Simplification of Polygons from Point Cloud Data for Automated Floorplan Generation

Antonio Carlone[1] and Felix Gabler[1]
[1]Fachgebiet für Bauinformatik, Technische Universität Berlin, Gustav-Meyer-Allee 25, 13355 Berlin, Germany
E-mail(s): carlone@tu-berlin.de, f.gabler@tu-berlin.de

Abstract: This paper investigates methodologies for identifying significant edges in polygons, with a particular focus on applications in floor plan analysis. The primary objective is to develop and evaluate an algorithm that can accurately simplify and refine polygonal representations of rooms, ensuring that key structural details are preserved. The study introduces a novel simplification algorithm that combines holistic filtering, regression techniques, and a mechanism for closing polygons. This algorithm is designed to address the shortcomings of existing methods by providing a more comprehensive approach to reducing complexity while maintaining essential geometric features. The algorithm is validated through testing on two datasets of floor plans derived from real laser scans. The findings contribute to the field of automated floor plan generation, offering a scale-independent solution for simplifying redundant polygonal data while preserving spatial accuracy.
Keywords: indoor environment, polygon simplification, automated floorplan generation, scan-to-BIM

Erschienen in Tagungsband 35. Forum Bauinformatik 2024, Hamburg, Deutschland, DOI: 10.15480/882.13546

1 Introduction

As digitalization continues to transform the Architecture, Engineering, Construction, and Operations (AECO) industry, floor plans have evolved from hand-drawn sketches to derivations from digital BIM models. Automation has become integral to various processes, including the creation of Indoor Floor Plans (IFPs). IFPs are crucial in fields such as architectural design, indoor navigation, and safety assessment. However, the creation of IFPs is not only of relevance during the planning phase of new buildings, but is also essential for depicting existing buildings and as-built documentation. Traditionally, the manual creation of IFPs is labor-intensive and time-consuming, leading to a growing interest in their automatic generation. Consequently, scan-to-BIM approaches have emerged as a popular solution [1]. Creating floor plans from laser scans allows for the recreation of existing building layouts, often resulting in models that are more accurate than the original plans due to potential building inaccuracies [2].

2 Related Research

To automate IFP creation and ensure their consistency with original indoor layouts, many methods for generating IFPs from indoor 3D point clouds have been proposed. [3]
There are several sources of inaccuracies for room polygons compared to the planned building model. Inaccuracies occur during the construction phase but can also be caused by measurement errors and the standard deviation of laser scanners [2], [4]. In the worst case, these errors accumulate. The major challenges in dealing with inaccuracies in 2D floor plans involves handling corners and more vertices within a polygon than necessary [5], [6].
Existing approaches that could provide a solution are the Douglas-Peucker algorithm [7], the RANSAC algorithm [8], the alpha shapes algorithm [9], the Line Segment Detector (LSD) [10], and the algorithm proposed by Phan, Huynh, and Truong-Hong [11].
The Douglas-Peucker algorithm is a line reduction algorithm that takes a polygonal chain as input. It detects significant vertices and connects them to a new polygonal chain [7]. This algorithm cannot repair corners that are not represented by a vertex, because it cannot generate new vertices. Bad representations of corners often occur in measured room polygons and hence the algorithm is not suitable for simplification after pre-processing [6].
The RANSAC algorithm is used to perform a regression on a set of data points omitting points of gross errors. It can be applied in 2D and 3D, but does not aim to simplify the data points to exactly one point per corner and therefore may not simplify it to the desired degree [8].
The alpha shape algorithm proposed by Edelsbrunner, Kirkpatrick, and Seidel [9] finds a closed polygonal chain in a set of points that encloses every point of the set. Varying the alpha value of the algorithm defines the degree of fineness. This approach does not simplify the polygon to a degree where only the clear corners are represented by a vertex and therefore is not suitable here while still being used for pre-processing.
The LSD works in the field of line recognition in pictures, therefore is not suitable to simplify room polygons [10]. Nevertheless, principle steps of this algorithm can be adopted to the current problem.
Phan, Huynh, and Truong-Hong [11] use the RANSAC and alpha shapes algorithm to generate 2D floor plans from 3D point clouds. Their simplification of the room polygons needs an absolute length value whereas our approach works with percentages relative to the room polygons perimeter, making it scalable.

3 Simplification Algorithm

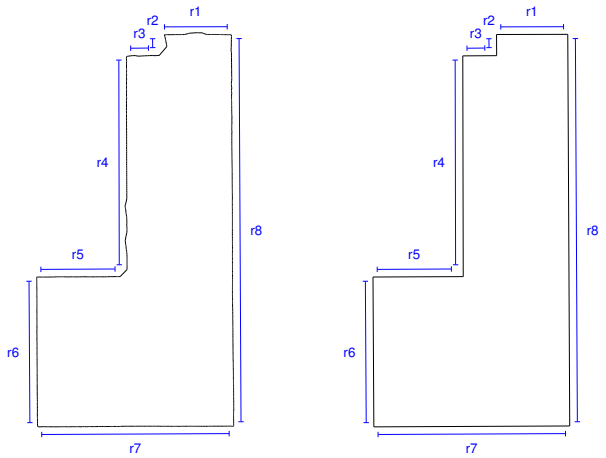
In this chapter, a new algorithm is presented to simplify a polygon representing a room in a floor plan. First, the research objectives are defined in Section 3.1 followed by the definition of the input data in Section 3.2. Next the processing stages are explained in Section 3.3 and finally, examples and limitations are discussed in Section 3.4.

3.1 Objectives

While point cloud data naturally consists of a large number of points, polygons benefit from as less points as needed. The description of the footprint of a room usually only requires one vertex per

corner and a linear connection between them. Figure 1 shows two polygons in different simplification stages of the same room. The polygon depicted in Figure 1a represents the concave hull derived from a segmented and sliced point cloud, consisting of 1192 vertices. The polygon in Figure 1b represents the same room. It has been modelled using a CAD-tool and consists of only 8 vertices.

The polygons in Figure 1 are divided into regions (r1 to r8) and corners (e.g. r1/r2) between them. While the region r4 is represented by exactly one edge for the modelled polygon, the region r4 of the left polygon consists of multiple. Corners however, see corner r4/r5 for the left polygon, may show diagonal characteristics and may not be clearly presented. Depending on which information is of interest, e.g. the area of the polygon/room, additional vertices do not contribute to information enrichment.



(a) Polygon from point cloud with 1192 vertices.

(b) The same polygon modelled with 8 vertices.

Figure 1: Comparison of two room polygons in different quality stages.

This paper aims to offer a solution to the simplification of closed planar polygons derived from point clouds. The main focus is on reducing the number of vertices while sharpening corners at the same time. Limiting the field of application to indoor situations of the built environment allows focusing on typical room geometries. Floor plans with exclusively one straight line per region as area boundaries are classified as typical room geometries while occurrences of arcs or diagonal lines are referred to as irregularities [12]. While the simplest room geometry is a rectangle, more complicated polygons may occur. Considering the possibility of slanted walls means that we are not restricted to the manhattan world assumption. Taking building elements such as columns into consideration, room polygons may have protrusions. Niches however, may be created by recesses. It is assumed that combinations of these features cover the majority of room geometries.

3.2 Preliminaries

Polygons are the input of our algorithm. They must neither be self-intersecting nor self-touching. However, polygons may have holes. Therefore polygons must consist of one exterior boundary and may have none to multiple intersecting-free interior boundaries. As interior and exterior boundaries,

both are closed edge-loops, they will be treated equally. Each single polygonal chain is processed individually and is referred to as polygon in the following.

3.3 Processing Stages

Our algorithm consists of three processing stages, as shown in the first column of Table 1. The first stage, the Holistic Filter, receives a closed polygon as input and involves the most important step of the algorithm, the identification of significant edges. The result of the Holistic Filter, a collection of significant edges, is further processed by a Regression on Sectors. Resulting line segments are then connected to achieve a closed polygonal chain.

Table 1: Processing Stages and Parameter Configuration of the Simplification Algorithm determined from the two datasets.

Processing Stage	Parameter	Value
Holistic Filter	angle threshold length threshold	2.7° 10% of the perimeter
Regression on Sectors	sector angle threshold on edge threshold	45° 0.787% of the perimeter
Connecting Representatives	distance margin complete threshold	10% of the perimeter 90% of the edge

3.3.1 Holistic Filter

The Holistic Filter represents the core of the presented approach. Its function is to identify and collect significant edges of the polygon. It is assumed that a subset of edges is sufficient to approximately represent the polygon while preserving geometric information such as the area. The input is a closed polygon and its output is a set of edges. We define them as *significant edges*. However, the original order of the vertices is preserved for the last step of our algorithm.
In a first step, for each edge a group of supporting edges including itself is identified. All edges whose angle to the examined edge is smaller than a specific angle threshold are members of this group. Using an angle threshold of 2.7° (see table 1) showed the best results for the test data. In a second step, the accumulated length of the resulting sets of edges is evaluated. If a length of a group is greater than 10% of the perimeter of the original polygon, the corresponding edge is a *significant edge*. Filtering edges by its group length instead of its individual edge length ensures the identification of protrusion edges as *significant edges*. It is assumed, that protrusion edges have a similar orientation as other edges. The set of *significant edges* is passed to the next step.

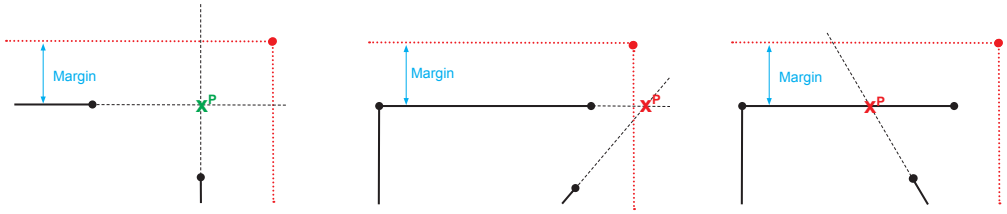
3.3.2 Regression on Sectors

The second step of our algorithm groups the significant edges to *sectors* and calculates a representative edge per *sector*. *Significant edges* belong to the same *sector* if two conditions are met: First, the distance of the vertices of two edges must be below a defined threshold (see *on edge threshold* in table 1), relative to the perimeter of the polygon. Second, the angle between the two edges, must be less than the defined *sector angle threshold* of 45°.
For each *sector*, a weighted regression, considering the length of each edge, is performed. The result of each regression is one edge: the *representative* of the corresponding *sector*. Consequently, in case

of multiple edges per *sector*, the number of edges is reduced to one. The result of this processing step is a number of *sectors*, each represented by a single edge, the *representative*.

3.3.3 Connecting Representatives

The result of the previous processing step is approximately representing the original polygon. However, the edges must be connected to form a closed polygonal chain. This step is executed here. For each pair of adjacent *representatives*, defined by the order in the list, an intersection point is calculated and evaluated for its validity (see Figure 2). A major problem for existing algorithms lies in identifying rooms with missing or unsatisfactory measurement data for certain walls [5]. This approach will be able to handle such an example if the surrounding walls are properly represented by measurement data.
A region may consist solely of edges that do not collect enough group length to pass the Holistic Filter. The resulting *sector* of these regions will not have a *representative* (see r2 in the polygon of Figure 3c). This issue is detected in this approach if two neighbouring *sectors* form a too tight inside angle (see Figure 2b) or if a wall edge is shortened by the intersection point (considering a specific threshold, referred to as complete threshold see Figure 2c and Table 1). A very tight inner angle of a corner is considered to be an unusual feature of a floor plan (see Chapter 3.1) and is detected if the intersection point lies outside of the polygon by a distance margin times the perimeter of the polygon (see Table 1). The second criterion is employed, because we assume, there is no wall, that starts in the middle of another wall.



(a) Valid

(b) Invalid: Outside of margin

(c) Invalid: Intersecting significant edge

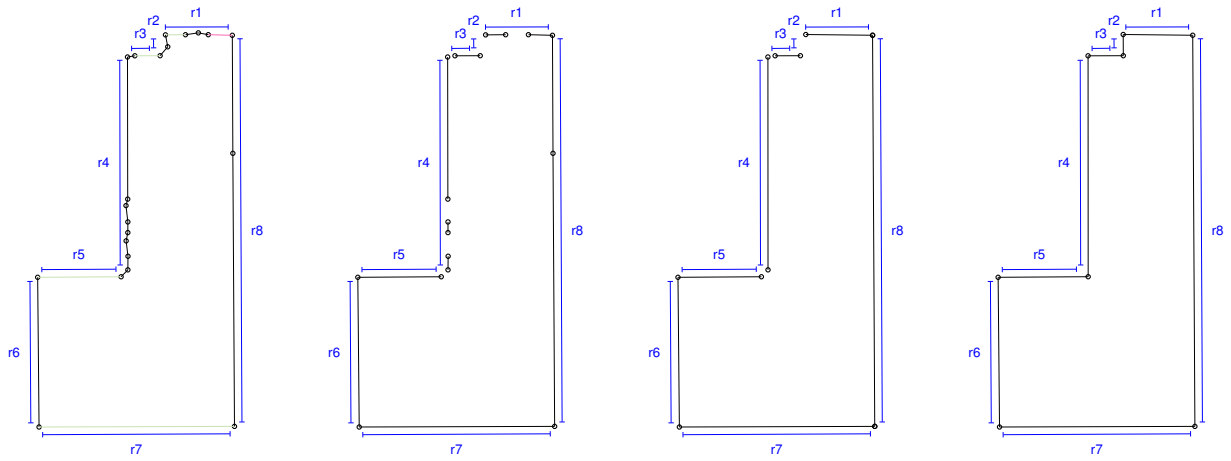
Figure 2: Possible cases of intersection points.

If a missing *representative* is detected, it is recreated. This is done by connecting the last vertex of the prior *representative* to the first vertex of the following *representative*. The resulting connecting line e is rotated to the angle of the *representative*, whose angle differs the least from the angle of e. This can be done under the assumption of parallel walls.

3.4 Examples and Limitations

The proposed algorithm has been tested using real-life data provided by ReconTOP project partner Elsafty [13]. The point cloud data captured with a terrestrial laser scanner has been preprocessed into two test datasets. The preprocessing involves the registration, segmentation, and planar slicing of the point cloud data. Our input polygons are the concave hulls of point sets per room, using the Alpha shape algorithm [9]. The provided data consists of 27 room polygons from 5 different apartments. The polygons of the first dataset, consisting of 1000 to 2000 vertices, have been simplified using again the

Alpha shape algorithm. The polygons of the second dataset consist of 15 to 50 vertices. Taking the two different quality stages of the room polygons into account, we tested our algorithm on 54 different polygons. Throughout this chapter, the selected polygon in Figure 1a will be used in its simplified version as input to demonstrate the progress of the algorithm. The simplified version is chosen for the demonstration as its longer edges are easier to see for the reader. This polygon has multiple corners in r2/r3, r3/r4 and r4/r5 that connect regions with diagonal edges. That means, the polygon has more vertices than would be necessary to properly represent its geometry.
The following example illustrates how the Holistic Filter ensures the representation of protrusions. The pink edge in the polygon of Figure 3a shows an edge from a protrusion that is short and therefore would be filtered by every plain length filtering. The accumulated length of the pink and every similarly oriented edge (green edges of the polygon shown in Figure 3a) though, is longer than the specified length threshold and therefore the pink edge is not filtered by the Holistic Filter. The polygon after the Holistic Filter is shown in Figure 3b. It shows an open polygon which has no diagonal connections of corners opposed to the input polygon in Figure 1a.
The polygon in Figure 3c shows the combined edges of the Holistic Filter after the Regression on Sectors and the polygon in Figure 3d shows the resulting closed polygon with the recreated edge in r2 after Connecting Representatives.



(a) Protrusion edge in pink (top right), similar edges in green.

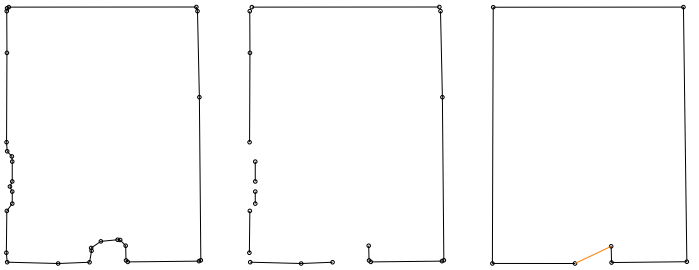
(b) Polygon after Holistic Filter.

(c) Polygon after Regression on Sectors.

(d) Polygon after Closing.

Figure 3: Different processing stages of the example room polygon.

The main limitation of the methodology occurs if two or more neighbouring *sectors* lack a *representative*, meaning all their edges got filtered out by the Holistic Filter (see Figure 4b). The algorithm recognises that two or more neighbouring *sectors* are missing after first noticing that one *sector* is missing. If either the start or end point of this *sectors* recreated *representative* lies outside of the distance margin (see Figure 2b) it is detected that both neighbouring *sectors* are missing. Subsequently the polygon is closed by connecting the neighbouring available *representatives* of the empty *sectors* as shown in orange in Figure 4c. The resulting polygon may then deviate from the expected outcome and be unsatisfactory. This only occurred on an example of the further simplified polygon data set.



(a) The input polygon.

(b) A missing pair of *sectors* (bottom center).

(c) Result polygon with provisional edge.

Figure 4: The processing stages of the polygon from the dataset, for which the algorithm does not find a satisfactory solution.

4 Conclusion and Outlook

A solution for the simplification of closed 2D floor plan polygons, with a particular focus on corners and the reduction of vertices is proposed. The room polygons are divided into their exterior and interior boundaries, meaning that each boundary is considered independently rather than in relation to the entire polygon. In 9 out of 54 polygons unrepresented *sectors* were identified and artificially recreated. 8 of those 9 polygons have only one missing representative in a row and are closed in a satisfactory manner. The polygon shown in Figure 4 is the only polygon that lacks a representative edge in two neighbouring *sectors*. The polygon is closed by an edge connecting the *sectors* before and after the two empty *sectors*. However, this result is unsatisfactory because, despite significant vertex reduction prior to processing, the original shape of the room was lost. In conclusion a total of 54 polygons of varying number of vertices per region were processed using this approach and using different parameter configurations. A parameter configuration that satisfactorily processed the 27 not further simplified polygons in a satisfactory manner has been found (see column "values" of Table 1), fine tuning initial assumptions for the values in a case study. This is a promising result and shows that the algorithm should be used for polygons with as much information (high number of vertices) as possible rather than polygons being preprocessed by another algorithm. The goal of our approach is to find a parameter configuration that works for every polygon representing a room in a floor plan. The user is not ought to adjust the parameters himself but rather get presented a finished algorithm. To achieve this, further research on a larger dataset of room polygons is necessary. Future developments of this algorithm could expand its application to entire floor plans or extend to 3D space, using concave hulls consisting of faces rather than edges.

Acknowledgements

Special thanks to A. Elsafty from the ReconTOP research project for providing the preprocessed point clouds. The ReconTOP research project is a joint research project executed together with Prof. Georg Suter, TU Wien, Austria, Prof. Timo Hartmann, TU Berlin, and the Bauinformatik group at TU Berlin. ReconTOP is funded by the Austrian Science Fund (FWF - Project number I 5171-N) and by the German Research Foundation (DFG - Project number 454008779).

The research presented in this paper results from the student research project conducted by Antonio Carlone [14] and is closely linked to the ReconTOP research project.

References

[1] F. Gabler and W. Huhnt, “Where is the end of the wall: Decomposition of air and material into spaces and building components”, in Aug. 2023. DOI: 10.1007/978-3-031-32515-1_27.

[2] S. Yang, M. Hou, and S. Li, “Three-dimensional point cloud semantic segmentation for cultural heritage: A comprehensive review”, *Remote Sensing*, vol. 15, no. 3, p. 548, 2023. DOI: https://doi.org/10.3390/rs15030548.

[3] X. Gao, R. Yang, J. Tan, and Y. Liu, “Floor plan reconstruction from indoor 3d point clouds using iterative ransac line segmentation”, *Journal of Building Engineering*, vol. 89, 2024. DOI: https://doi.org/10.1016/j.jobe.2024.109238.

[4] .

[5] H. Fang, F. Lafarge, C. Pan, and H. Huang, “Floorplan generation from 3d point clouds: A space partitioning approach”, *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 175, pp. 44–55, 2021. DOI: https://doi.org/10.1016/j.isprsjprs.2021.02.012.

[6] J.-P. Bauchet, R. Sulzer, F. Lafarge, and Y. Tarabalka, “Simplicity: Reconstructing buildings with simple regularized 3d models”, *arXiv preprint arXiv:2404.08104*, 2024. DOI: https://doi.org/10.48550/arXiv.2404.08104.

[7] D. H. Douglas and T. K. Peucker, “Algorithms for the reduction of the number of points required to represent a digitized line or its caricature”, *Cartographica: The International Journal for Geographic Information and Geovisualization*, vol. 10, no. 2, pp. 112–122, 1973. DOI: 10.3138/FM57-6770-U75U-7727.

[8] R. C. Bolles and M. A. Fischler, “A ransac-based approach to model fitting and its application to finding cylinders in range data”, in *International Joint Conference on Artificial Intelligence*, 1981.

[9] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel, “On the shape of a set of points in the plane”, *IEEE Transactions on Information Theory*, vol. 29, no. 4, pp. 551–559, 1983. DOI: 10.1109/TIT.1983.1056714.

[10] R. Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, “Lsd: A line segment detector”, *Image Processing On Line*, vol. 2, pp. 35–55, Mar. 2012. DOI: 10.5201/ipol.2012.gjmr-lsd.

[11] A. T. Phan, T. Huynh, and L. Truong-Hong, “Generating 2d building floors from 3d point clouds”, in Dec. 2023, pp. 1660–1668. DOI: 10.1007/978-981-99-7434-4_179.

[12] Z. Lu, T. Wang, J. Guo, *et al.*, “Data-driven floor plan understanding in rural residential buildings via deep recognition”, *Information Sciences*, vol. 567, pp. 58–74, 2021. DOI: https://doi.org/10.1016/j.ins.2021.03.032.

[13] A. Elsafty, Room Point Clouds, Feb. 2024.

[14] A. Carlone, “Identification of significant edges in polygons”, Student Research Project, Technische Universität Berlin, May 2024.