PLANAR POLYGONS DETECTION IN LIDAR SCANS BASED ON SENSOR TOPOLOGY ENHANCED RANSAC

Stéphane A. Guinard[1], Zoumana Mallé[1], Oussama Ennafii[1], Pascal Monasse[2], Bruno Vallet[1]*

[1] LASTIG, Univ Gustave Eiffel, ENSG, IGN, F-94160 Saint-Mandé, France - firstname.lastname@ign.fr
[2] LIGM, École des Ponts, Univ Gustave Eiffel, CNRS, Marne-la-Vallée, France - pascal.monasse@enpc.fr

Commission II, WG II/3, II/4ICWG

KEY WORDS: Lidar processing, 3D reconstruction, plane detection, planar polygons extraction

ABSTRACT:

Detecting planar structures in point clouds is a very central step of the point cloud processing pipeline as many Lidar scans, in particular in anthropic environments, present such planar structures. Many improvements have been proposed to RANSAC and the Hough transform, the two major types of plane detection methods. An important limitation however is that these methods detect planes running across the whole scene instead of more localized planar patches. Moreover, they do not exploit the sensor information that often comes with Lidar point cloud (sensor topology and optical center position in particular). In this paper we address both issues: we aim at detecting planar polygons that have a limited spatial extent, and we exploit sensor topology. The latter is used to enhance a RANSAC framework on two aspects: to make seed points selection more local and to define more compact sets of inliers through sensor space region growing.

1. INTRODUCTION

Plane detection in point clouds is a widely researched topic in the geometry processing, photogrammetry, and computer vision communities. The problem is central to scene interpretation, polyhedral reconstruction and structure extraction (Pu et al., 2011), as in many anthropic structures (inside and outsides of all types of buildings in particular). Reconstruction of piecewise planar objects has thus received an important attention over the past two decades. Recent approaches use either global plane detection or more local planar patch segmentation. However both approaches are still limited. Global plane detection is usually used to create a plane arrangement whose complexity increases combinatorially with the number of planes, thus not scaling up. Local approaches become a topological nightmare when trying to reconnect all the individual pieces.

1.1 Previous Works

Detecting planar shapes in a point cloud is a problem of fitting possibly multiple instances of given primitives to data points. Historically, the two main approaches to solve this problem have been the Hough transform and RANSAC.

The Hough transform (Borrmann et al., 2011) (Knopp et al., 2010) uses a voting principle: it defines a parameter space (the space of parameters of the primitive of interest) and find the interesting primitives as accumulation maxima in this space. It was applied to architectural modeling in (Chen and Chen, 2008).

RANdom SAmple Consensus (RANSAC) is a stochastic greedy search where primitives are sampled by selecting the appropriate number of points to define them from the point cloud. The primitives with most inliers (sufficiently close supporting data points) are selected in a greedy manner (Schnabel et al., 2007). RANSAC is very popular and used in a number of polyhedral/building reconstruction approaches (Monszpart et al., 2015) (Nan and Wonka, 2017) to detect planes used to build an arrangement in which the final polyhedra will be searched. Such methods do not scale up well in practice as the size of the plane arrangement grows combinatorially when the number of detected planes increases.

To solve this problem, several works (including ours) have been interested in finding compact planar primitives. This is usually done by region growing (Rabbani et al., 2006), inspired by k-means for instance (Cohen-Steiner et al., 2004). More recent approaches have suggested to adapt a graph clustering algorithm (Landrieu and Obozinski, 2017) to find planar patches in a more global optimization framework (Guinard et al., 2019). Another way to use an optimization approach to the multi model fitting is to pose the problem as a model selection (Isack and Boykov, 2012). Finally, a method was recently proposed to explore structural scales by shape collapsing (Fang et al., 2018).

1.2 Data

The data used to experiment our method is a Mobile Lidar Scan (MLS) acquired with the Stéréopolis II Mobile Mapping System (MMS) (Paparoditis et al., October 2012). Our method exploits the sensor topology inherent to such MLS acquisition, that is often lost during export. This paper advocates that this information is very useful to speed up and enhance the quality of planar patches detection. To explain the concept of sensor topology, we need to give some more details on the MLS acquisition.

The used LiDAR scanner is a RIEGL VQ-250 that rotates at 100 Hz and emits 3000 pulses per rotation which corresponds to an angular resolution around $0.12°$. For each emitted pulse, 0 to 8 echoes are recorded, producing an average of 250 000 points per second in typical urban scenes. The sensor records information for each pulse (direction (θ, ϕ), time of emission) and echo (amplitude, range, deviation).

*Corresponding author

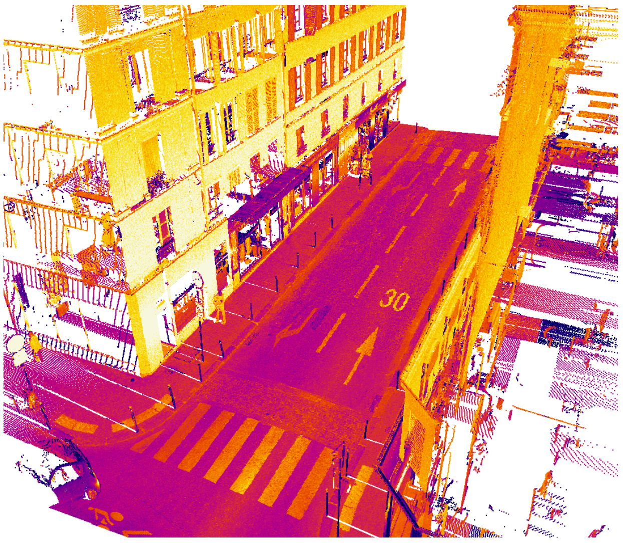

Figure 1. A view of the Mobile Laser Scan (MLS) used in this study, colored with reflectance.

The Stereopolis II mobile mapping system is also composed of an Applanix POS-LV220 Inertial Navigation System (INS) combining D-GPS, an Inertial Measurement Unit (IMU) and an odometer. This system outputs the reference frame of the INS in geographical coordinates at 100 Hz. However, the GPS masks that frequently occur in urban areas induce drifts that can reach one meter for a 2 minutes mask. This initial georeferencing is corrected based on tie points with aerial photography, resulting in around 10 cm standard deviation in planimetry, and 15 cm in altimetry. The Lidar sensor is calibrated in order to recover the transformation between the sensor and Applanix coordinate systems which allows to transform the coordinates from the sensor frame to the INS frame then to a world coordinate system.

Combining all this information, a point cloud in world coordinates can be constructed. In order to decrease memory footprint, this point cloud is often expressed by (x, y, z) coordinates in a local georeferenced frame, which allows storing them as 32 bit floats without the loss of precision that would result if absolute coordinates were used (often in the order of 10^6).

The resulting scan is very anisotropic because while points are very dense along scanlines (a few millimeters on the road below the sensor), scanlines can be separated by several centimeters according to the vehicle speed (5 cm at a typical acquisition speed of 5 m/s = 18 km/h). In addition to the (x, y, z) coordinates (in sensor space), the sensor records multiple information for each pulse (direction, time of emission) and echo (amplitude, range, deviation). The amplitude being dependent on the range, it is corrected into a relative reflectance. This is the ratio of the received power to the power that would be received from a white diffuse target at the same distance expressed in dB. The reflectance represents a range independent property of the target and is used to color the MLS point cloud displayed in Figure 1. This scan was performed in the city of Paris and the resulting point cloud is stored in individual blocks consisting of 1 second of acquisition each, whose moderate size (300 000 pulses, 250 000 echoes) is well suited to display and processing.

While standard export chains produce sets of (x, y, z) information with per point attributes, we modified the export to keep all the sensor information. Our data structure has two core objects: pulses and echoes, each carrying its own attributes (time of emission and θ, ϕ angle for pulses, range, amplitude, reflect-

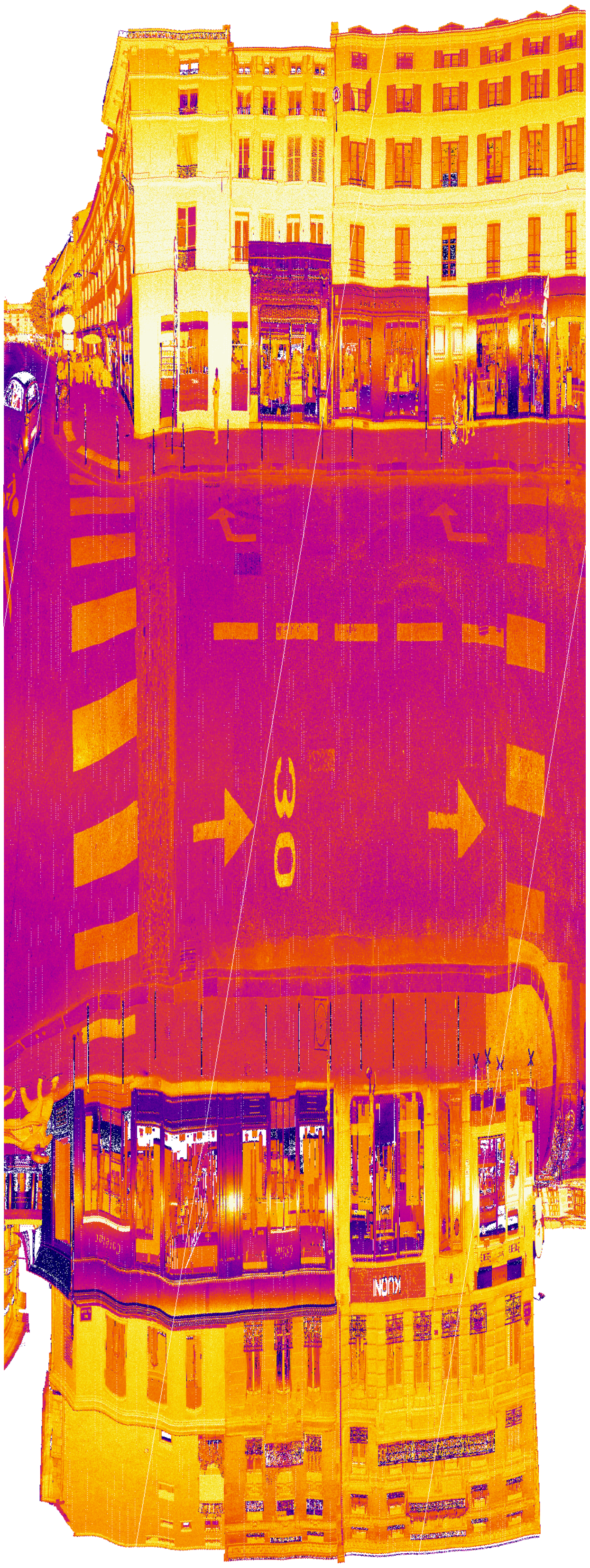

Figure 2. A view of the same Mobile Laser Scan (MLS) colored with reflectance in sensor space (horizontally: time, vertically: θ). For multiple echoes, the last is used. For no returns, we keep the white background color.

ance and deviation for echoes). Because pulses are emitted regularly (at 300 kHz) they can be stored in the regular structure of a vector allowing to define for each pulse the next and previous pulses; whereas, since we have also access to the θ angle, the angular position of the Lidar beam rotation, we can find the neighbors in the previous and next scan rotations. In practice, the number ppr of pulses per full 2π rotation of the laser beam is not an integer, so each pulse has two neighbors in each neighbor rotation. A pulse of index p has thus six neighbors: the previous and next pulses in the same rotation of respective index $p-1$ and $p+1$; two neighbors in the previous rotation of index $p - \lfloor ppr \rfloor - 1$ and $p - \lfloor ppr \rfloor$; two neighbors in the next rotation of index $p + \lfloor ppr \rfloor + 1$ and $p + \lfloor ppr \rfloor$.

This means we can create a regular hexagonal 2D pulse structure for the point cloud, represented in Figure 2. Considering echoes, our implementation allows to access the pulse corresponding to each echo and the echoes corresponding to each pulse. So, we define as neighboring echoes of an echo e all the echoes whose pulse is a neighbor of the pulse of e. Note that they are not necessarily geometric neighbors as adjacent pulses can generate echoes at very different ranges. We call these neighbors the sensor neighborhood, defining the sensor topology, and we will make abundant use of it in the paper and demonstrate its utility.

1.3 Overview

In this paper, we propose several improvements to the basic RANSAC approach to adapt it to planar polygon extraction and increase both its quality and efficiency. Our main contributions are:

- Adapting RANSAC to produce planar polygons instead of planes, which are more compact and represent a better abstraction of the data, as we ensure that all the surface of an extracted polygon is supported by data points.
- Exploiting the sensor topology to optimize the sampling strategy, greatly improving performance.
- Proposing a dynamic criterion to automatically adapt the number of iterations to the current best primitive.
- Exploiting sensor topology to define more compact sets of inliers through region growing, also accelerating the inlier computation, which is the bottleneck of RANSAC.

The paper is structured as follows: Section 2 presents the usual RANSAC method and the various improvements we propose. Experimentation and evaluation are provided in Section 3. Finally, conclusions are drawn and perspectives proposed in Section 4.

2. METHOD

Our method is based on several improvements to the classical RANSAC algorithm. We begin by recalling this simple algorithm, then detail our improvements.

2.1 RANSAC

RANSAC is a very popular object extraction algorithm. It is robust to noise and outliers, simple, efficient and very general. It consists in sampling possible objects and greedily keeping the best ones. What we detail here is its application to multiple plane detection in point clouds.

1. Randomly select triplets of (non aligned) points P_i, P_j and P_k (random samples) in the point cloud. Each point triplet defines a plane $\mathcal{P}$ passing through the three points and oriented by its normal:
$$\vec{n}(\mathcal{P}) = \frac{\overrightarrow{P_iP_j} \wedge \overrightarrow{P_iP_k}}{||\overrightarrow{P_iP_j} \wedge \overrightarrow{P_iP_k}||} \qquad (1)$$
2. Compute how many points (called inliers) are close to the plane passing through these 3 points. More formally, the condition for a point P to be an inlier is:
$$d(P, \mathcal{P}) = |\overrightarrow{P_i\mathcal{P}}.\vec{n}(\mathcal{P})| < t_{\text{in}}, \qquad (2)$$
where t_{in} is the inlier distance threshold.
3. Iterate 1-2 n_{it} times, add the best plane (with most inliers) to the solution set, and remove the inliers from the point cloud.
4. Iterate 1-3 while the number of inliers of the best plane is above $n_{\min}$, the minimum number of inliers to justify a plane detection.

This simple version of plane RANSAC has only three parameters:

1. The inlier distance threshold t_{in}. The choice of t_{in} depends on the noise but also the expected deviation of real planar structures from perfect planes, or in other terms on the expected generalization level, as structures larger than this threshold are lost.
2. The number of RANSAC iterations n_{it}: the more iterations the more likely we are to find the best plane in the remaining points, but the longer the computation time.
3. The minimum number of points on a selected plane $n_{\min}$, the stopping criterion. This parameter controls how many planes are output (the lower, the more planes) as it defines the detection sensitivity.

These parameters should often be tuned for each dataset in order to obtain satisfactory results. In particular:

1. n_{it} should be constantly adapted since the first primitives much larger than the stopping criterion $n_{\min}$ are found easily in a few iterations while smaller ones closer to the threshold may require many more iterations to be found. However, the total number of points decreases as points from previously found primitives are removed (see Section 2.5).
2. $n_{\min}$ is hard to tune on point clouds with varying density and on scenes with planar patches of very variable size, as encountered in MLS. We propose in Section 2.2 a solution to the first problem relying on sensor topology.

In addition to parameter tuning issues, we tackle specifically:

1. The sampling problem: planar patches are local, while the default RANSAC sampling strategy samples points across the whole scene, with low probability that the sampled plane corresponds to a local planar structure of the scene. We propose in Section 2.3 to solve this problem by a local sampling in sensor topology.

2. The locality problem: a plane is not a local structure, so points anywhere in the scene can be attributed coincidentally to a planar structure localised in another part of the scene. We propose in Section 2.3 a sensor space region growing strategy to ensure compactness of the planar patches.

2.2 Surface Weighting

The scan resolution or density (defined as points per m^2 of scene surface for instance) is varying in MLS, as the scanned objects can be at very different distances and viewed under different angles. Moreover, the density depends on the speed of the vehicle and its rotation. All this makes tuning the $n_{\min}$ parameter very difficult, and requires this tuning to be adjusted on each dataset. An easier approach would be to consider a minimum patch surface, as this would only depend on the scene geometry.

An obvious solution would be to use a surface mesh reconstruction algorithm, but this is often very time and memory consuming. Instead, we propose a simple estimate of the scene surface corresponding to each point using the sensor topology. We use the 6 neighboring pulses to build 6 triangles around each echo. In case of multiple echoes, we choose the closest. In case the neighbor pulse has no return, we do not build the triangles. To avoid very high weighting of depth discontinuities, we also remove triangles with a circumradius above a threshold set at 50 cm in our experiments as done in (Guinard and Vallet, 2018). Finally, because each triangle is shared by 3 echoes, we divide the result by 3 to get a good estimate of the area of the underlying planar patch.

Once these surface weights are computed, we simply replace the score of each planar patch (number of inliers) with the sum of inlier weights and replace the threshold $n_{\min}$ by a minimum patch area $A_{\min}$.

2.3 Neighborhood Sampling

The first improvement we propose is sensor topology sampling. Most point cloud generation technologies allow defining a 2D neighborhood (which point is after and before a point in lines and columns). Image depth maps have this topology from the image lines and columns. Fixed Lidar stations discretise their surrounding in θ and ϕ also leading to a regular grid structure (a depth map in spherical coordinates). Planar Lidars acquire points with a rotating ray, allowing to define the previous and next point, but also the corresponding point in the previous and next rotations, as the point with the closest angle. If the information is not available, we can define a neighborhood based on closest distance points, which only adds a computing cost, which is rather limited thanks to acceleration structures such as Kd-trees (Bentley, 1975).

Because we are looking for compact planar patches and not planes across the whole point cloud, selecting our three samples randomly will generate a large number of planes with few inliers. Because we have local planar patches in the data, we only need to sample points in a relatively small neighborhood in order to find them. For 2D sensor neighborhoods, they can be defined as rectangular windows in the grid structure, for distance neighborhoods, the kNN (k nearest neighbors) can be used. In practice, we replace step (1) by first selecting a random point in the whole cloud, then two other points in its neighborhood.

2.4 Sensor Region Growing

An important limitation of RANSAC is that it does not scale very well, as each iteration requires to compute the distances of all the points of the cloud to the sampled plane. So if we consider that the number of extracted planes grows as the number of points, the computing time grows quadratically. It is in fact even worse if we do not use neighborhood sampling because RANSAC will require more iterations to find the same planes when more data points are added. We propose to make this step more local by growing a region from the first sample. We use the same definition for neighborhood as before (possibly with a different size) and only test the plane distance for the seed neighbors, then iteratively adding neighbors only for inliers. This way, points that fall by chance on this plane in another part of the scan will not be affected to the plane. This allows for both more compact planar point sets extraction (the inliers of the returned planar patches) and faster computing times as much less distance computation is required.

2.5 Dynamic Iteration

Choosing the number of RANSAC iterations is not easy. If the compromise is clear (more iterations gives more optimality guarantees for more computing time), the choice is not easy as larger structures require few iterations to be found, but the smaller ones require more. Moreover, the number of points evolves as the previous inliers of the best planes are removed in the subsequent iterations. We propose to compute this number dynamically, with a single user parameter: the probability to miss a minimum primitive, that is a primitive with the minimum number of points (which is our stopping criterion). Let us call p_{miss} this probability. Without any prior information, a minimum primitive with $n_{\min}$ points in a point cloud of n_{pts} points is detected in a RANSAC iteration if the three points are on the primitive, an event of probability $(n_{\min}/n_{\text{pts}})^3$. Such a primitive will be missed in n_{iter} RANSAC iterations with probability $p_{\text{miss}} = (1-(n_{\min}/n_{\text{pts}})^3)^{n_{\text{iter}}}$, so if the user wants to control this probability, the appropriate choice for the number of iterations is:

$$n_{\text{iter}} = \frac{log(p_{\text{miss}})}{log(1-(n_{\min}/n_{\text{pts}})^3)}. \tag{3}$$

As soon as we have found a primitive with more than $n_{\min}$ points, we can update $n_{\min}$ to the number of inliers of our best current candidate as we only aim at finding a better primitive, with more points, which reduces n_{iter} as the iterations run. Moreover, n_{pts} can also be updated each time a primitive is selected. To take into account the surface weighting of Section 2.2, we simply replace $n_{\min}$ with $A_{\min}$ and n_{pts} with the sum of weights over the remaining points.

2.6 Alpha Shape

At this point, our sensor space RANSAC produces compact planar sets of points. Approximating these sets by whole planes is not a good data abstraction as only a local part of the plane is supported by inliers. We want a compact localised representation instead, for which we propose a planar polygon. We do this in these few steps:

1. Estimate the best fitting plane to the inliers in the least squares sense, which is simply the plane passing through the barycenter of the points and whose normal is given by the eigenvector of the inertia matrix $\sum PP^t$ corresponding to the smallest eigenvector.

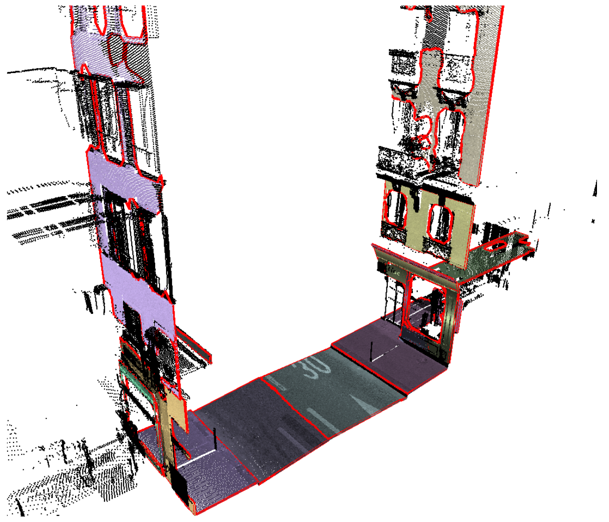

Figure 3. A view of a result of our sensor space enhanced polygonal RANSAC with a reference set of parameters. Point attributed to a planar patch have a different hue for each patch, unattributed points are in black. The extracted planar polygonal outlines are in red.

2. Project each point on this plane.

3. Create a local 2D coordinate frame of this plane.

4. Compute the 2D α-shape of the points in this frame.

5. Extract the exterior border of the α-shape, resulting in a set of 2D polygons lying on the plane.

6. Recover the 3D coordinates of the 2D polygon vertices.

The 2D α-shape is simply a 2D Delaunay Triangulation of the 2D points (Xiaobo et al., 1999), removing all the triangles of circumradius larger than α and edges of length above α, allowing to define a "border" for a point set at a given scale (α). For $\alpha \to \infty$, the result is the convex hull of the points, whereas for $\alpha \to 0$ the result is simply the point cloud itself with no reconnection made as all triangles and edges are removed. The result of this step is thus a planar polygon that is always close to a supporting data point, with α defining this proximity in the supporting plane of the polygon and d_{in} the orthogonal distance. This parameter can be used to define the minimum point density on a detected polygon as only points closer than α will be reconnected to form a polygon.

3. EXPERIMENTS AND EVALUATIONS

3.1 Experiments and Parameter Influence

The improved RANSAC with all enhancements proposed above has been tested on the dataset described in Section 1.2 with a reference set of parameters:

1. $p_{\text{miss}} = 0.1\%$, meaning we ensure to find the best planar patch with 99.9% certainty, and the number of iterations is adapted automatically.

2. $n_{\min} = 2000$, meaning we ask our method to produce only planar patches containing more than 2000 points.

3. d_{in} =3 cm, meaning we generalize our data and lose details smaller than 3 cm.

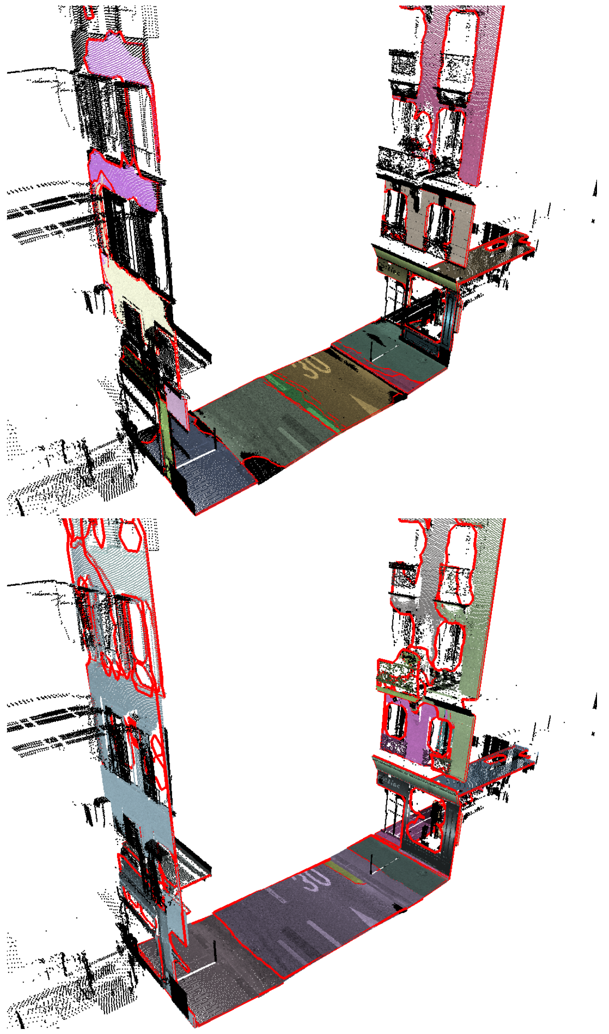

Figure 4. Results of SSP-RANSAC for varying inlier threshold: top d_{in} =1 cm, bottom d_{in} =10 cm.

4. Sample window width=20: samples will be taken in a window of radius 20 pulses in sensor topology.

5. Growing window width=4: samples up to a distance of 4 pulses in sensor topology will be added to the region growing, making it able to overcome small gaps or holes.

6. $\alpha = 5\,\text{cm}$, which means we create a polygonal border for our planar patches encompassing all inliers with this level of detail.

We display a result with this parameterization in Figure 3. We can see that the scene is abstracted by a low number of polygons properly delineating the planar point patches. The road being not perfectly planar but curved across its main direction, it is split in this direction. We now analyse the influence of the main parameters.

3.1.1 Inlier Threshold The level of detail of the resulting patches depends on the inlier threshold. As shown in Figure 4, a higher threshold loses details, mistaking the road and sidewalk surface for instance, while a smaller threshold captures more detail.

3.1.2 Minimum Patch Size The minimum patch size is a detection threshold. A small value capture most planar structures including inside the buildings (Figure 5 top) while a large

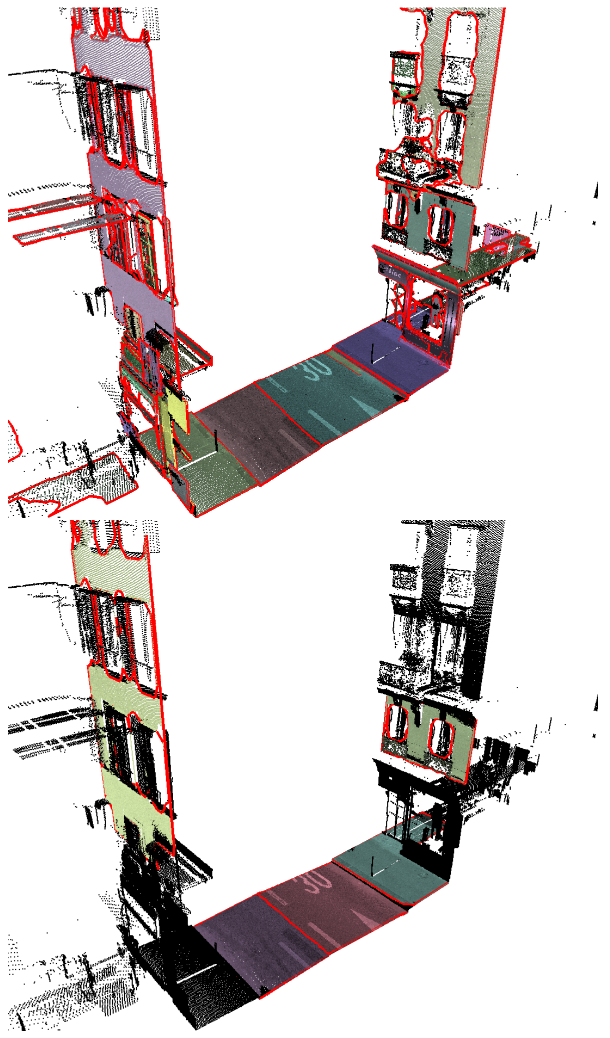
Figure 5. Results of SSP-RANSAC for varying minimum patch size: top $n_{\mathrm{min}} = 500$, bottom $n_{\mathrm{min}} = 10\,000$.

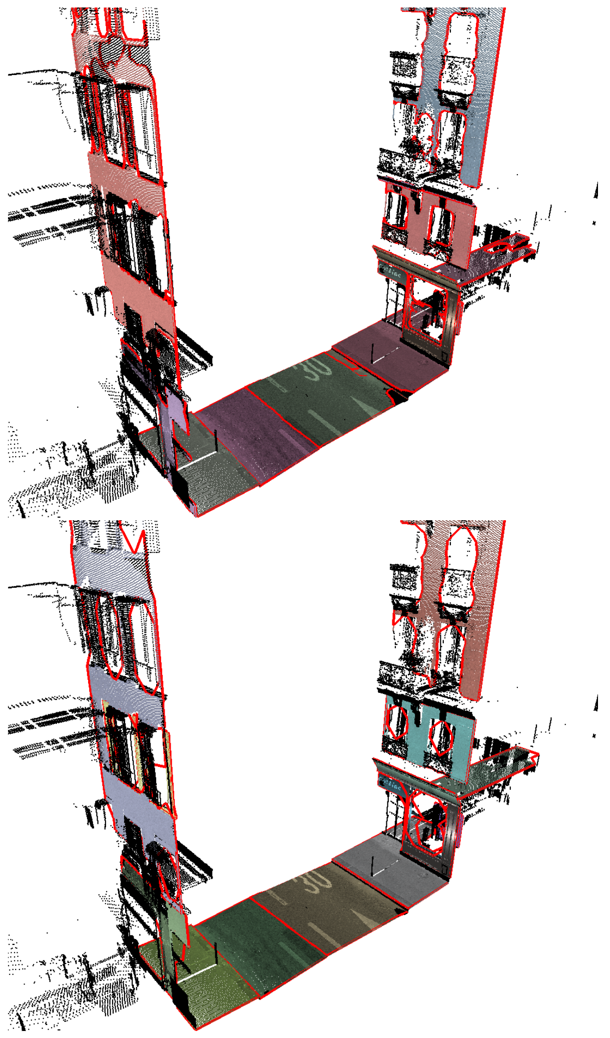
Figure 6. Results of SSP-RANSAC for varying apha shape radius: top α =2 cm, bottom α =20 cm.

value allow to focus on the most prominent structures such as the road, sidewalk and facades (Figure 5 bottom).

3.1.3 Alpha Shape Radius

The alpha shape radius is a generalization parameter for the polygonal borders. A small value makes the polygon border very detailed (Figure 6 top) while a large value yields simpler polygons (Figure 6 bottom).

3.1.4 Region Growing

Region growing is not a parameter but an enhancement that modifies the nature of the algorithm: instead of looking for complete planes, it looks for more localized and compact planar patches. This is exhibited on Figure 7 where region growing was disabled, inducing confusion between non adjacent regions, such as a part of the road adjacent to a sidewalk being in the same patch as the opposite sidewalk.

3.2 Evaluation

We now evaluate the various enhancements to RANSAC proposed in this paper. Evaluation of the compactness will be qualitative as it is not the objective of planar RANSAC itself so comparing quantitatively would make no sense. For the other evaluation metrics, we propose to see RANSAC as an optimization heuristic, aiming at finding a compromise between maximizing the number of inliers while minimizing the number of primitives so our main metrics will be the number of primitives (the lower the better) and number of inliers (the higher the better). Moreover, the computing time will also be given as most enhancements aim at better performance.

3.2.1 Evaluation of Sensor Space Region Growing

In order to evaluate the effectiveness of sensor space sampling, we provide the numbers of inliers and extracted polygons for a fixed number of RANSAC iterations (for easier interpretation) as well as the computing time on an Intel Core i7 CPU @ 3.33 GHz. The point cloud used is a single one-second block of acquisition containing 270 000 points. Because RANSAC is a randomized algorithm, we give the results as minimum and maxium values over 10 runs of the algorithm with the same parameterization: with/without Sensor Space Sampling (SSS), with/without Region Growing (RG) and with the number of RANSAC iterations $n_{\mathrm{it}} = 500$ or 2000. The results are given in Table 1.

First we note that SSS is purely an optimization enhancement, allowing a better sampling thus a better solution achieved for the same number of iterations, or equivalently requiring fewer iterations and less computing time for the same result quality. This is clearly supported by the evaluation that shows always more inliers with SSS than without, but the same computing time for the same number of RANSAC iterations. For SSRG,

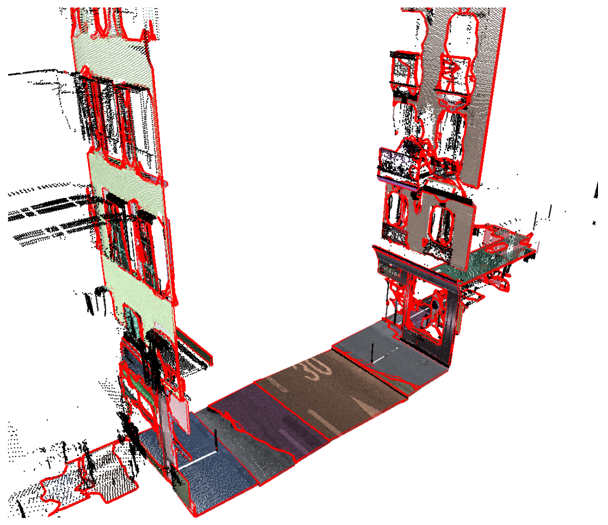
Figure 7. Results of SSP-RANSAC without Sensor Space Region Growing (SSRG) exhibiting more confusion due to long range interactions.

SSS	SSRG	n_{it}	inliers	polygons	time (s)
no	no	500	90–111k	5–8	11
yes	no	500	135–145k	9–11	11
no	yes	500	75–106k	4–7	2.1–2.8
yes	yes	500	115–128k	7–10	2.1–2.8
no	no	2000	109–131k	7–9	42–49
yes	no	2000	145–153k	10–12	42–49
no	yes	2000	87–111k	4–7	7–8
yes	yes	2000	128–137k	9–11	7–8

Table 1. Evaluation of the influence of Sensor Space Sampling (SSS) and Sensor Space Region Growing (SSRG) on the number of inliers and planar polygons found by RANSAC for 500 and 2000 RANSAC iterations (min-max values over 10 runs).

the objective is different as the extracted patches are compact with SSRG whereas this is not enforced without. This additional constraint explains that there are fewer inliers with SSRG than without. In terms of computing time, we see the benefit of region growing that requires much less distance computation, close to a factor 4.

3.2.2 Evaluation of Dynamic Iteration To evaluate the effectiveness of dynamic iteration adjustment, we use the same metrics as in the previous section on the same point cloud, but now compare a set of experiments without dynamic iteration and tuning n_{it} with dynamic iteration and tuning p_{miss}, also on batches of 10 runs and on the same point cloud. The results are provided in Table 2.

This experiment exhibits clearly RANSAC's nature to guarantee better results for more iterations/computing time. For an equivalent computing time, dynamic iteration is always better than static. We also see that dynamic allows this increase in performance on two fronts: having better results for the same number of iterations as they are used more wisely, and having lower computing time for the same number of iterations because more iterations are used later in the process when the cloud has fewer points (because large primitives have already been found) thus iterations are faster but primitives are smaller, hence harder to find. Moreover, parameter tuning is easier with dynamic iteration as it will be automatically adapted to the scene and primitive sizes.

Fixed number of iterations

n_{it}	inliers	polygons	# it	time (s)
100	93–114k	5–7	500–700	2.9–3.3
200	109–126k	7–10	1400–2000	6.2–7.1
500	125–130k	9–10	4500–5000	16–17
1000	127–136k	9–10	9000–10000	29–33

Dynamic number of iterations

p_{miss}	inliers	polygons	# it	time (s)
0.3	103–124k	7–10	647–978	1.7–2.5
10^{-1}	118–130k	8–11	1.4–2.2k	3–4.1
10^{-2}	118–132k	8–11	2.7–4.1k	5.9–7.8
10^{-3}	118–136k	8–11	4.2–6.9k	8.4–11
10^{-4}	120–137k	8–11	5.5–7.7k	11–14
10^{-5}	130–137k	10–11	8.8–9.7k	15–17

Table 2. Evaluation of the interest of dynamic iteration on the number of inliers and planar polygons found by RANSAC for a varying parameterization (min-max values over 10 runs).

4. CONCLUSIONS AND FUTURE WORK

This paper presented a method for planar polygon extraction from a point cloud for which the sensor topology is available. This condition is not very restrictive as most Lidars have this characteristic: MLS as demonstrated in this paper, Aerial Lidar Scanning (ALS) has the exact same topology, and TLS often have an array topology (in spherical coordinates) that is often available. The problem is more of a procedural nature, that is not losing this information when exporting the point cloud from the raw data.

We demonstrated clearly the advantage of exploiting sensor topology, even if some of the proposed enhancements could have been possible using a k Nearest Neighbors (kNN) topology or even a Delaunay Triangulation (DT). Future works could consist in comparing sensor topology with these other choices. However, sensor topology will always have the advantage of being free, requiring no computing time as long as it was stored, while kNN and DT have an important computing cost on large point clouds.

A comparison between our enhanced RANSAC and Hough voting based or graph based methods would be interesting, in particular the latter that also aims at patch compacity.

Finally, we aim at exploiting this very efficient planar polygon extraction method for three main applications:

- Point cloud registration (Takai et al., 2013), as these polygons should be easy to match between unregistered point clouds as they accurately abstract a large number of points with few primitives. Moreover, they average the sensor noise over a large number of points so they might even allow to achieve a registration accuracy better than the noise level. Our method would obviously be applicable to point cloud registration on 3D city models as proposed in (Monnier et al., 2013) as planar polygons are the only structures that can be put in correspondence in this case.
- Polyhedral/3D reconstruction (Boulch et al., 2014), (Nan and Wonka, 2017): many man made structures are polyhedral, such as roofs and urban furniture, and most are hybrid, mixing planar areas with more freeform areas, which could be represented by a hybrid mesh mixing triangle mesh and large planar polygons. In both cases, our planar polygon extraction method should prove a very useful preprocessing.

- Façade, sidewalk and road surface extraction (Nurunnabi et al., 2013), (Hervieu and Soheilian, 2013) as they are caracterised as large horizontal or vertical planar patches, but also the detection of road curb (El-Halawany et al., 2011), (Zhao and Yuan, 2012) and building footprints as the limits of sidewalk polygons.

ACKNOWLEDGMENTS

This research has been funded by ANR (Agence Nationale de la Recherche-France) Project BIOM (ANR-17-CE23-0003).

REFERENCES

Bentley, J. L., 1975. Multidimensional binary search trees used for associative searching. Communications of the ACM 18(9), pp. 509–517.

Borrmann, D., Elseberg, J., Lingemann, K. and Nüchter, A., 2011. The 3D Hough transform for plane detection in point clouds: A review and a new accumulator design. 3D Research 2(2), pp. 3.

Boulch, A., de La Gorce, M. and Marlet, R., 2014. Piecewise-planar 3D reconstruction with edge and corner regularization. In: Computer Graphics Forum, Vol. 33/5, Wiley Online Library, pp. 55–64.

Chen, J. and Chen, B., 2008. Architectural modeling from sparsely scanned range data. International Journal of Computer Vision 78(2-3), pp. 223–236.

Cohen-Steiner, D., Alliez, P. and Desbrun, M., 2004. Variational shape approximation. ACM Trans. Graph. 23(3), pp. 905–914.

El-Halawany, S., Moussa, A., Lichti, D. D. and El-Sheimy, N., 2011. Detection of road curb from mobile terrestrial laser scanner point cloud. In: International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences, Vol. 2931, pp. 29–31.

Fang, H., Lafarge, F. and Desbrun, M., 2018. Planar shape detection at structural scales. In: Proc. of the IEEE conference on Computer Vision and Pattern Recognition (CVPR).

Guinard, S. and Vallet, B., 2018. Weighted simplicial complex reconstruction from mobile laser scanning using sensor topology. Conférence Française de Photogrammétrie et de Télédétection (CFPT), Champs-sur-Marne, France.

Guinard, S., Landrieu, L., Caraffa, L. and Vallet, B., 2019. Piecewise planar approximation of large 3d data as graph-structured optimization. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W5, pp. 365–372.

Hervieu, A. and Soheilian, B., 2013. Semi-automatic road/pavement modeling using mobile laser scanning. In: ISPRS Annals of Photogrammetry, Remote Sensing and the Spatial Information Sciences, Vol. II-3/W3, pp. 31–36.

Isack, H. and Boykov, Y., 2012. Energy-based geometric multi-model fitting. International Journal of Computer Vision 97(2), pp. 123–147.

Knopp, J., Prasad, M., Willems, G., Timofte, R. and Van Gool, L., 2010. Hough transform and 3D SURF for robust three dimensional classification. In: European Conference on Computer Vision, Springer, pp. 589–602.

Landrieu, L. and Obozinski, G., 2017. Cut pursuit: Fast algorithms to learn piecewise constant functions on general weighted graphs. SIAM J. on Imaging Sciences 10(4), pp. 1724–1766.

Monnier, F., Vallet, B., Paparoditis, N., Papelard, J.-P. and David, N., 2013. Registration of terrestrial mobile laser data on 2d and 3d geographic database by use of a non-rigide icp approach. In: ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. II-5/W2, pp. 193–198.

Monszpart, A., Mellado, N., Brostow, G. J. and Mitra, N. J., 2015. Rapter: rebuilding man-made scenes with regular arrangements of planes. ACM Trans. Graph. 34(4), pp. 103–1.

Nan, L. and Wonka, P., 2017. Polyfit: Polygonal surface reconstruction from point clouds. In: The IEEE International Conference on Computer Vision (ICCV).

Nurunnabi, A., West, G. and Belton, D., 2013. Robust locally weighted regression for ground surface extraction in mobile laser scanning 3d data. In: ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. 1.2, pp. 217–222.

Paparoditis, N., Papelard, J.-P., Cannelle, B., Devaux, A., Soheilian, B., David, N. and Houzay, E., October 2012. Stereopolis II: A multi-purpose and multi-sensor 3D mobile mapping system for street visualisation and 3D metrology. In: Revue Francaise de Photogrammétrie et de Télédétection 200: 69-79.

Pu, S., Rutzinger, M., Vosselman, G. and Elberink, S. O., 2011. Recognizing basic structures from mobile laser scanning data for road inventory studies. ISPRS Journal of Photogrammetry and Remote Sensing 66(6, Supplement), pp. S28 – S39. Advances in LIDAR Data Processing and Applications.

Rabbani, T., Van Den Heuvel, F. and Vosselmann, G., 2006. Segmentation of point clouds using smoothness constraint. International archives of photogrammetry, remote sensing and spatial information sciences 36(5), pp. 248–253.

Schnabel, R., Wahl, R. and Klein, R., 2007. Efficient RANSAC for point-cloud shape detection. In: Computer graphics forum, Vol. 26/2, Wiley Online Library, pp. 214–226.

Takai, S., Date, H., Kanai, S., Niina, Y., Oda, K. and Ikeda, T., 2013. Accurate registration of mms point clouds of urban areas using trajectory. In: ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. 1.2, pp. 277–282.

Xiaobo, W., Shixing, W. and Chunsheng, X., 1999. A new study of Delaunay triangulation creation. ACTA Geodaetica et Cartographic Sinica.

Zhao, G. and Yuan, J., 2012. Curb detection and tracking using 3d-lidar scanner. In: Image Processing (ICIP), 2012 19th IEEE International Conference on, IEEE, pp. 437–440.