2019 International Conference on Robotics and Automation (ICRA)
Palais des congres de Montreal, Montreal, Canada, May 20-24, 2019

Dense 3D Visual Mapping via Semantic Simplification

Luca Morreale and Andrea Romanoni and Matteo Matteucci
Politecnico di Milano
{luca.morreale,andrea.romanoni,matteo.matteucci}@polimi.it

Abstract— **Dense 3D visual mapping estimates as many as possible pixel depths, for each image. This results in very dense point clouds that often contain redundant and noisy information, especially for surfaces that are roughly planar, for instance, the ground or the walls in the scene. In this paper we leverage on semantic image segmentation to discriminate which regions of the scene require simplification and which should be kept at high level of details. We propose four different point cloud simplification methods which decimate the perceived point cloud by relying on class-specific local and global statistics still maintaining more points in the proximity of class boundaries to preserve the infra-class edges and discontinuities. 3D dense model is obtained by fusing the point clouds in a 3D Delaunay Triangulation to deal with variable point cloud density. In the experimental evaluation we have shown that, by leveraging on semantics, it is possible to simplify the model and diminish the noise affecting the point clouds.**

I. Introduction

Dense 3D visual mapping from images aims at building a 3D model which recovers as much part of the scene as possible. In this field, two main trends emerged: high accuracy dense mapping [1], [2], [3], [4], [5] and dense mapping from sparse data [6], [7], [8], [9]. The former most widespread, methods estimate a model as accurately as possible, computing per-pixel dense correspondences among images and fusing them into the 3D space by means of patch-based or volumetric representation such as voxels or 3D Delaunay triangulations [2], [3], [4], [5]; in some cases a further refinement step is added to improve accuracy and resolution [1], [10]. They are able to produce impressive results, but they are computationally expensive, especially when dealing with large scale environments; they strongly leverage the architecture optimizations, either on GPU or CPU side, to efficiently compute, move and store data [11], [12]. On the other hand, methods that are more efficient in terms of space and memory can better estimate a dense map by relying just on sparse data coming from structure from motion methods [6], [7], [8], [9]. They are able to run in real-time on a single core of a CPU [9], without a relevant resources consumption. Nevertheless, they output a low resolution 3D model.

In the former case the resulting model is usually rich of redundancies, especially where the scene presents planar or piecewise-planar regions, while in the latter case 3D points computed by Structure from Motion are too sparse to provide sufficient and satisfying information to reconstruct the scene fine grain details. Only few works address the problem of trading-off between these two approaches. Li *et al.* [13] adapt the resolution of the reconstructed mesh such that regions with less details contain fewer vertices. The method effectively limits the dimension of the mesh, but it is only applied to the mesh refinement stage. Lafarge *et al.* [14] proposed an interesting method which combines the mesh recovered via Multi-view Stereo with high level geometric primitives. This approach simplifies regions of the mesh which fit geometric primitives by means of a complex jump diffusion scheme acting together with mesh refinement after the initial dense mesh is estimated. Wu *et al.* [15] use generalized cylinders and *swept surfaces* to recover a dense simplified structure from Structure from Motion point clouds, while Gallup *et al.* [16] fit a n-layer map into the depth map to produce a compact and robust representation.

Schindler *et al.* [17] fit piece-wise planar surfaces to point clouds to obtain a schematic reconstruction of man-made environments, while Bourki *et al.* [18] propose an ad-hoc method for man-made structures: they simplify the depth maps, used to build a patch-based representation, by estimating the vanishing points and by fitting planes on the Structure from Motion output. Finally, some methods apply planar priors to limit the model noise, but they keep the model resolution fixed [19], [20].

None of these methods leverages on the semantic information carried by the images which can be effectively estimated by means of semantic image segmentation methods [21], [22], [23]. Semantics provides very useful priors to reason about which parts of the scene can be simplified and which parts need a more detailed model. Some semantic categories, as the ground or the walls, when reconstructed, contain redundant information which can be simplified, whilst others, as vegetation, require more points to describe their shape with high fidelity.

To the best of our knowledge only the method in Schneider *et al.* [24] proposes a simplified reconstruction that leverage on semantics; the authors jointly optimize the 3D position, the size and the semantic coherence of 3D stixels. However, differently from 3D meshes, stixels are a partial representation of the environment made up by sets of piece-wise planar 3D regions.

In this paper we propose a novel method showing how semantics can be effectively be integrated in a 3D mapping pipeline to simplify the dense point clouds (Section III) before fusing them into a volumetric representation based on 3D Delaunay triangulation (Section II). Fig. 1 shows the entire pipeline, we highlighted in red our contribution. We leverage on semantic image segmentation to understand which part of the scene requires less details, *i.e.*, less

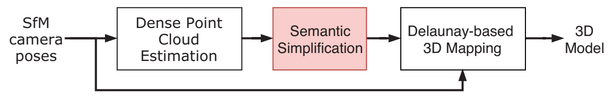


Fig. 1. Dense 3D Mapping pipeline.

points, and which shall be maintained as it is. We propose different heuristic to simplify such subsets of the dense point cloud while still accurately representing the scene. In Section IV we evaluated our method both quantitatively and qualitatively against two publicly available datasets, proving semantic-based simplification approaches are more effective than non-semantic one. Furthermore, our method is able to significantly reduce the number of points in the reconstruction while keeping its accuracy close, and in some case even better, with respect to the mesh estimated using the point cloud. We also tested the simplification method in an incremental setting suitable for classicals vehicle survey scenarios.

II. Delaunay Based Mapping

The baseline algorithm we use to build the 3D model of the scene is a batch version of the method in [9], takes as input a set of cameras, a 3D point cloud, and the camera-to-points visibility to estimate a manifold mesh which is consistent with its visibility constrains. Here we briefly describes how the map is estimated, for more information we refer the reader to [8] and [9]. Usually this class of methods have been applied to Structure from Motion data [7], [8], [9]. In our setting, to obtain a denser and accurate model suitable for optional mesh refinement [1], we densify the input point cloud, combining plane sweeping [25] and stereo matching [26] to produce a dense map DM_i for each camera C_i and to estimate the corresponding dense point cloud.

The mesh reconstruction algorithm creates a 3D Delaunay triangulation of the 3D point cloud, and for each tetrahedron Δ_i it defines μ_i^{free} and μ_i^{matter} representing respectively the free-space and matter weights. Each visibility ray Cp from camera C to point p is traced into the triangulation; for each tetrahedron traversed, the value of μ_i^{free} is incremented by $1 - e^{\frac{d^2}{2\sigma_{\text{free}}}}$, where d is the distance from p to the closest facet of Δ_i, $\sigma_{\text{free}} = 0.05m$. Then, Cp is extended behind the point p by $10\sigma_{\text{matter}}m$ and, for each tetrahedron traversed by this extension, the weights μ_i^{matter} of are incremented by $1 - e^{\frac{d^2}{2\sigma_{\text{matter}}}}$, where $\sigma_{\text{matter}} = 0.01m$

The mesh reconstruction procedure bootstraps from the tetrahedron with the highest free-space vote and it collects iteratively all the nearby free-space tetrahedron such that the manifold property is always fulfilled. One tetrahedron is considered free-space if its votes for free-space are higher than the matter. The boundary between collected and not collected tetrahedra is the reconstructed 3D manifold model.

III. Point Cloud Semantic Simplification

The output of the dense mapping method from Section II is often redundant especially for some specific areas of

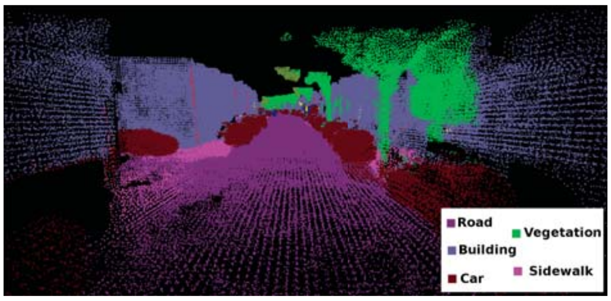


Fig. 2. Point Cloud colored accordingly to the semantics.

the scene; indeed, the ground, which is usually piece-wise planar, ends up to be represented by a relevant number of mesh facets that do not add significant details to the model.

To limit such redundancies and, at the same time, to diminish the number of outliers in the model, we propose to simplify the estimated dense point cloud on which the baseline algorithm builds the 3D Delaunay triangulation instead of post processing the mesh afterward as done in the literature. To this aim we leverage on semantic image segmentation to discriminate which part of the scene is worth simplifying, in our experiments the ground and the walls, and which part requires a detailed point cloud, and therefore is not affected by the simplification procedure.

To estimate the semantic class of each 3D point p_i, we project each point of the dense point cloud, into the segmented image corresponding to the first camera which observed it and we label p_i according to the class of the pixel where the point is projected (see Fig. 2). Although more complex approaches could be employed, *e.g.*, taking the most frequent class among the projection on more images, or weighting the class contribution of each camera depending on the camera-to-point distance, our simple approach resulted effective in all the experiments and we did not find any improvement with complex ones.

The basic idea behind point cloud semantic simplification is to reduce the number of points belonging to a class composed by piece-wise planar surfaces which likely contain redundant information. This decimation procedure can either consider all the points with same semantics at the same time, *i.e.*, globally, or it can be applied region-wise, *i.e.*, locally. In the former and simpler case, it is only possible to discard points uniformly per class. Instead, if we adopt a more flexible region-wise method we are able to adapt the simplification rate depending on the local point density.

We decided for a local approach and we tested our method with both a fixed radius spherical region and an adaptive region defined by means of the K-Nearest Neighbors (KNN) technique. The former is more suitable when the size of the scene is approximately known, while the latter in case different regions show widely different densities. Our approaches are independent on how such areas and neighborhood are defined, therefore, we present here a general region selection algorithm, while in Section IV we explicitly compare the performance of the two approaches, *i.e.*, spherical search and

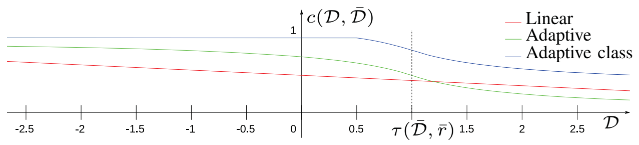


Fig. 3. The conservation factor trend for Linear simplification (with $\bar{c} = 0.5$ and $\tau(\bar{\mathcal{D}},\bar{r})) = 1.0$), Adaptive Simplification ($w = 1$) and (with $\tilde{p} = 0.5$ and $\tau(\bar{\mathcal{D}},\bar{r})) = 1.0$)

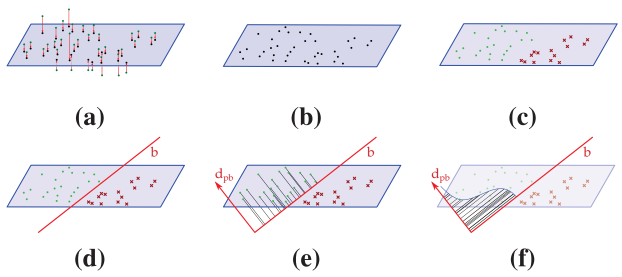


Fig. 4. Step by step procedure to derive P_B for a point P. We project the points on the plane (a-b), we assign a binary label that indicates if a point belongs to the class of the point that generated the region (c) we estimate the separating line (d) we project them to the axis d (e) to compute the point probability of being maintained or discarded (e).

KNN.

For each point p_i we define a region ρ_i either by K-Neighest Neighbors or radius search. Then we define:

$$c(\mathcal{D},\bar{\mathcal{D}},\tilde{p}) \in [0;1], \tag{1}$$

as the *conservation factor*; intuitively it represents the percentage of points in ρ_i maintained after the decimation. We then define the *region density* $\mathcal{D}$ as the number of points inside ρ_i divided by its area. Since we are considering flat surfaces, when ρ_i is computed by radius search, the area is defined as the circle having radius equal to the search radius. When ρ_i is computed with a KNN approach the area is equal to the largest rectangle among the faces of the parallelepiped containing its points. *Average density* $\bar{\mathcal{D}}$ is defined as the average among the density of all the regions belonging to the class of p_i, and $\tilde{p}$ is the percentage of points in the selected area having a different semantics with respect to p_i.

We assume that classes requiring simplification, in our case the ground and the walls, represent surfaces which are mostly flat. For each target class label l, we identify a plane over points labeled as l by means of RANSAC and we define its normal as n_l. Intuitively, we use n_l as the target normal for the flat surface we want to reconstruct. Similarly, we estimate the normal n_i of each point p_i by means of tangent plane approximation [27] to formulate the following ranking function:

$$r(p_i) = (n_i - n_l)^T(n_i - n_l). \tag{2}$$

The closer n_i is to n_l, the smaller $r(p_i)$.

We use *ranking score* $r(p_i)$ to choose which subset of points is kept, *i.e.*, for each region ρ_i we rank all the points p according to the *ranking score* $r(p_i)$ and we discard the last $n = (1 - c) \cdot |\rho_i|$, where $|\rho_i|$ represents the number of points in ρ_i. The outcome of the point simplification depends on how we define the conservation factor c. We here propose four different approaches that we discuss and evaluate in the experimental section (Section IV).

Linear Simplification (LS): The simplest simplification method is based on a linear conservation factor; it is based on the region and average densities as follows:

$$c(\mathcal{D},\bar{\mathcal{D}}) = -\frac{\bar{\mathcal{D}}}{\bar{c}} * (\mathcal{D} - \bar{\mathcal{D}}) + \bar{c}, \tag{3}$$

where $\bar{c}$ is the target conservation factor for an area with average density (*e.g.*, red line in Fig. 3). Lets consider for instance $\bar{c} = 0.4$; the idea of LS is to keep around 40% of the points in the areas where the density is similar to the average density and to retain more than 40% of points for less dense areas.

Adaptive Simplification (AS): To have a higher simplification where the point density diverges with respect to the average of the point cloud density, in Adaptive Simplification we use a function f similar to the sigmoid function (green curve in Fig. 3):

$$f(x) = 0.5 + \frac{wx}{2(1 + w|x|)}, \tag{4}$$

where w is a stretching factor: a small w corresponds to larger difference between two similar densities. We formalize the conservation factor with:

$$c(\mathcal{D},\bar{\mathcal{D}}) = f(\mathcal{D} - \tau(\bar{\mathcal{D}},\bar{r})). \tag{5}$$

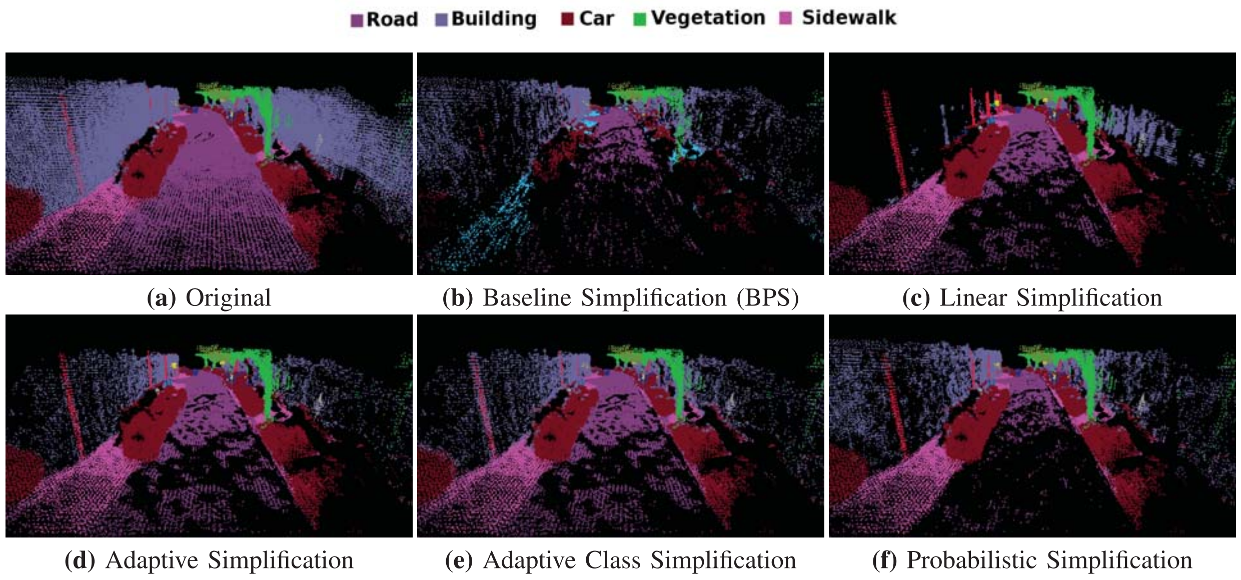
where $\tau(\bar{\mathcal{D}},\bar{r})$ is defined such that $f(\bar{\mathcal{D}}) = \bar{c}$.

Adaptive Class Simplification (ACS): When the region ρ_i contains points belonging to different classes, we might prefer to limit the simplification since it is more likely that the region is close to the boundary of two or more semantic regions. The points near to the class boundaries may correspond to changes in surface orientation; therefore, they are useful to define the right shape of the reconstructed model. To do so, with *Adaptive Class Simplification*, we remove the points depending on the semantically different points contained in ρ_i. In this case we define c by means of $\mathcal{D}$, $\bar{\mathcal{D}}$ and $\tilde{p}$, *i.e.*, the percentage of the points belonging to other classes with respect to the one p_i belongs to. Equation (5) becomes:

$$c(\mathcal{D},\bar{\mathcal{D}},\tilde{p}) = (1 + \tilde{p})f(\mathcal{D} - \tau(\bar{\mathcal{D}},\bar{r})), \tag{6}$$

when f is close to 1 and $\tilde{p}$ is not zero, c might exceed 1, to avoid this issue we saturate it imposing $c = \min(1, c(\mathcal{D},\bar{\mathcal{D}},\tilde{p}))$. Intuitively, when $\tilde{p}$ is small the area contains few points of other classes, hence it slightly increases c and the behaviour is similar to the adaptive simplification. On the other hand, when $\tilde{p}$ is close or above 0.5, *i.e.*, the center of the region is close to the boundary, c is largely increased, and consequently we remove significantly less points. For instance in Fig. 3 we plot the trend of Equation (6) when $\tilde{p} = 0.5$.

Probabilistic simplification (PS): Differently from previous approaches we define the probability P_c that a point p_i is conserved. Moreover, while in previous methods we



(a) Original **(b)** Baseline Simplification (BPS) **(c)** Linear Simplification

(d) Adaptive Simplification **(e)** Adaptive Class Simplification **(f)** Probabilistic Simplification

Fig. 5. Street perspective of the KITTI point clouds.

extracted a region for each point, in this case, to limit the computational burden, we compute a predefined number of regions N_r experimentally fixed to $\frac{|\rho|}{8}$, where $|\rho|$ is the size of the cloud. The regions are defined around N_r points randomly chosen among those belonging to the class l we are going to simplify. Even if in principle this choice may leave some area uncovered by the simplification method, experiments showed that in practice this issue is not statistically relevant.

We define P_c as the composition of two probabilities:

$$P_c(p|\tilde{p}, \mathcal{D}, \bar{\mathcal{D}}, \rho_i) = \\ i(\tilde{p}) \cdot P_I(p|\mathcal{D}, \bar{\mathcal{D}}) + (1 - i(\tilde{p})) \cdot P_B(p|\mathcal{D}, \bar{\mathcal{D}}, \rho_i), \quad (7)$$

where $i(\tilde{p})$ is an indicator function defined as follows:

$$i(\tilde{p}) = \begin{cases} 0 & \tilde{p} \geq 0.1 \\ 1 & \text{otherwise} \end{cases}. \quad (8)$$

Intuitively, when the percentage of points with a different semantics is less then 10%, *i.e.*, when the region ρ_i is far from the inter-class boundary, only the probability P_I is active, otherwise, *i.e.*, when ρ_i is close to the boundary, $P_c(p|\tilde{p}, \mathcal{D}, \bar{\mathcal{D}}, \rho_i) = P_B$.

The probability P_I represents the idea that the closer a point is to the center of a flat surface, the lesser it is useful. To formalize P_I we project every point $p \in \rho_i$ to the plane π_l, having normal n_l. For every point we then used a gaussian distribution, $\mathcal{N}$, to estimate the importance of the point in terms of probability. Specifically, we define:

$$P_I(p|\mathcal{D}, \bar{\mathcal{D}}) = 1 - \mathcal{N}(\mu, \Sigma), \quad (9)$$

where $\mathcal{N}$ is the normal probability distribution depending on μ, *i.e.*, the projection of p_i on π_l and Σ, *i.e.*, a diagonal matrix formalized as:

$$\Sigma = \begin{bmatrix} \tilde{d} \cdot \sigma^2 & 0 \\ 0 & \tilde{d} \cdot \sigma^2 \end{bmatrix}, \quad (10)$$

where $\tilde{d}$ is the ratio of $\mathcal{D}$ and $\bar{\mathcal{D}}$, and σ is an a-priori defined standard deviation.

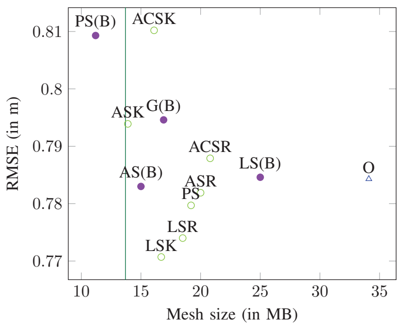
When a region is close to the class boundaries it may happen that a point in the center of the region is near to the boundary. In this case we want to keep such points. For this reason we need to explicitly detect the boundary and then remove points far from it. To this extent we formulate P_B. We consider all the points inside the region ρ_i and we project them on the plane π_l fitted to the points (Fig. 4(a-b)). Let l be the class considered; locally the points belonging and not belonging to l are usually separated by a line $\mathbf{b}$ which is roughly the class boundary (Fig. 4(c)). We find $\mathbf{b}$ as a linear classification problem through a Support Vector Machine (SVM) with linear kernel (Fig. 4(d)). For each point p we compute the distance $d_{p\mathbf{b}}$ from the estimated boundary $\mathbf{b}$ and we compute the probability of point conservation $P_B(p)$ (Fig. 4(f)) as:

$$P_B(p) = P_B(d_{p\mathbf{b}}) \sim \mathcal{N}(0, \sigma^2), \quad (11)$$

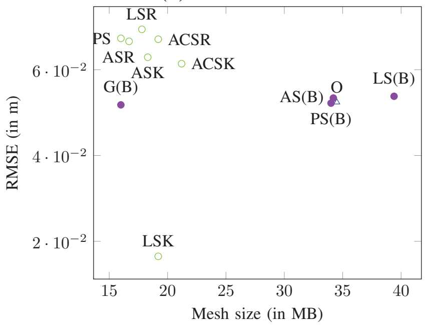
the closer a point to the boundary the more important it is. Therefore, to assign the maximum importance to points close to the boundary we center the Gaussian distribution on the line, *i.e.*, with distance 0. The point importance decreases as the probability. Once P_c is known, we remove a point from the cloud if, extracting a random number, it is greater than $P_c(p|\tilde{p}, \mathcal{D}, \bar{\mathcal{D}}, \rho_i)$.

IV. Experiments

To prove the effectiveness of the proposed semantic-based simplification methods, and demonstrate semantic approaches are superior to non-semantic one, we test them against several baselines which do not rely on semantics. In the first group of baselines we consider the whole point cloud as if of single class and we simplify with the methods proposed in Section III, *i.e.*, without semantics. The Linear



(a) KITTI



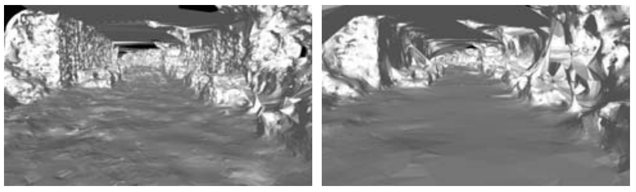
(b) fountain-P11

Fig. 6. Mesh statistics for each method: in blue the original model (O), in purple the baselines and in green the proposed methods.

and Adaptive Simplifications act on the point cloud purely based on the region density, while in the Probabilistic method the P_B term is neglected since there are no more boundaries. A second baseline is the purely geometric approach proposed in [28], which, by relying on surface curvature, focuses the simplification on flat surfaces.

A. *Simplification Results*

We evaluated our simplification methods against the sequence 95 of the KITTI dataset [29] and the fountain-P11 of EPFL dataset [30]. In the former case, stereo pairs are provided, therefore we generate a dense point cloud by means of semi-global block matching [26]; in the latter case, we adopt the plane sweeping algorithm [25] to the 11 high-resolution images provided. Then, for each image we choose to predict the semantic segmentation by means of ReSeg [21] for the KITTI dataset and MultiBoost [31] for the fountain dataset (since a very limited set of images is available). In principle whatever image segmentation method is applicable. In the experiments we simplify these two semantic point clouds and we evaluate how this impacts the accuracy of the recovered surfaces. In particular, for the KITTI dataset, we evaluate the accuracy of the reconstructed model, for each camera, by comparing the points of the Velodyne projected into the image plane with the corresponding depth map generated by the 3D model. For the fountain-P11 dataset we follow the evaluation procedure presented in the paper [30] and we compare the depth map generated by the dense ground truth point cloud and by the 3D model. The accuracy measure we use is the Root Mean Squared Error (RMSE) of



(a) Original **(b)** Simplified

Fig. 7. Reconstruction of the KITTI dataset. (a) shows the original reconstruction, while (b) the simplified version using the Probabilistic simplification. Let notice, how more smooth the simplified version is.

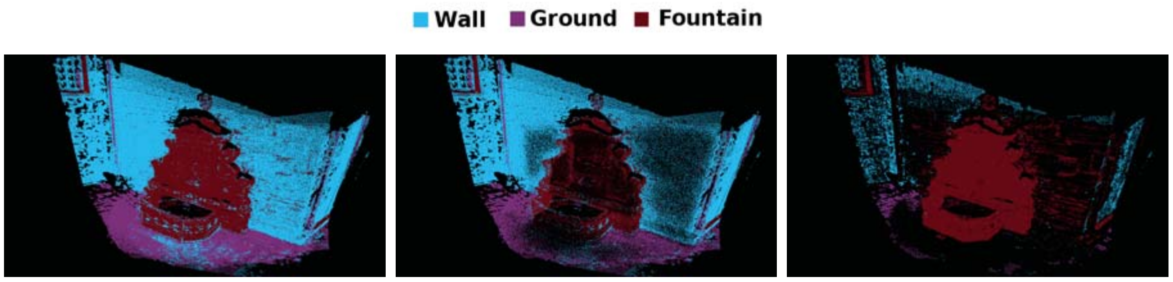
the depth values (where depths are valid).

Fig. 6(a) shows the statistics about the reconstructed meshes in the KITTI dataset in terms of disk occupancy (the mesh is stored as an OFF file without compression) with respect to the reconstruction accuracy. The blue mark represents the original mesh (O) the green marks represent the proposed methods, *i.e.*, LS, AS, ACS, PS, where the final letter K or R, denotes the search method used, *i.e.*, KNN or Radius. Finally, the purple points are the baselines: Baseline LS, AS, PS and the baseline based on geometry curvature (BG). Both baselines and proposed approaches significantly reduce the size of the reconstruction with respect to the original. In particular, the number of points belonging to ground and walls, *i.e.*, the two classes involved in the simplification procedure, decreases up to two orders of magnitude. However, while the accuracy of the baselines is in general slightly worse than the original, the proposed linear and probabilistic methods consistently improve the accuracy of the reconstruction. Indeed, they provide a good trade-off between mesh over-simplification and the removal of redundant and noisy data (see Fig. 7).

In Fig. 6(a) the green vertical line represents the size achievable by removing all the ground and wall points. Therefore, a class-aware method too close to the line is over-simplifying, whilst a method too distant from it is under-simplifying. Since the baselines simplifies the whole mesh and therefore are not limited to ground and wall, they can cross this line.

As a qualitative comparison, we refer the reader to the supplementary video and we illustrate in Fig. 5 the original point cloud colored according to semantics, compared with the Baseline Probabilistic Simplification (BPS) and the other proposed simplification approaches. As expected, BPS decimates the point cloud evenly over the entire scene while PS, Fig. 5(f), is able to keep the details of the trees and the cars and it significantly simplifies the walls and the ground. For this reason the 3D mapping from Probabilistic Simplification point cloud is sensibly better than the one obtained via BPS which suffers from over-smoothing.

We tested our approaches also with the fountain-P11 dataset and Fig. 6(b) shows the reconstruction statistics: the Linear simplification method achieves the best result, while the other proposed approaches achieve similar performances among each other and they are close to the Baseline Geometric simplification. Other baseline methods are overcome. In



(a) Original **(b)** Baseline Simplification (BPS) **(c)** Probabilistic Simplification

Fig. 8. Frontal perspective of the fountain-P11 point clouds.

this case the dataset is significantly different with respect to KITTI, especially because the fountain contains many details and its density is much higher than the ground and the wall. Then, even if a simplification procedure is applied to that region, the mapping accuracy does not drop significantly.

Therefore, baselines can simplify the highly redundant surface of the fountain, without virtually loss in accuracy, while the proposed method simplifies only walls and ground. To have a fairer comparison, we tested PS combined with a random simplification over the fountain. We randomly retained 40% of the points composing the fountain. The proportion has been chosen accordingly to the percentage of points retained by the BG. As final model we obtained a smaller 3D model, $13MB$ while both PS and BG originally achieved around $16MB$, furthermore modified PS accuracy reaches $0.0655m$ RMSE error, which is similar to BG and PS.

Regarding the region selection method, radius search, proved to be easier to tune and to use in both scenarios with respect to KNN, indeed a good initial radius search can be estimated knowing the size of the cloud to be simplified. KNN strictly depends on the density of the cloud: a very dense cloud, as fountain-P11, requires a very high K.

B. *Incremental Point Cloud Simplification*

The main goal of point simplification is to improve the trade-off between accuracy and speed, therefore a typical target application is to improve the performance of an incremental mapping algorithm as [9], which runs in real time on a CPU by using Structure from Motion points, but is not able to handle very dense point clouds still maintaining realtime performance. To test the effectiveness of our proposal in this scenario, we added the dense point estimation and our Probabilistic Simplification module to the algorithm proposed in [9], which is the incremental version of the pipeline described in Section II.

We tested the sequences 95 and 104 of the KITTI dataset [29], and we compute the model accuracy in the same way. The semantic segmentation are computed using [32] pre-trained over ADE20K dataset (only 11 among the 150 classes have been used).

In Table I we compare the incremental mapping performance from dense point clouds with or without the proposed simplification. The accuracy is measured with the RMSE adopted in the previous section, while the speed is the total number of frames of the sequence over the seconds spent to build all the mesh incrementally. The results show how the simplification method is able to keep a similar accuracy while improving the frame rate.

TABLE I
INCREMENTAL MAPPING RESULTS.

Sequence	Original		Incremental Simplification	
	Accuracy RMSE(m)	Frame rate (f/s)	Accuracy RMSE(m)	Frame rate (f/s)
95	**0.7135**	1.0082	0.7173	**1.5434**
104	**0.7775**	0.4708	0.7846	**0.6397**

Speed-up depends on the scene composition. Sequence 104 contains many elements which are not addressed by the simplification procedure, *e.g.*, trees, vegetation and cars, therefore the frame rate increase is limited. Instead, sequence 95 is an heterogeneous scenes, therefore the frame rate increases of more than 50%.

V. CONCLUSIONS

In this paper, we leverage on semantics to simplify redundant regions of a dense point cloud prior to build a 3D mesh upon to its points. We fuse these points into a Delaunay triangulation which easily adapts to the point cloud even if we decrease the density of some regions, and we extract a visibility consistent mesh. We designed four methods which provide different alternatives to decimate the point cloud, halving its size without relevant loss in accuracy. Among these methods the probabilistic approach in general is easier to tune and proved to be able to identify a good trade off between simplification and map accuracy. We have also shown how a state-of-the-art incremental mapping method can benefit from the proposed semantic point simplification to process less redundant data and to consequently speed-up the mapping. As future work, we plan to evaluate viable energy efficient optimizations [33] for the proposed simplification procedure and we plan to design a flexible method able to deal with every semantic class, *e.g.*, by identifying a per class simplification probability.

REFERENCES

[1] Vu, H.H., Labatut, P., Pons, J.P., Keriven, R.: High accuracy and visibility-consistent dense multiview stereo. IEEE Transactions on Pattern Analysis and Machine Intelligence **34**(5) (2012) 889–901

[2] Zach, C., Pock, T., Bischof, H.: A globally optimal algorithm for robust tv-l 1 range image integration. In: Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, IEEE (2007) 1–8

[3] Savinov, N., Häne, C., Ladicky, L., Pollefeys, M.: Semantic 3D reconstruction with continuous regularization and ray potentials using a visibility consistency constraint. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 5460–5469

[4] Schönberger, J.L., Zheng, E., Pollefeys, M., Frahm, J.M.: Pixelwise view selection for unstructured multi-view stereo. In: European Conference on Computer Vision (ECCV). (2016)

[5] Galliani, S., Lasinger, K., Schindler, K.: Massively parallel multiview stereopsis by surface normal diffusion. In: The IEEE International Conference on Computer Vision (ICCV). (June 2015)

[6] Labatut, P., Pons, J.P., Keriven, R.: Efficient multi-view reconstruction of large-scale scenes using interest points, delaunay triangulation and graph cuts. In: Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, IEEE (2007) 1–8

[7] Litvinov, V., Lhuillier, M.: Incremental solid modeling from sparse and omnidirectional structure-from-motion data. In: BMVC. (2013)

[8] Romanoni, A., Matteucci, M.: Incremental reconstruction of urban environments by edge-points delaunay triangulation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE (2015) 4473–4479

[9] Piazza, E., Romanoni, A., Matteucci, M.: Real-time cpu-based large-scale three-dimensional mesh reconstruction. IEEE Robotics and Automation Letters **3**(3) (2018) 1584–1591

[10] Romanoni, A., Ciccone, M., Visin, F., Matteucci, M.: Multi-view stereo with single-view semantic mesh refinement. In: Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW). (2017) 706–715

[11] Zoni, D., Flich, J., Fornaciari, W.: Cutbuf: Buffer management and router design for traffic mixing in vnet-based nocs. IEEE Transactions on Parallel and Distributed Systems **27**(6) (2016) 1603–1616

[12] Zoni, D., Colombo, L., Fornaciari, W.: Darkcache: Energy-performance optimization of tiled multi-cores by adaptively power-gating llc banks. ACM Trans. Archit. Code Optim. **15**(2) (May 2018) 21:1–21:26

[13] Li, S., Siu, S.Y., Fang, T., Quan, L.: Efficient multi-view surface refinement with adaptive resolution control. In: European Conference on Computer Vision, Springer (2016) 349–364

[14] Lafarge, F., Keriven, R., Brédif, M., Vu, H.H.: A hybrid multiview stereo algorithm for modeling urban scenes. IEEE transactions on pattern analysis and machine intelligence **35**(1) (2013) 5–17

[15] Wu, C., Agarwal, S., Curless, B., Seitz, S.M.: Schematic surface reconstruction. In: Computer Vision and Pattern Recognition (CVPR), IEEE (2012) 1498–1505

[16] Gallup, D., Pollefeys, M., Frahm, J.M.: 3D reconstruction using an n-layer heightmap. In: Pattern Recognition. Springer (2010) 1–10

[17] Schindler, F., Wörstner, W., Frahm, J.M.: Classification and reconstruction of surfaces from point clouds of man-made objects. In: Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on, IEEE (2011) 257–263

[18] Bourki, A., de La Gorce, M., Marlet, R., Komodakis, N.: Patchwork stereo: Scalable, structure-aware 3d reconstruction in man-made environments. In: Winter Applications of Computer Vision (WACV), 2017 IEEE Conference on, IEEE (2017) 292–301

[19] Holzmann, T., Oswald, M.R., Pollefeys, M., Fraundorfer, F., Bischof, H.: Plane-based surface regularization for urban 3d reconstruction. In: 28th British Machine Vision Conference. (2017)

[20] Osman Ulusoy, A., Black, M.J., Geiger, A.: Patches, planes and probabilities: A non-local prior for volumetric 3d reconstruction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 3280–3289

[21] Visin, F., Ciccone, M., Romero, A., Kastner, K., Cho, K., Bengio, Y., Matteucci, M., Courville, A.: ReSeg: A recurrent neural network-based model for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. (2016) 41–48

[22] Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence **40**(4) (2018) 834–848

[23] Lin, G., Milan, A., Shen, C., Reid, I.: Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In: IEEE conference on computer vision and pattern recognition (CVPR). Volume 1. (2017) 3

[24] Schneider, L., Cordts, M., Rehfeld, T., Pfeiffer, D., Enzweiler, M., Franke, U., Pollefeys, M., Roth, S.: Semantic stixels: Depth is not enough. In: Intelligent Vehicles Symposium (IV), 2016 IEEE, IEEE (2016) 110–117

[25] Häne, C., Heng, L., Lee, G.H., Sizov, A., Pollefeys, M.: Real-time direct dense matching on fisheye images using plane-sweeping stereo. In: 3D Vision (3DV), 2014 2nd International Conference on. Volume 1., IEEE (2014) 57–64

[26] Hirschmüller, H.: Stereo processing by semiglobal matching and mutual information. IEEE Transactions on pattern analysis and machine intelligence **30**(2) (2008) 328–341

[27] Rusu, R.B.: Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments. PhD thesis, Computer Science department, Technische Universitaet Muenchen, Germany (October 2009)

[28] Cignoni, P., Montani, C., Scopigno, R.: A comparison of mesh simplification algorithms. Computers & Graphics **22**(1) (1998) 37–54

[29] Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the KITTI vision benchmark suite. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE (2012) 3354–3361

[30] Strecha, C., von Hansen, W., Van Gool, L., Fua, P., Thoennessen, U.: On benchmarking camera calibration and multi-view stereo for high resolution imagery. In: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, IEEE (2008) 1–8

[31] Webb, G.I.: Multiboosting: A technique for combining boosting and wagging. Machine learning **40**(2) (2000) 159–196

[32] Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ade20k dataset. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2017)

[33] Zoni, D., Fornaciari, W.: Modeling dvfs and power-gating actuators for cycle-accurate noc-based simulators. J. Emerg. Technol. Comput. Syst. **12**(3) (September 2015) 27:1–27:24