Reconstruction of interior walls from point cloud data with min-hashed J-linkage

Luca Magri, Andrea Fusiello
DPIA - University of Udine
Via delle Scienze, 208, - 33100 Udine, IT
surname.name.1@gmail.com, name.surname@uniud.it

Abstract

The automatic reconstruction of the walls of an interior environment is a fundamental task in any "scan2BIM" application. In this work, we address this problem resorting to an original and improved version of J-Linkage that leverages on the min-Hash technique to boost the efficiency without sacrificing the accuracy. A framework to automatically and robustly extract floor plans from large-scale point clouds is described and validated on real-word publicly available data.

1. Introduction

Understanding the structure of indoor environments is an important task that, in recent years, has attracted the interest of researchers yielding to several applications that span many fields such augmented and virtual reality, navigation, design, maintenance, monitoring of buildings and energy simulation analysis.

Specifically, the increasing availability of high quality and massive 3D data, the diffusion of BIM and the sustained need of accurate as-built architectural models has driven the attention towards automatic methods capable of dealing with large-scale point clouds and, at the same time, able to retrieve detailed geometry, avoiding time-consuming and costly human intervention as much as possible.

The demand of higher-level, automated and accessible rendition of indoor models, can be regarded as an occurrence of the broader aim of bridging the semantic gap that separates automatic perception from human comprehension which ultimately inspires many efforts in Computer Vision. A first preliminary step along this direction is taken by geometric multi-structure recovery (also known as multi-model fitting), which aims at aggregating input data, usually corrupted by noise and outliers, into multiple instances of geometric parametric models.

In the context of indoor modeling, this translates mainly in the robust extraction of multiple geometric primitives consisting of the primary facility surfaces – such as floors, walls, and ceilings – from the point cloud. Usually planar structures are detected exploiting 3D plane fitting techniques *e.g.* [20]. Alternatively, the description of the environment is conveniently addressed from a 2D perspective and the architectural components are described as a mixture of multiple line segments, as in a blueprint e.g. [18]. Eventually, under the assumption of vertical surfaces, a 3D model can be easily extruded from the blueprint.

The typical challenges of multi-model fitting are lurking here. The structure estimation process must tolerate pseudo-outliers and rogue measurements, due to the presence of furniture causing clutter and complex building layouts which determine occlusions. Moreover, the fit of geometric models and the assignment of points to the estimated structures are intrinsically entangled tasks and give rise to a sort of chicken-and-egg-dilemma. In addition, the number of sought structures is unknown in advance and must be estimated together with the parametric models, entailing implicitly or explicitly some model selection criterion.

Multiple structure recovery is a common essential step involved in the majority of the works about indoor modeling, including the most elaborate methods, which make use of it as a preparatory phase to further semantic or topological analysis. Some long-established algorithms are customarily adopted: [18, 1, 25] rely on *Hough transform* [9] to detect collinear configurations of 2D wall samples. Unfortunately this technique is not very robust to clutter and outliers, especially as the number of structures grows and the distribution of inliers per structure is uneven. Along the same line, [14] resorts to *mean-shift* [7], whose parametrization can be critical and which is not intrinsically robust. Many work, e.g. [19, 17, 16], make use of *sequential-Ransac* to fit multiple planes in space. Despite the robustness of Ransac in single model estimation, it has been recognized that the local and incremental nature of this sequential strategy is suboptimal when dealing with multiple structures and easily prone to miss global scene-level features. To conclude, we would like to mention [24], where the problem of outdoor modeling is addressed by resorting to *Pearl* [11], a more recent energy-based multi-model fit-

ting method, which is used to detect 2D lines. Its effectiveness, however, is connected to the guess of a correct trade-off between the terms of the energy optimized, which in some cases can be a thorny model selection problem.

In this paper we propose to tailor J-Linkage, a multi-model fitting technique that has proven successful in overcoming some of the mentioned limitations of Ransac and Hough transform [22], to the indoor modeling scenario. With respect to the standard algorithm, we propose here a variation based on min-Hash that improves its efficiency thereby allowing to deal with large ($> 10^8$) point clouds.

2. J-Linkage and min-Hash

In this section, we describe how min-Hash can be employed to improve the efficiency of J-Linkage without affecting its accuracy. For the convenience of the reader we briefly recall the original J-Linkage algorithm in Section 2.1, while Section 2.2 describes our approach.

2.1. J-Linkage algorithm

At high-level, J-Linkage implements a two steps *first-represent-then-clusterize* scheme: at first, data are represented by the preferences they grant to a pool of model hypotheses, then a greedy agglomerative clustering is performed to obtain a partition of the data.

Let X be the set of data point, $\mathrm{err}\colon X \times H \to \mathbb{R}$ an error function to measure residuals between data and models, and ϵ an inlier threshold provided as input. The method starts by generating a set of random provisional models $H = \{h_1, \ldots, h_m\}$ by drawing m subsets of data points with the minimum cardinality necessary to instantiate a model. Then a $n \times m$ binary matrix P is built by defining its (i, j)-th entry as

$$P(i,j) = \begin{cases} 1 & \text{if } \mathrm{err}(x_i, h_j) \leq \epsilon \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Each row P_i can be easily identified with the *preference set* $\mathrm{PS}(x_i)$ of a given point x_i, *i.e.* the subset of structures in H that support that point. The main intuition is that points belonging to the same model will have similar preference sets, and therefore can be clustered in the conceptual space $\{0, 1\}^m$ to reveal the structures hidden in the data.

The preference set of a subset $Y \subseteq X$ is composed by the models that fit all the points in Y:

$$\mathrm{PS}(Y) = \bigcap_{x \in Y} \mathrm{PS}(x). \tag{2}$$

The clustering algorithm proceeds in a bottom-up manner. At first every data is put in its own cluster. The distance between clusters is computed as the Jaccard distance [12] between the respective preference sets. The Jaccard distance between two sets U, V is defined as $1 - J(U, V)$, where

$$J(U,V) = \frac{|U \cap V|}{|U \cup V|}, \tag{3}$$

denotes the Jaccard similarity.

Starting from singletons, each sweep of the algorithm merges the two clusters with the smallest Jaccard distance, until all the preference sets of clusters are disjoint. The parameters of the returned structures are estimated by least squares fitting on each cluster of points. It is worth noting that, if outliers are not present in the data, the number of clusters is automatically detected by this algorithm. Moreover this preference approach is robust to outliers, that can be recognized as observations whose preferences deviate significantly from the rest of the data, and tend to emerge as micro-clusters, that can be pruned out a posteriori.

2.2. Jaccard distances with min-Hash

In this section, we describe how to estimate efficiently the Jaccard distances. The cardinal idea, borrowed from the context of duplicate retrieval in large database [4, 6], is to take advantage of min-Hash, a local sensitive hashing procedure, to approximate the Jaccard similarity between two preference sets.

The min-Hash of a (preference) set $U \subseteq H$ is a natural number such that the probability that a second set $V \subseteq H$ has the same min-Hash value is equal to the Jaccard coefficient $J(U, V)$. Specifically, the min-Hash of U is defined as

$$\mu(U, f_k) = \arg\min_{h \in U} f_k(h) \tag{4}$$

where $f_k\colon H \to \mathbb{N}$ is a bijective random function with the property that $\mathrm{Prob}\left(f_k(h_i) < f_k(h_j)\right) = 0.5$. In other words, $\mu(U, f_k)$ is the minimum element of U under the ordering naturally induced by f_k[1]. It is easy to verify that

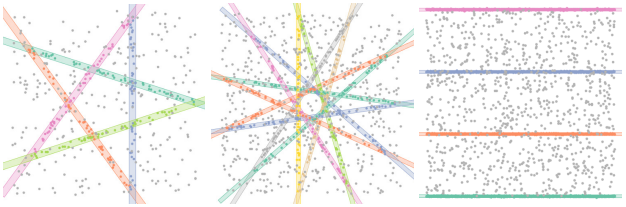
$$\mathrm{Prob}\left(\mu(U, f_k) = \mu(V, f_k)\right) = J(U, V). \tag{5}$$

As a matter of fact, let $h' = \mu(U \cup V)$. Since, f_k is a random hash function, every element of $U \cup V$ has the same probability of being the minimum element, thus h' can be thought as being randomly extracted from $U \cup V$. If $h' \in U \cap V$ then $\mu(U, f_k) = \mu(V, f_k)$, otherwise $\mu(U, f_k) \neq \mu(V, f_k)$ and equation (5) holds.

In principle, f_k can be realized by a random permutation σ defined on $\{1, \ldots, m\}$ setting $f_k(h_i) = h_{\sigma(i)}$, but, in practice, it can be implemented as a linear transformation on a convenient finite field [5].

A number κ of independent hash functions $\{f_k\}_{k=1}^{\kappa}$ is considered, and for each preference set the corresponding κ min-Hash values are computed and collected into a vector termed *min-Hash signature*. The Jaccard similarity between two sets is estimated as the number α of common

[1] An order on H is defined by $h_i < h_j \iff f_k(h_i) < f_k(h_j)$



(a) *Star5* (b) *Star11* (c) *Line4*

Figure 1: Datasets used for synthetic experiments

min-Hash values between their signatures, over the number κ of function adopted, because α/κ turns to be an unbiased estimator of $J(U, V)$. In conclusion, the Jaccard distance between two preference sets can be approximated as easily as computing the Hamming distance of the respective min-Hash signatures. This is an appealing feature as it opens the possibility of exploiting specialized implementations to accelerate the computation of the Hamming distance.

The clustering procedure can be summarized as in Algorithm 1.

Algorithm 1 min-hashed J-linkage clustering

1. Put each datum in its own cluster;
2. Define the preference sets of clusters using (2);
3. Compute min-hash signature for the preference sets;
4. Among all current clusters, pick the two clusters with the small hamming distance between their min-Hash signatures;
5. Replace these two clusters with the union of the original ones;
6. Update min-Hash signatures and hamming distances;
7. Repeat from step (4) while the smallest hamming distance is lower than 1.

As one could expect, the theoretical upper bound of the min-Hash estimation is a function of the number κ of functions employed: a higher value of κ increases the accuracy of the estimation, but at the cost of larger signature vectors and longer computation times. In particular, the Chernoff bound implies that the number κ of min-Hash values needed to achieve an estimated error upper bounded by $\theta \in (0, 1]$ with confidence $1 - \delta$, $\delta \in (0, 1]$, must satisfy the constraint [21]

$$\kappa \geq \frac{2 + \theta}{\theta^2} \log\left(\frac{2}{\delta}\right). \tag{6}$$

2.3. Validation

We assess the benefits of exploiting min-Hash to approximate Jaccard distances on both synthetic and real data. The experiments are run on an 2,6 GHz Intel Core i7 machine with 16 GB RAM. Matlab code is available upon request from the authors.

Synthetic experiments This set of simulations is designed to investigate in practice the benefits brought by min-Hash to J-Linkage with respect to the number κ of hash functions. We consider the problem of fitting multiple lines to the three datasets reported in Figure 1: *Star5* and *Star11*, taken from [22], are respectively composed by 5 and 11 line-structures of 50 inliers points each. These two datasets are both contaminated by 50% of outliers. The third dataset *Line4* has been introduced to evaluate the outcomes on a larger-size problem, it is composed by 4000 points equally divided among 4 lines and 1000 gross outliers.

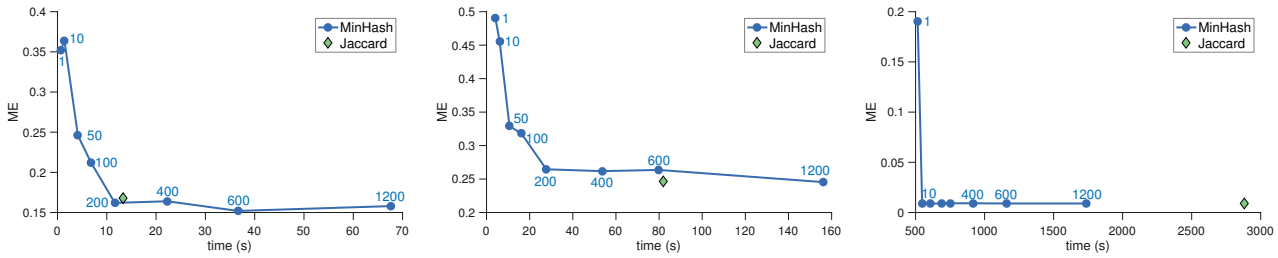
The figures of merit we take into account are the computational time and the misclassification errors (ME), which counts the percentage of misclassified points with respect to the ground-truth label. Please note that, even if the ME is restricted to deal with single-label assignments, some intersecting structures are present in the first two datasets: 8.20% of points in *Star5* and 17.09% in *Star11* could have legitimately been assigned to a different ground-truth label. Due to this fact, the ME could show fluctuations that are not significant and this should be kept in mind when results are evaluated.

We compare, on the basis of the same preference matrix P, the original version of J-Linkage to our min-Hash variant implemented with different numbers of hash functions, namely $\kappa \in \{1, 10, 50, 100, 200, 400, 600, 1200\}$ –1200 being the total number of sampled model hypotheses in P. Outliers are pruned out, using the simple strategy based on cluster cardinality presented in [13].

Results are summarized in Table 1 and displayed in Figure 2. As regards the first two datasets, 200 functions are enough for min-Hash to match the performances of J-Linkage in terms of ME, within the margins of the aforementioned ME fluctuations. The time is almost the same for *Star5*, but on *Star11* the min-Hash version is 3 times faster. In *Line4* the structures are well separated and do not intersect, therefore the ME does not fluctuate and, when $\kappa \geq 10$, the min-Hash version matches exactly the accuracy of J-Linkage, with the added merit that, for $\kappa = 10$ the solution is more than $4\times$ faster.

These examples show that, as the number of processed points grows, the advantages of min-Hash become more tangible, making it very suitable for dealing with large-scale data.

Vanishing point detection In this experiment we test the effects of using min-Hash on a real dataset. We take into account the York Urban Line Segment Database [8], a collection of 102 images of urban environments, comprising



(a) *Star5* (b) *Star11* (c) *Line4*

Figure 2: Using min-Hash to approximate the Jaccard distances: the blue points depict the ME and time achieved by min-hash as the number κ of hash functions varies. The green diamond represents the classical J-linkage with Jaccard distance.

	Jaccard		min-Hash		
	ME [%]	time [s]	ME [%]	time [s]	κ
Star5	16.80	13.33	16.2	11.65	200
Star11	24.64	81.96	26.45	27.73	200
Line4	0.92	2281	0.92	547	10

Table 1: Results on line fitting experiments

		Jaccard	min-Hash
ME [%]	mean	2.83	2.83
	median	1.58	1.58
time [s]	mean	0.90	0.15

Table 2: Results on York Urban DB

Figure 3: Sample images of the York Urban DB

annotated line-segments with ground-truth labels – sample images can be seen in Figure 3. The aim is to retrieve two or three points on the projective plane (possibly lying on the line at infinity) which identify mutually orthogonal vanishing directions. If a segment is thought to "belong" to a vanishing point when its supporting line, embedded in the projective plane, is incident to it, this problem plainly becomes an instance of multiple structures recovery. Our choice falls on this dataset because later on, we will rely on vanishing point detection to exploit the Manhattan Word assumption for indoor modeling. Table 2 summarizes the results: the same ME scores realized by J-Linkage can be attained leveraging on min-Hash with considerable time savings.

To sum up, the experiments conducted demonstrate that the approximation of Jaccard distances introduced by min-Hash does not affect the strength of J-Linkage, in particular its robustness, as outliers continue to emerge as small clusters that can be easily recognized and discarded. In practice, we observe that the theoretical bound of Equation (6) is not strict, and relatively few hash functions succeed in coping with real datasets. Moreover the advantages in terms of efficiency are substantial, especially for large-size datasets.

3. Indoor modeling

We propose a simple procedure to automatically generate a manageable 2D architectural model from a 3D point cloud. The main idea is to operate on a 2D plane, where points have been projected, and to extract the overall structure of the environment by fitting line segments to the main building features leaning on J-Linkage equipped with min-Hash.

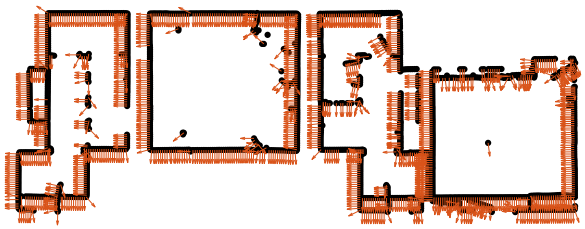
In particular, two major steps can be distinguished. First, after estimating ceiling and floor, the remaining points are projected onto a discretized plane. Second, our min-hashed J-Linkage comes into play to efficiently extract lines and to identify their dominant orientations.These steps are described in detail in the next sections.

3.1. Preprocessing

The first task is to reduce the 3D point cloud to a set of sampled planar points – referred to as "*wall samples*" – enriched with information about their local orientations. For this preliminary step, common to several methods, we follow an approach similar to [23].

Assuming that the coordinate system is defined to have

(a) input point cloud



(b) 2D projected point cloud with normals

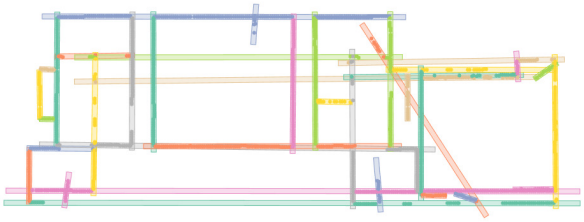
Figure 4: The preprocessing step reduces the input 3D point cloud (top) to a 2D set of points (with normals) (bottom).

the vertical component representing elevation, the inspection of 3D measurements along the z-axis would reveal floor and ceiling as dominant modes on the heights distribution. For this reason, an histogram of heights is computed and the potential floor/ceiling bins are identified as the bottom-most and top-most local maxima. 3D planes are fitted via IRLS on the points belonging to these bins, the corresponding inliers are labeled as floor and ceiling respectively and are then discarded from further analysis.

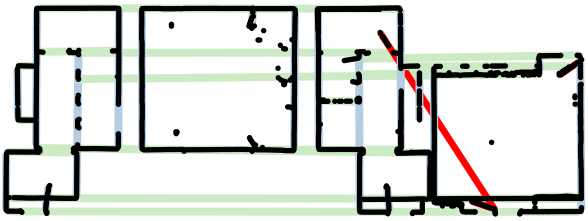
The rest of the 3D points are projected perpendicularly onto the x-y plane which is uniformly discretized in a grid of ground-cells. The width of the cells should be smaller than the expected resolution of the architectural features we desire to describe in the final model. If enough points [2] fall inside a ground-cell their median position is taken as a *wall sample* representative for that cell. At last, for each wall sample a normal vector is locally estimated using Principal Component Analysis together with a confidence measure of the co-planarity of the corresponding points. Only normals with a confidence higher than a certain threshold are retained.

For the sake of illustration, we demonstrate the main steps of this process on a small working example extracted from [2]. The input point cloud is shown in Figure 4a, whereas Figure 4b displays the outcome of the preprocessing stage.

[2] in our implementation more than 3, but a different threshold cardinality can be used depending on the application



(a) line recovery outcome



(b) results of vanishing point detection

Figure 5: Min-hashed J-Linkage is employed to extract multiple lines (top) and to determine their dominant orientations (bottom). Lines that do not conform to the principal orientations are marked in red.

3.2. Line extraction

The next step consists in organizing the wall samples into linear structures, as depicted in Figure 5a. Min-hashed J-Linkage can be employed to this end in a straightforward way, except for an additional expedient which enables us to exploit the normal vectors of wall samples.

Indeed, by taking into account the orientations of points, in addition to their positions, it is possible to distinguish more accurately points that lie on the same wall and thus to increase the robustness of the method against furniture and clutter. For this reason, we put a tentative line into the preference set of a planar point, if their residuals are below the inlier threshold and if, at the same time, the line is perpendicular to the normal of that point within a predefined tolerance.

Several heuristics can be adopted to further improve the detected set of lines. As a preliminary pruning, we deemed as outliers those lines that are supported by few wall samples, and the ones that do not conform to the so called Manhattan Word assumption, which states that man-made environments exhibit a high degree of organization along mutually orthogonal and parallel directions.

With this perspective as a guide, the set of lines are grouped by fitting vanishing points as explained in Section 2.3. Then, the lines of the two clusters with the larger cardinalities are retained, whereas all the others, which do not comply with the two dominant orientations, are pruned out.

For example in Figure 5b the two lines corresponding to the doors of the rightmost room, are recognized as outliers because they are not well aligned with the orthogonal frame defined by the main walls.

3.3. Topological refinement

The collection of detected lines defines a partition of the plane in multiple regions. Some of them are separated by walls, while others are intuitively perceived as the same room.

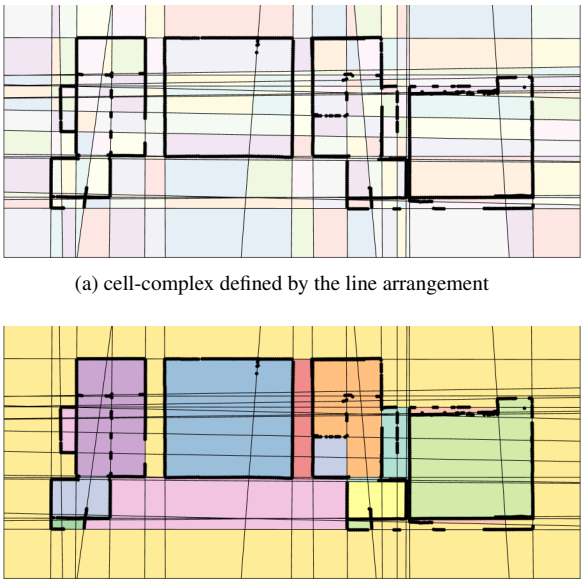
Our aim is to make use of the topological relations between these spaces in order to enhance the architectural models by rejecting spurious wall segments. To this end, we derive a simple but effective room segmentation method that could also serve as an intermediate result intended to facilitate subsequent semantic analysis and to simplify the process of converting point cloud data into building information models.

From the estimated line arrangement, the induced subdivision of the plane is computed and encapsulated in a 2D cell-complex stored in a doubly connected edge list (dcel), a convenient structure commonly used in computational geometry that consists of vertices, edges, and faces (see Figure 6a). Lines are split into segments to represent section of connected walls: at first, clustered wall samples are projected to their supporting lines and linearly ordered, then a segment is detected every time the distance between two consecutive points exceeds a threshold, fixed to 3ϵ in our implementation.

Building on the approach of [15], space segmentation can be formulated as a clustering problem on the faces of the cell-complex. The rationale is to group together subsets of planar faces that are adjacent and, at the same time, "see" a consistent extent of common walls. For this reason, we propose a dissimilarity measure that combines the topology of the line arrangements with a notion of visibility. Similarly, [10] introduces a visibility vector for pixels of an occupancy image of the environment, to segment rooms with k-mediods.

Here, we define the *visibility set* of a face as the set of wall segments that are visible from the cell centroid. Visibility sets can be computed, for example, using techniques based on line sweep such as [3]. However, for our purpose, it is sufficient to employ a simpler method that makes use of the Manhattan Word property enforced through the vanishing point detection stage.

Assuming a Cartesian coordinate frame aligned with the two dominant directions, at first each segment is projected onto the axis which corresponds to its orientation. Then every centroid is projected onto each axis and, if it lies on a projected segment, that segment is marked as visible. In practice, this procedure boils down to effortless comparison between point coordinates.



(a) cell-complex defined by the line arrangement

(b) segmented spaces

Figure 6: The estimated lines divide the space into cells (top) which are then aggregated into rooms by clustering (bottom). The segmentation is color-coded.

The dissimilarity is defined as follows. If two faces are not adjacent or if they are separated by a segment which occupies more than the 50% of the length of their common edge, their dissimilarity is set to 1. Alternatively, the dissimilarity is computed as the min-hashed Jaccard distance of their visibility sets. Dissimilarities are hence aggregated through single-linkage clustering which group cells into labeled regions.

As Figure 6b demonstrates, this technique produces reasonable segmentations and it is able to single out not only rooms in the strict sense, *i.e.* spaces completely enclosed by walls, but also to locate less structured regions, such as vestibules, aisles, halls and corridors.

At the end, the acquired topological information is exploited to refine the 2D models: segments that separate cells belonging to distinct clustered regions are retained as dominant walls. Walls situated in the interior of a room are lifted to 3D as planar patches and, according to the cardinality of their supporting inliers, are either kept or discarded. As a reference, a sample of extruded model is visualized in Figure 7.

The room configuration can also be fruitfully combined with additional 3D analysis, for example opening detection can take advantage of the environment topology to locate doors as those apertures, within a predefined length, that separate regions with different labels.

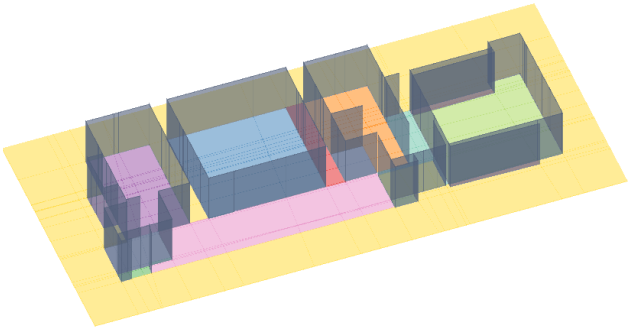


Figure 7: Sample result with extruded walls

	# points	area $[m^2]$	# rooms
Area1	44 026 810	965	45
Area4*	37 523 035	870	46
Area6	34 903 596	935	53

Table 3: Point clouds figures.

3.4. Qualitative results

We explore the outcomes of the proposed method on large-scale indoor environments taken from the Stanford Large-Scale 3D Indoor Spaces Dataset [2]. The input point clouds, captured by a Matterport 3D camera, consist in facilities used for educational and office work and include several furnished rooms and hallways. We consider the 3 point clouds whose outlines satisfy the Manhattan assumption, namely *Area1*, *Area4* and *Area6*. Three rooms of the last dataset develop on two floors, we analyze only the main ground-floor, dubbed *Area6**. However, please notice that the floor-plan generation can be easily extended to deal with multiple stores, as explained in [23]. Statistics on the scanned inputs are collected in 3.

Qualitative results are collected in Figure 8 where the input point clouds, whose ceilings have been removed to disclose the internal organization of the facilities, are juxtaposed to the dominant walls, extruded from the segmented cell-complexes.

We can appreciate that the projected points have been aggregated in meaningful segments and the overall configuration of the reconstructed structures follows closely the layout of the buildings.

Having said that, the 2D analysis alone is far from perfect, and some defects can be spotted. For example, short wall segments, originated form clutter, remain in the final model. Also, in Figure8b, the method misses to reconstruct the exterior wall corresponding to the leftmost room of *Area1*, which consists in a bathroom divided in several closets. The reason can be ascribed to the presence of several minute structures, such as doors, that snatch inliers from the main wall in the 2D domain, undermining the topological scrutiny.

A more elaborated 3D analysis should easily remove these artifacts.

4. Conclusion and future work

We have proposed and examined a faster min-Hash-based version of J-Linkage that achieves a considerable speed-up in terms of computational times without neglecting accuracy. The benefits of this improvement, that paves the way towards the adoption of J-Linkage in large-scale problems, involve a broad range of geometric multi-model fitting tasks, including indoor modeling.

In this scenario, we conceived a simple and effective 2D modeling framework resorting to the main strengths of J-Linkage: first, the proposed method is robust to clutter and outliers. Second, the number of sought architectural structures is automatically estimated, and the main input parameter, the inlier threshold, turns to be an educated guess because it is naturally related to the desired model resolution. Third, the min-Hash approach allows to cope efficiently with large-scale point clouds.

This method could serve as a reliable stepping stone to further improve the architectural model. A more elaborate analysis of the environment that stems from wall detection and incorporate additional 3D information is in plan for future work. Yet, we hope that our approach could already ease the burden of user post-processing.

The generalization of the line-arrangement clustering to 3D planes and spaces will be subject of further investigation.

Acknowledgement

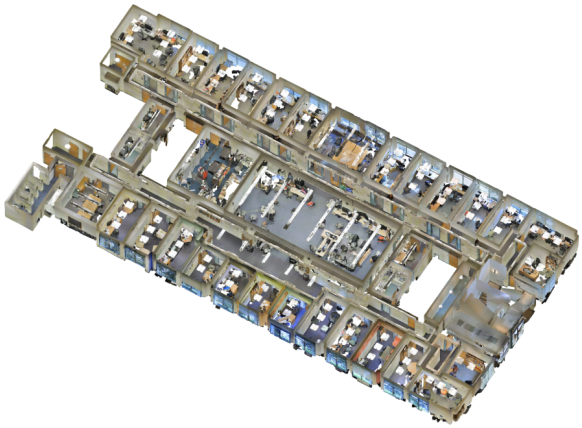
Luca Magri gratefully acknowledges the support of 3Dflow srl. This research was funded by the "HEaD Higher Education and Development Project" FP1619942003.

References

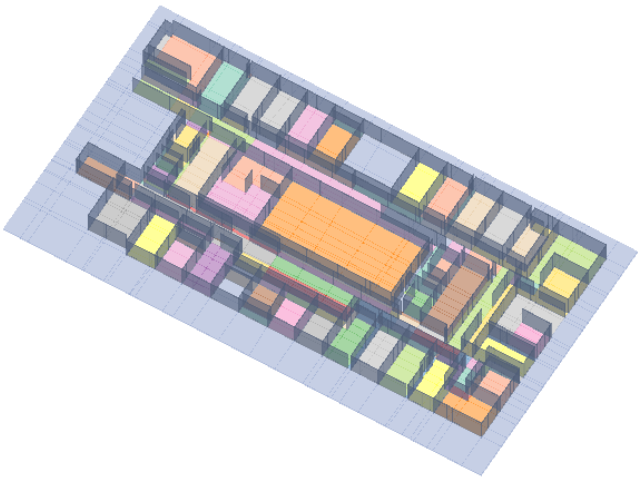
[1] A. Adan and D. Huber. 3d reconstruction of interior wall surfaces under occlusion and clutter. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on*, pages 275–281. IEEE, 2011. 1

[2] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese. 3d semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1534–1543, 2016. 5, 7

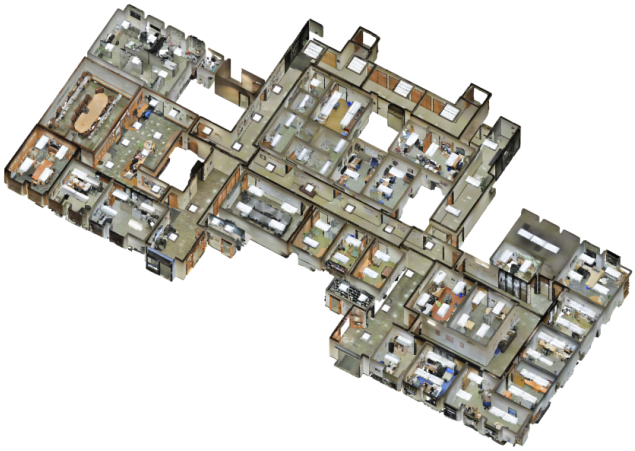
[3] T. Asano. An efficient algorithm for finding the visibility polygon for a polygonal region with holes. *IEICE TRANSACTIONS (1976-1990)*, 68(9):557–559, 1985. 6



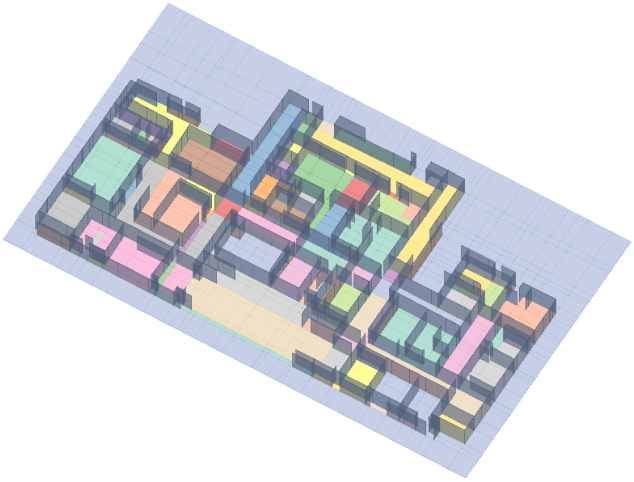
(a) *Area1* input point cloud



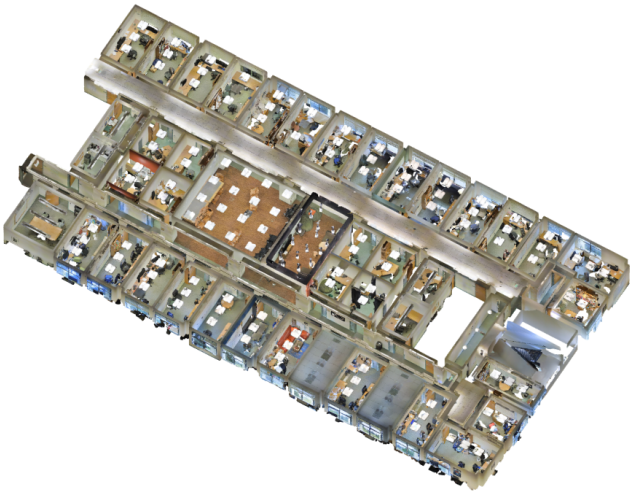
(b) reconstructed walls



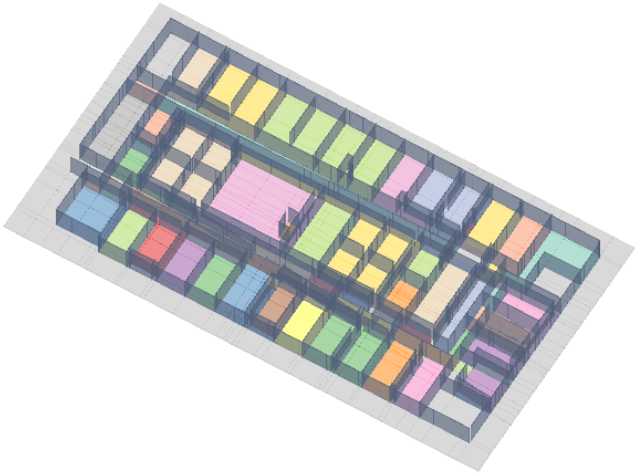
(c) *Area4* input point cloud



(d) reconstructed walls



(e) *Area6* input point cloud



(f) reconstructed walls

Figure 8: Results on the Manhattan scenes of the Stanford Large-Scale 3D Indoor Spaces Dataset

[4] A. Z. Broder. On the resemblance and containment of documents. In *Compression and Complexity of Sequences 1997. Proceedings*, pages 21–29. IEEE, 1997. 2

[5] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher. Min-wise independent permutations. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 327–336. ACM, 1998. 2

[6] O. Chum, J. Philbin, A. Zisserman, et al. Near duplicate image detection: min-hash and tf-idf weighting. In *BMVC*, volume 810, pages 812–815, 2008. 2

[7] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002. 1

[8] P. Denis, J. H. Elder, and F. J. Estrada. Efficient edge-based methods for estimating manhattan frames in urban imagery. In *Proceedings of the European Conference on Computer Vision*, pages 197–210, 2008. 3

[9] R. O. Duda and P. E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, 1972. 1

[10] S. Ikehata, H. Yang, and Y. Furukawa. Structured indoor modeling. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1323–1331, 2015. 6

[11] H. Isack and Y. Boykov. Energy-based geometric multi-model fitting. *International Journal of Computer Vision*, 97(2):123–147, 2012. 1

[12] P. Jaccard. Étude comparative de la distribution florale dans une portion des Alpes et des Jura. *Bulletin del la Société Vaudoise des Sciences Naturelles*, 37:547–579, 1901. 2

[13] L. Magri and A. Fusiello. T-Linkage: A continuous relaxation of J-Linkage for multi-model fitting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3954–3961, June 2014. 3

[14] C. Mura, O. Mattausch, A. J. Villanueva, E. Gobbetti, and R. Pajarola. Robust reconstruction of interior building structures with multiple rooms under clutter and occlusions. In *Computer-Aided Design and Computer Graphics (CAD/Graphics), 2013 International Conference on*, pages 52–59. IEEE, 2013. 1

[15] C. Mura, O. Mattausch, A. J. Villanueva, E. Gobbetti, and R. Pajarola. Automatic room detection and reconstruction in cluttered indoor environments with complex room layouts. *Computers & Graphics*, 44:20–32, 2014. 6

[16] S. Murali, P. Speciale, M. R. Oswald, and M. Pollefeys. Indoor scan2bim: Building information models of house interiors. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pages 6126–6133. IEEE, 2017. 1

[17] S. Ochmann, R. Vock, R. Wessel, and R. Klein. Automatic reconstruction of parametric building models from indoor point clouds. *Computers & Graphics*, 54:94–103, 2016. 1

[18] B. Okorn, X. Xiong, B. Akinci, and D. Huber. Toward automated modeling of floor plans. In *Proceedings of the symposium on 3D data processing, visualization and transmission*, volume 2, 2010. 1

[19] M. Previtali, L. Barazzetti, R. Brumana, and M. Scaioni. Towards automatic indoor reconstruction of cluttered building rooms from point clouds. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2(5):281, 2014. 1

[20] V. Sanchez and A. Zakhor. Planar 3d modeling of building interiors from point cloud data. In *Image Processing (ICIP), 2012 19th IEEE International Conference on*, pages 1777–1780. IEEE, 2012. 1

[21] C. H. Teixeira, A. Silva, and W. Meira Jr. Min-hash fingerprints for graph kernels: A trade-off among accuracy, efficiency, and compression. *Journal of Information and Data Management*, 3(3):227, 2012. 3

[22] R. Toldo and A. Fusiello. Robust multiple structures estimation with J-Linkage. In *Proceedings of the European Conference on Computer Vision*, pages 537–547, 2008. 2, 3

[23] E. Turner and A. Zakhor. Watertight as-built architectural floor plans generated from laser range data. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*, pages 316–323. IEEE, 2012. 4, 7

[24] W. Wang and W. Gao. Efficient multi-plane extraction from massive 3d points for modeling large-scale urban scenes. *The Visual Computer*, pages 1–14, 2018. 1

[25] J. Xiao and Y. Furukawa. Reconstructing the worlds museums. *International journal of computer vision*, 110(3):243–258, 2014. 1