i - \sigma_j|/|\sigma_i + \sigma_j|$ from the sizes σ_i and σ_j of two adjacent shapes;
- *z-axis deviation* that compares the relative orientation of two adjacent shapes with the z-axis $\mathbf{n}_z$ through the expression $|\ |\mathbf{n}_i \cdot \mathbf{n}_z| - |\mathbf{n}_j \cdot \mathbf{n}_z|\ |$.

For each descriptor, we create an histogram describing the distribution over all the pairs of adjacent shapes. We then normalize each histogram and concatenate them into a 18-bin feature vector, as illustrated in Figure 5. We denote by $\mathbf{f}_i$ the feature vector of shape representation i. Such a simple feature vector summarizes the main geometric characteristics of a shape representation as mutual position, orientation, size and alignment of pairs of adjacent shapes.

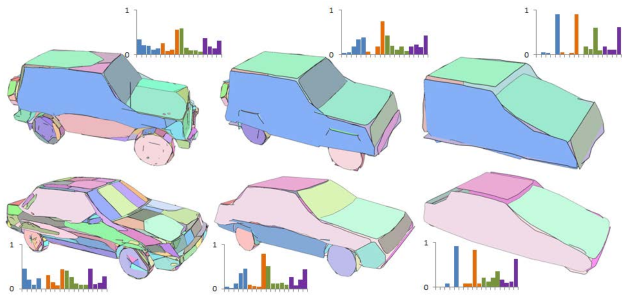
Figure 5. *Feature vector.* Feature vectors (see histograms) can discriminate between shape representations that capture different structural levels of man-made object, here cars. Four bins are used for both normal alignment (orange) and z-axis deviation (navy), and five bins for centroid distance (blue) and area variation (green).

Energy minimization. Recall that from an initial configuration composed of n planar shapes, repeated collapsing generates a trajectory with n shape representations. Given a finite set of structural scales $\mathcal{L} = \{1, 2, ..., K\}$, we consider a random variable $l_i \in \mathcal{L}$ that associates a structural scale to the i^{th} shape configuration of the trajectory. The quality of a label assignment $l = (l_i)_{i \in [1,n]}$ over a trajectory is measured through an energy U of the form

$$U(l) = \sum_{i=1}^{n} \psi_i(l_i) + \gamma \sum_{i=1}^{n-1} \varphi_{i,i+1}(l_i, l_{i+1}) \tag{2}$$

where $\psi_i(l_i)$ is a unary data term, $\varphi_{i,i+1}(l_i, l_{i+1})$ is a pairwise potential that accounts for temporal consistency between two successive shape representations, and $\gamma > 0$ is a weight balancing the two terms. In all our experiments, γ has been fixed to 0.5. Note that this formulation is basically a Hidden Markov model, so the configuration that minimizes energy U is found by dynamic programming using the Viterbi algorithm [33].

Choice of ψ_i. The unary data term of shape representation i is formulated using a classifier trained by Random Forests [3]. It is expressed by:

$$\psi_i(l_i) = -\frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}}\log(P_t(l_i|\mathbf{f}_i)), \tag{3}$$

where $\mathcal{T}$ denotes a set of decision trees, $|\mathcal{T}|$ the number of trees, and P_t the prediction probability of the label l_i for the decision tree t.

Choice of $\varphi_{i,i+1}$. The pairwise potential promotes temporal consistency along the trajectory: it penalizes scale changes between successive representations when geometrically too similar. This potential is defined through

$$\varphi_{i,i+1}(l_i, l_{i+1}) = w_{i,i+1}\cdot T(l_i, l_{i+1}) \tag{4}$$

where $w_{i,i+1} = \exp(-d_{\text{EM}}(\mathbf{f}_i, \mathbf{f}_{i+1})/2)$ is a weight measuring the similarity between feature vectors $\mathbf{f}_i$ and $\mathbf{f}_{i+1}$. The distance d_{EM} is defined as the L_2 norm of the Earth Mover distances for each descriptor histogram using a L_1 ground distance. This weight favors high geometric variation between two successive representations with different labels. The term $T(l_i, l_{i+1})$ measures jump coherence from scale l_i to scale l_{i+1}, and is defined as

$$T(l_i, l_{i+1}) = \begin{cases} 0 & \text{if } l_{i+1} = l_i \\ 1 & \text{if } l_{i+1} = l_i + 1 \\ +\infty & \text{otherwise} \end{cases} \tag{5}$$

The role of $T(l_i, l_{i+1})$ is to weakly penalize a jump between two successive scales while preventing other jumps in scale.

The resulting labeled sequence assigns a same label to a whole range of representations. The *first* shape representation with a given label is selected as representative of the object structure at this scale. With this choice, every planar shape is a relevant component of the object structure.

6. Experiments

We tested our method on three datasets with (i) different man-made objects (buildings, cars, sofa and indoor scenes), and (ii) different input data including synthetic/real-world surface meshes and point clouds. We only considered three scales in all our experiments: one scale with fine details, one with general structure and no fine details, and one with an overly-simplified general shape; but any (typically small) number of scales can be used.

- *CAD dataset.* The Princeton Shape database [28] is used to generate noise-free input point clouds that uniformly sample CAD models. Models are mainly composed of free-form shapes, including cars and sofas. The three structural scales are levels of abstraction that were specified by an expert.
- *MultiView Stereo dataset.* We created a dataset of buildings represented by dense surface meshes generated from MultiView Stereo (MVS [34]). These dense meshes contain fine details such as chimneys, but have a high amount of defects in the form of noise, holes and erroneous topology. We trained the algorithm to recognize the Levels Of Details 1, 2 and 3 defined by the cityGML formalism [9] as structural scales.
- *RGB-D dataset.* We also evaluated our algorithm on point clouds generated by RGB-D cameras from the Sun3D database [36] and datasets from [13]. These 3D point sets correspond to indoor scenes, each representing a room with walls, floor and furniture. Inputs are defect laden with variable noise, heterogeneous spatial density and severe occlusions. The three structural scales are levels of abstraction that were specified by an expert.

For each class of man-made objects, we randomly selected one third of the models for training, and the two remaining third for testing. To create planar configurations at representative structural scales for the training set, we created sequences of configurations by our automatic shape collapsing process and then assigned a scale label to each configuration by visual inspection. To speed-up the annotation, we visually detect the pairs of successive configurations where the scale changes, and then automatically annotate the configurations in between. Such a training procedure is (i) fast, *i.e.,* from 30 minutes (Multiview dataset) to 2 hours (RGB-D dataset) to create the full training set, and (ii) consistent with our two-step strategy since training samples are also generated from shape collapsing.

Qualitative and quantitative evaluation. Figure 6 presents some qualitative results on small portions of the three datasets. We observe that the computed representative shapes for each structural scale on testing examples are structurally similar to those in the training samples. Our framework is flexible enough to learn shape detection from both existing formalisms such as the CityGML LODs for representing buildings, and expert-specified levels of abstraction of man-made objects. Table 1 demonstrates that our resulting scale labeling is fairly accurate. One may note that accuracy on the MultiView Stereo dataset is much higher than for the other datasets; two main reasons explain this difference: buildings are less free-form than cars or furnitures, and levels of abstraction for building are less subjective. Once trained on a specific class of object, the classifiers do not generalize particularly well when tested on

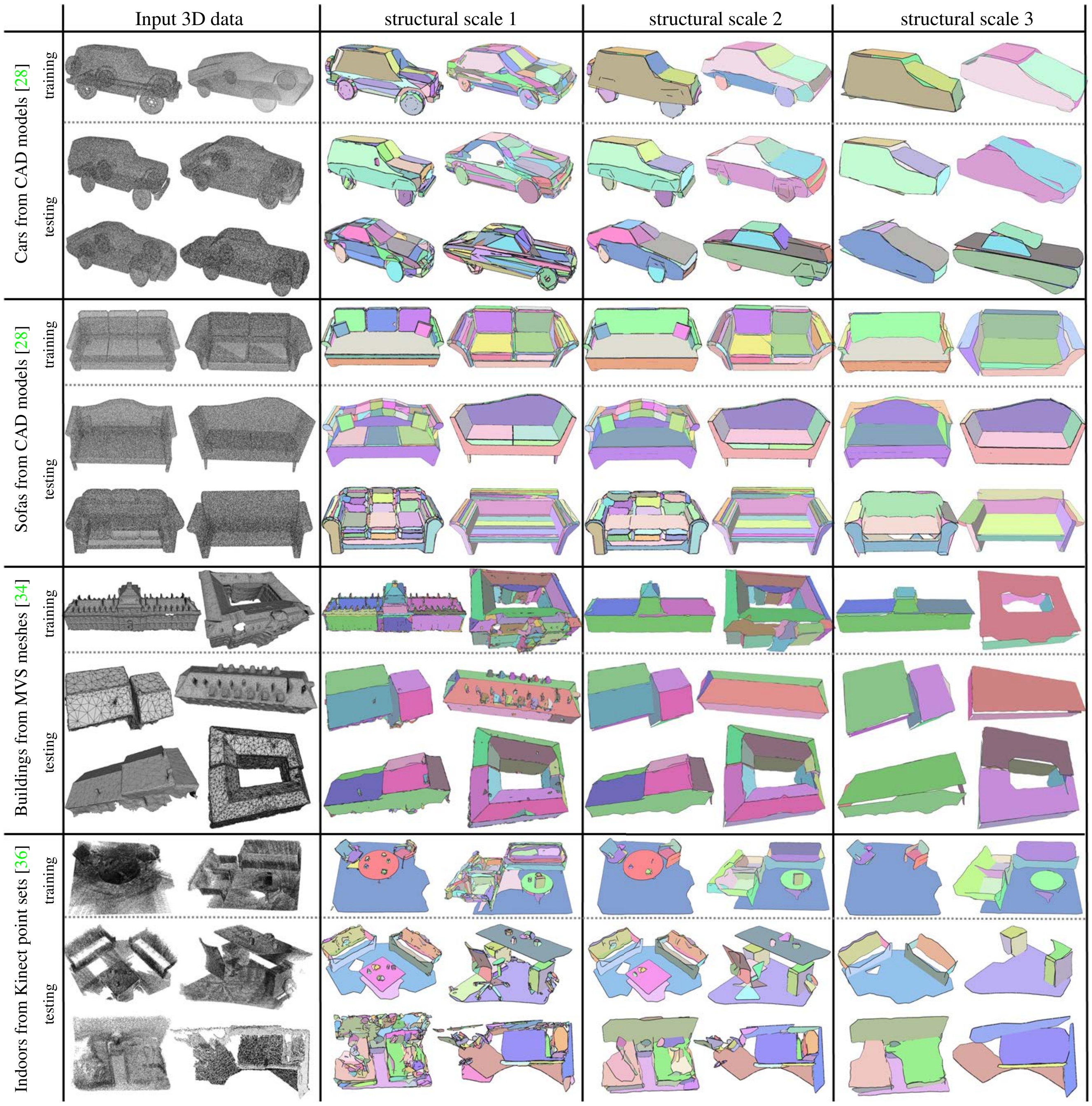

Figure 6. *Results on different man-made objects.* The shape representations archetypical of each structural scale generated by our algorithm on testing examples have similar structures to the training samples. In particular, our algorithm is able to learn the CityGML formalism and produce meaningful shape representations of buildings at different LODs. For indoor scenes, both furniture and permanent elements such as floor and walls exhibit the same level of detail at a given scale. Even for less structured objects such as cars or sofas, the level of abstraction conveyed by planar elements remains consistent between training and testing. Note in particular how cars at scale 1 have their bonnet described by many elements, which turn into a single element at scale 2, before merging with the windshield at scale 3.

other object categories: accuracy typically decrease proportionally to the similarity between objects, *e.g.* applying the ”Car” classifier on the ”Sofa” dataset decreases accuracy from 86% to 63%.

Robustness to data defects, object size and initialization. As scale detection is performed using normalized features, our algorithm is only weakly affected by noise: adding 1% random noise in the car dataset only decreases the general accuracy by 1.8%. Initialization can be an issue if we start

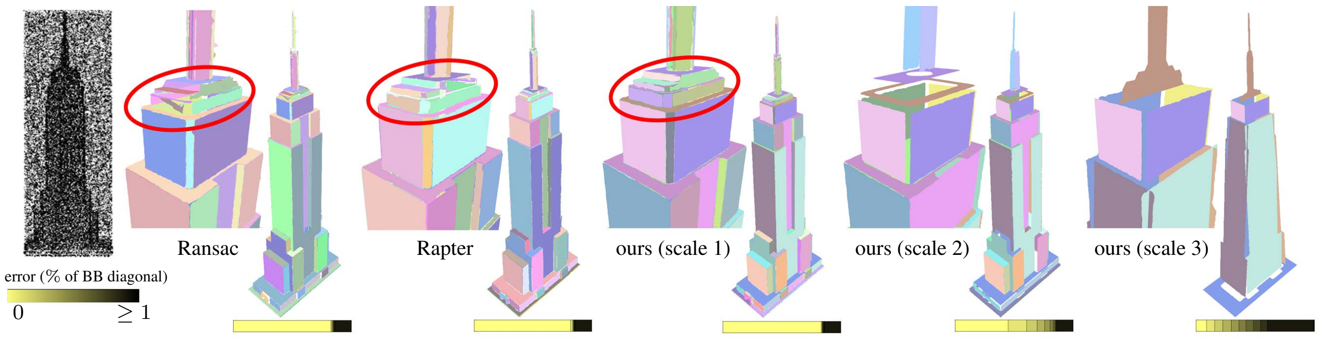

Figure 7. *Comparisons on Empire.* The result from Rapter [19] (courtesy of the authors) finds a visually-significant configuration of planar shapes to describe the building, whereas the one from Ransac [27] was obtained by manual parameter tuning to obtain a result as close as possible as our scale 1. While Ransac and Rapter exhibit similar error distributions with respect to input points (see color histograms from yellow to black), our algorithm produces three output representations that strongly differ in terms of geometric accuracy and number of planar elements, while guaranteeing a similar coverage. Our representation at scale 1 is more meaningful than those obtained by these two methods. In particular, Ransac and Rapter omit fine planar components on the top of the tower.

with too large ϵ and σ values that are located after the first scale. In practice, there is no accuracy difference on the MVS meshes if we start with $\epsilon = 0.05\%$ and $\epsilon = 0\%$, *i.e.*, with each triangular facet as a shape. Since histograms of descriptors are normalized, our classifier is robust to object size variability as well: while the buildings in Figure 6 have quite different sizes (from small cottages to entire blocks), their shape representations are consistent at each scale.

Object class	#training samples	#testing samples	training accuracy	testing accuracy
CAD car	5K	12K	98.53%	82.88%
CAD sofa	3K	4K	97.60%	85.88%
MVS building	9K	12K	99.61%	99.30%
RGB-D indoor	20K	26K	96.90%	80.60%

Table 1. Accuracy of scale labeling on training and testing sets for different object classes.

Timings. Learning the classifier on the different datasets requires from 5 seconds for the MultiView Stereo dataset (9K training samples) to 2.5 minutes for the RGB-D dataset (20K training samples) for a random forests training with 100 trees and 25 levels. Table 2 details timings for testing on one representative sample of each object class. Shape collapsing is the most time-consuming step, whereas the timing for scale detection is negligible and independent of the input complexity.

Comparisons with shape detection methods. We compared our algorithm to an advanced Ransac-based method [27], and the Rapter labeling mechanism [19]. A fair comparison must consider three main evaluation criteria: geometric fidelity, coverage and output complexity. We chose as measures the root mean square distance of detected shapes to inliers, the ratio of points assigned to shapes, and the number of shapes respectively. Contrary to our algorithm, these other methods required tuning some parameters as the fitting tolerance. Table 3 presents the evaluation scores from two input point clouds representing complex buildings, whereas Figure 7 shows visual results with error distributions. Our output shape representations at three different scales better capture the structure of the buildings while remaining competitive with existing methods in terms of geometric fidelity, coverage and output complexity.

Object class	Input complexity	Initialization	Shape collapse	Scale detection
CAD car	143K pts	4.05s	10.7s	0.24s
CAD sofa	142K pts	4.79s	21.6s	0.16s
MVS mesh	3.3K facets	0.31s	0.54s	0.22s
RGB-D indoor	1.15M pts	114s	12min	0.72s

Table 2. Running times for testing on one representative sample of each object class (see the first testing model for each class in Figure 6). Experiments have been done on a single-core Intel Core i7 processor clocked at 2GHz.

	RMS	coverage	#planes
Ransac [27]	0.034	0.808	128
Rapter [19]	0.042	0.817	163
Ours (scale 1)	0.017	0.816	239
Ours (scale 2)	0.29	0.816	40
Ours (scale 3)	1.03	0.816	9

Table 3. Comparisons on *Empire* in terms of Root Mean Square distance (RMS) of detected shapes to inliers (unit expressed as % of the bounding box diagonal), coverage (ratio of inliers) and number of shapes. Note that the shape collapsing process guarantees an identical coverage for outputs at different scales.

Application to surface reconstruction. By connecting our algorithm to a polyhedral surface reconstruction method [5], we can generate compact piecewise-planar 3D models of bulidings at different LODs from dense defect-laden meshes. As shown on Figure 8, we outperform the state-of-

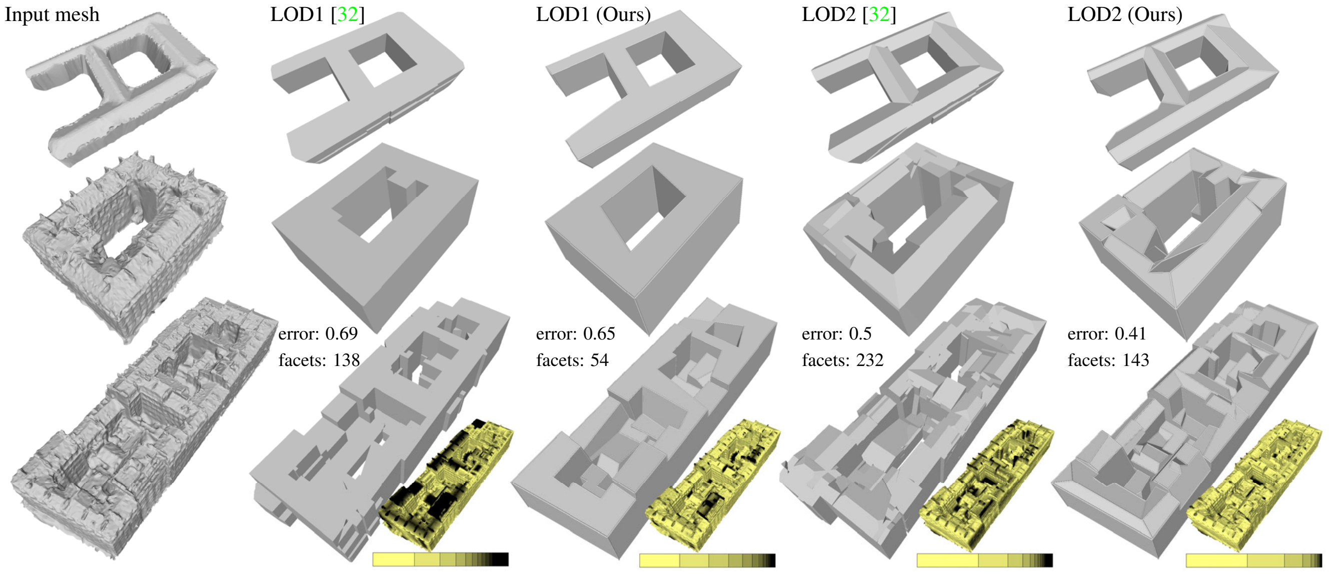

Figure 8. *Application to reconstruction of LOD models of buildings.* Our algorithm combined with a piecewise planar reconstruction algorithm [5] produces compact LOD1 and LOD2 models from dense defect-laden meshes that outperform those delivered by a building-specific LOD generation method [32] in terms of both geometric accuracy—as shown using color histograms from yellow (0 meter error) to black ($\geq$ 2 meter error)—and output complexity.

the-art method of [32] in terms of geometric accuracy and output complexity while conforming to the LOD CityGML formalism. Although [32] is specialized in producing LOD models of buildings, our learning strategy allows us to generate meaningful configurations of planes without explicitly specifying the rules of this LOD formalism.

Limitations. Although our framework is designed to be flexible, the choice of the metric (Equation 1) that specifies the priority weights during shape collapsing is independent of the object's category. As suggested by Table 1, our choice is relevant in the case of buildings for exploring LODs, but not always optimal for more free-form objects such as furniture. Ideally, this metric should be learned from a training set of trajectories. This variant would however be very costly in practice as shape collapsing and scale detection are no longer performed serially. Additionally, our algorithm does not discover and preserve geometric regularities such as parallelism, orthogonality or symmetry of shapes contrary to recent shape detection methods as [19]. This does not affect geometry fidelity and coverage, but may lead to suboptimal shape abstractions that fail to respect these specific features.

7. Conclusion

Our work provides a parameter-free algorithm for detecting piecewise-planar shapes from 3D data. Contrary to existing methods that require tedious parameter tuning, our algorithm extracts multiple representations of an input shape at key structural scales whose characteristics are learned from a training set. Our framework is flexible enough to learn both existing structural formalism such as the CityGML Levels Of Details for representing buildings, and expert-specified levels of abstraction on man-made objects. Experiments demonstrate the added value of our approach with respect to existing shape detection methods, as well as its potential to help with surface reconstruction and approximation.

As future work we wish to create priority metrics for shape collapsing that allow a more refined exploration and tracking of the structural scales in the parameter space. In particular, we would like to learn such metrics from a training set of trajectories. We also wish to extend our approach to quadric and volumetric primitives. This would however require a more involved shape collapsing strategy.

Acknowledgments

Hao Fang and Florent Lafarge thank CSTB for financial support. Mathieu Desbrun gratefully acknowledges the Inria International Chair program and the entire Titane team.

References

[1] D. Borrmann, J. Elseberg, K. Lingemann, and A. Nchter. The 3D Hough Transform for plane detection in point clouds: A review and a new accumulator design. *3D Research*, 2(2), 2011. 1

[2] A. Boulch, M. de La Gorce, and R. Marlet. Piecewise-planar 3d reconstruction with edge and corner regularization. *Computer Graphics Forum*, 33(5), 2014. 1

[3] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001. 5

[4] P. Carr, Y. Sheikh, and I. Matthews. Monocular object detection using 3d geometric primitives. In *ECCV*, 2012. 1

[5] A.-L. Chauve, P. Labatut, and J.-P. Pons. Robust piecewise-planar 3D reconstruction and completion from large-scale unstructured point data. In *CVPR*, 2010. 1, 3, 7, 8

[6] J. Chen and B. Chen. Architectural modeling from sparsely scanned range data. *IJCV*, 78(2-3), 2008. 1

[7] E. Fernandez-Moral, W. Mayol-Cuevas, V. Arevalo, and J. G. Jimenez. Fast place recognition with plane-based maps. In *ICRA*, 2013. 1

[8] A. Golovinskiy, V. G. Kim, and T. Funkhouser. Shape-based recognition of 3D point clouds in urban environments. In *ICCV*, 2009. 2

[9] G. Groger and L. Plumer. Citygml interoperable semantic 3d city models. *Journal of Photogrammetry and Remote Sensing*, 71, 2012. 5

[10] H. Isack and Y. Boykov. Energy-based geometric multi-model fitting. *IJCV*, 97(2), 2012. 2

[11] A. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *PAMI*, 21(5):433–449, 1999. 2

[12] J. Knopp, M. Prasad, G. Willems, R. Timofte, and L. Van Gool. Hough transform and 3D surf for robust three dimensional classification. In *ECCV*, 2010. 1

[13] K. Lai, L. Bo, and D. Fox. Unsupervised feature learning for 3d scene labeling. In *ICRA*, 2014. 5

[14] J. Lalonde, R. Unnikrishnan, N. Vandapel, and M. Hebert. Scale selection for classification of point-sampled 3d surfaces. In *3DIM*, 2005. 2

[15] T. Lindeberg. *Scale-space theory in computer vision*, volume 256. Springer Science & Business Media, 2013. 2

[16] D. P. Luebke. *Level of detail for 3D graphics*. Morgan Kaufmann, 2003. 2

[17] R. Mehra, Q. Zhou, J. Long, A. Sheffer, A. Gooch, and N. J. Mitra. Abstraction of man-made shapes. *ACM trans. Graph.*, 28(5), 2009. 1, 2

[18] N. Mellado, G. Guennebaud, P. Barla, P. Reuter, and C. Schlick. Growing least squares for the analysis of manifolds in scale-space. *Computer Graphics Forum*, 31(5), 2012. 2

[19] A. Monszpart, N. Mellado, G. J. Brostow, and N. J. Mitra. Rapter: Rebuilding man-made scenes with regular arrangements of planes. *ACM Trans. Graph.*, 34(4), 2015. 2, 7, 8

[20] L. Nan and P. Wonka. Polyfit: Polygonal surface reconstruction from point clouds. In *ICCV*, 2017. 1

[21] J. Novatnack and K. Nishino. Scale-dependent 3D geometric features. In *ICCV*, 2007. 2

[22] S. Oesau, F. Lafarge, and P. Alliez. Object classification via planar abstraction. In *Proc. of the ISPRS congress*, 2016. 1

[23] S. Oesau, F. Lafarge, and P. Alliez. Planar Shape Detection and Regularization in Tandem. *Computer Graphics Forum*, 35(1), 2016. 1

[24] M. Pauly, R. Keiser, and M. Gross. Multi-scale feature extraction on point-sampled surfaces. In *Computer graphics forum*, volume 22, 2003. 2

[25] T.-T. Pham, T.-J. Chin, J. Yu, and D. Suter. The random cluster model for robust geometric fitting. In *CVPR*, 2012. 2

[26] T. Rabbani, F. van Den Heuvel, and G. Vosselman. Segmentation of point clouds using smoothness constraint. *ISPRS*, 36(5), 2006. 1, 2, 3

[27] R. Schnabel, R. Wahl, and R. Klein. Efficient ransac for point-cloud shape detection. In *Computer graphics forum*, volume 26, 2007. 1, 7

[28] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser. The princeton shape benchmark. In *Shape Modeling International*, 2004. 5, 6

[29] S. N. Sinha, D. Steedly, and R. Szeliski. Piecewise planar stereo for image-based rendering. In *ICCV*, 2009. 1

[30] J. Straub, G. Rosman, O. Freifeld, J. Leonard, and J. Fisher. A mixture of manhattan frames: Beyond the manhattan world. In *CVPR*, 2014. 1

[31] L. Teran and P. Mordohai. 3D interest point detection via discriminative learning. In *ECCV*, 2014. 2

[32] Y. Verdie, F. Lafarge, and P. Alliez. LOD Generation for Urban Scenes. *ACM Trans. Graph.*, 34(3), 2015. 2, 8

[33] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. on Information Theory.*, 13(2), 1967. 5

[34] H. Vu, R. Keriven, P. Labatut, and J. Pons. High accuracy and visibility-consistent dense multi-view stereo. In *PAMI*, volume 34, 2012. 5, 6

[35] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3D ShapeNets: A deep representation for volumetric shape modeling. In *CVPR*, 2015. 2

[36] J. Xiao, A. Owens, and A. Torralba. Sun3D: A database of big spaces reconstructed using sfm and object labels. In *ICCV*, 2013. 5, 6

[37] Q.-Y. Zhou and U. Neumann. 2.5D building modeling by discovering global regularities. In *CVPR*, 2012. 1

[38] Z. Zhou, H. Jin, and Y. Ma. Robust plane-based structure from motion. In *CVPR*, 2012. 1