GRAMMAR-SUPPORTED 3D INDOOR RECONSTRUCTION FROM POINT CLOUDS FOR "AS-BUILT" BIM

S. Becker [*], M. Peter, D. Fritsch

Institute for Photogrammetry, University of Stuttgart, 70174 Stuttgart, Germany – forename.lastname@ifp.uni-stuttgart.de

Commission III, WG III/4

KEY WORDS: Building, Modeling, Abstraction, Prediction, Automation

ABSTRACT:

The paper presents a grammar-based approach for the robust automatic reconstruction of 3D interiors from raw point clouds. The core of the approach is a 3D indoor grammar which is an extension of our previously published grammar concept for the modeling of 2D floor plans. The grammar allows for the modeling of buildings whose horizontal, continuous floors are traversed by hallways providing access to the rooms as it is the case for most office buildings or public buildings like schools, hospitals or hotels. The grammar is designed in such way that it can be embedded in an iterative automatic learning process providing a seamless transition from LOD3 to LOD4 building models. Starting from an initial low-level grammar, automatically derived from the window representations of an available LOD3 building model, hypotheses about indoor geometries can be generated. The hypothesized indoor geometries are checked against observation data - here 3D point clouds - collected in the interior of the building. The verified and accepted geometries form the basis for an automatic update of the initial grammar. By this, the knowledge content of the initial grammar is enriched, leading to a grammar with increased quality. This higher-level grammar can then be applied to predict realistic geometries to building parts where only sparse observation data are available. Thus, our approach allows for the robust generation of complete 3D indoor models whose quality can be improved continuously as soon as new observation data are fed into the grammar-based reconstruction process. The feasibility of our approach is demonstrated based on a real-world example.

1. INTRODUCTION

For years, research on the automatic and semi-automatic refinement of LOD2 building models to LOD3 models has been an interesting topic reaching a stage, now, where first approaches are to be integrated into commercial products. The next step would be to enable an automatic shift from LOD3 to LOD4 models in a similar way. 3D indoor models can support daily life in many fields. Examples are indoor navigation and positioning, emergency services, crowd management, architectural planning, simulations etc. Moreover, 3D indoor models are a valuable basis for efficiently handling restoration and maintenance measures. In the world of Building Information Modeling (BIM), 3D indoor models feature high geometric details and are equipped with rich semantics. In current literature, the term "as-built" BIM is used when dealing with BIM representations which document the state of the buildings at the moment of the survey (Hichri et al., 2013). Still, as-built BIM creation is a largely *manual* and, thus, time-consuming and subjective process (Tang et al., 2010).

In the GIS world, in contrast, the vast majority of 3D indoor representations are pure geometry models with limited geometric detail. This is owed to the fact that, here, the focus is on the *automatic* derivation of indoor geometries from observation data. Typical problems arising in this context are due to erroneous and incomplete observations caused by bad conditions for data collection, or by using sensors with low accuracy and resolution. However, building interiors are subject to numerous geometric and topological conditions which can be used to support the reconstruction of interiors. Powerful tools to facilitate this are formal grammars. The huge potential of formal grammars lies in the compact description of object knowledge, and the generative part which allows for efficient procedural modeling approaches.

In this paper, we present a grammar-based approach for the robust reconstruction of 3D interiors from point clouds. The approach is robust as it can cope with raw, unfiltered, and even incomplete point clouds. This is implemented by embedding the reconstruction process into an automatic learning and verification loop: An initial grammar derived from currently available observations is used to generate hypotheses about possible indoor geometries. As soon as new observations are available, the previously generated hypotheses can be checked against them, and the new observations can be used to automatically update the initial grammar. Continuing this loop finally results in a high-level grammar which is able to generate indoor geometries of high reliability even for those building parts for which few or no observation data are available. The approach can be used to refine existing LOD3 building models to LOD4 models, and reveals potential to significantly support as-built BIM creation.

The contributions of the paper are: (1) a 3D indoor grammar to support reconstruction or as a means to formally describe 3D indoor models in a compact way; (2) a grammar-based robust approach enabling a seamless transition from LOD3 to LOD4 building models based on the interpretation of 3D point clouds.

The paper is structured as follows: Section 2 gives an overview of related work on indoor reconstruction. The definition of our 3D indoor grammar is given in section 3. Section 4 describes how the grammar can be automatically derived from observation data during a parsing process; its application to a real-world example is shown in section 5. Conclusions are given in section 6.

2. RELATED WORK

Point clouds are established intermediate products of a variety of sensors and concepts, e.g. laser scanning, range cameras like the

[*] Corresponding author.

Microsoft Kinect, or Dense Image Matching. As all of these methods result in un-interpreted point clouds, many approaches exist which deal with the reconstruction of semantically enhanced boundary representation models from point clouds. Generally, parallelism and rectangularity are very prominent rules used in man-made construction which is why most reconstruction methods build on the well-known Manhattan World constraints. Jenke et al. (2009) describe the segmentation of the point cloud to planar patches and the retrieval of pairwise perpendicular patches. Embedding those patches in a graph structure and its analysis enables the detection of cuboids which subsequently can be merged to corridors and rooms. In a series of papers (e.g. Adan & Huber, 2011) a method is presented which bases on histogram analysis. The ceiling and ground planes are detected by analyzing a histogram along the z-axis. By use of the Hough transform in a xy-histogram, the rooms' ground plans are reconstructed. Furthermore, the detection of door and window openings in the extruded walls is described. Correspondingly, Budroni (2013) finds interior walls by means of a plane-sweep-algorithm assuming the walls to be rectangular or parallel.

In contrast to the low-level Manhattan World constraints, an efficient and compact possibility to formalize high-level object knowledge is given by production systems such as formal grammars. Early developments of rule-based production systems are given by Matsuyama & Hwang (1990) and Stilla & Michaelsen (1997) for the analysis of man-made structures in aerial images. Generally, formal grammars have been applied successfully for modeling geometric structures. Prusinkiewicz & Lindenmayer (1990) focus on line structures by e.g. simulating growth processes of plants through so-called *Lindenmayer systems* (L-systems). Parish & Müller (2001) applied the concept of L-systems to modeling streets and 3D building models. Since street networks behave quite similar to the branching character of growing plants, streets can be modeled very efficiently by L-systems. Problems, however, arise with modeling buildings: A building represents rather an iterative partitioning of available space than an unrestricted growing process. Furthermore, the integration of geometric conditions (e.g. parallelism, rectangularity) lead to long and intricate production rules. Grammars which are more appropriate to model architectural structures are *split grammars*. A split grammar uses split rules to partition a simple shape into more complex ones. Wonka et al. (2003) proposed a split grammar for façade reconstruction. Building on this, the CGA Shape Grammar of Müller et al. (2006) allows for the procedural generation of highly detailed 3D building models. A general problem of the grammars, mentioned so far, is that the production rules have to be set up manually which is very time-consuming and requires expert knowledge. Therefore, some approaches follow the principle of inverse procedural modeling, i.e., they try to automatically derive grammar rules from observation data. Examples are given by Müller et al. (2007), Aliaga et al. (2007), Becker (2009), Bokeloh et al. (2010) and van Gool et al. (2013).

For the modeling of building interiors, grammars are still very new. To our knowledge, Gröger & Plümer (2010) were the first ones proposing a grammar for the generation of complete 3D indoor models. The grammar contains mainly split rules which are applied to boxes and faces. The rules have to be predefined manually, and it is not possible to automatically adapt the grammar to observation data for modeling real interiors. Similarly, Marson & Musse (2010) as well as Mirahmadi & Shami (2012) employ higher-level knowledge modeled as tree maps for the automatic generation of residential house layouts. Khoshelham & Díaz-Vilariño (2014) present a parametric shape grammar for modeling indoor structures using merged cuboids.

3. CONCEPT OF 3D INDOOR GRAMMAR

Our 3D indoor grammar, which is an extension of our previously published indoor grammar for modelling 2D floor plans, is designed to reflect basic architectural principles which are normally valid for office buildings or public buildings like schools, hospitals or hotels: Typically, the geometric structure of such building interiors is organized through a horizontal partitioning of the building's body into floors, and a vertical partitioning of each floor into rooms. Particularly, the following properties of the building interiors are crucial for our grammar design: (1) To ensure convenient access to the rooms, the buildings are usually traversed by a system of hallways. (2) The system of hallways divides each floor into hallway-spaces and non-hallway spaces. Non-hallway spaces can be further divided into smaller room units which are mostly arranged in a linear sequence parallel to the adjacent hallway.

Properties 1 and 2 give reasons for pursuing two different modeling strategies: The course of the hallways, on the one hand, reminds of a network-like propagation of linear structures. The layout of the rooms, on the other hand, can be efficiently generated by a spatial partitioning applied to the interspaces of the hallway network. Taking into account the different characteristics of hallway and room layouts, our indoor grammar $G^{indoor}=(G^{hallways},G^{rooms})$ is composed of two grammars addressing the modeling of hallways and rooms separately: The grammar for modeling hallways bases on an enriched L-system, room configurations are generated by means of a split grammar. Together, the L-system and the split grammar allow for the complete formal description of building interiors.

Next, our grammar concept is introduced in more detail: Section 3.1 gives a formal definition of the indoor grammar, section 3.2 describes how the grammar can be applied.

3.1 Grammar Definition

To be flexible towards the huge variety of different indoor designs, both parts of our 3D indoor grammar $G^{indoor}=(G^{hallways}, G^{rooms})$ - the L-system $G^{hallways}$ and the split grammar G^{rooms} - are constructed as generic template grammars which can be automatically adapted to individual buildings based on observation data. The following sections 3.1.1 and 3.1.2 present the formal definitions of the two template grammars.

3.1.1 L-System for Modeling Hallway Networks

The design of our L-system $G^{hallways}=(V,\omega,P)$ consisting of V, a set of attributed symbols also called *modules*, the axiom ω, and the production rules P, bases on the approach for modeling 2D street networks developed by Parish & Müller (2001). Their concept of an enriched, though two-dimensional, L-system meets our requirements of a template-like grammar which can be easily adapted to observation data. The main idea is to organize the setting of attributes (e.g. length and orientation of linear structures), probabilities and constraints (induced by the geometric environment in which the structures are to be embedded) through *external functions*. Thus, all variable components are uncoupled from the generative grammar part, i.e. the production rules, which can stay fix after being defined once.

We distinguish three external functions which, together, fully control the behavior of the L-system. The first two functions *ActivationControl* and *LayoutSetting* control the growth of the hallway system on a global level: *ActivationControl* determines the sequential order in which the hallway network is developed in horizontal and vertical direction; *LayoutSetting* effects that the hallways follow a specific overall layout pattern by setting the attributes of the generic hallway modules. Since the attributes are determined due to their probabilities, slightly different layouts can be produced, however, in the same overall style.

The third external function, *ConsistencyConstraints*, controls the behavior of the hallway system on a local level. It ensures that the L-system is environmentally sensitive as well as self-sensitive. To obtain environmental sensitivity, the function checks the hallways as proposed by the external function *LayoutSetting* against the given exterior building shell. Geometric conflicts, e.g. caused by a hallway breaking through the building shell, are tried to be resolved by fitting the hallway's length and orientation to its local environment. Hallways which cannot be inserted at all are removed. Considering the fact that, due to technical construction principles, vertical hallway segments like stairways or elevators are usually restricted to the same horizontal position in all floors of a building, the vertical growth of the hallway network, representing such a stairway or elevator, is additionally controlled by means of a so-called "control image". Such an image describes the 2D location-based probability of the hallway network to be developed in vertical direction. To obtain self-sensitivity, *ConsistencyConstraints* checks the proposed hallways against already existing hallways: Intersections are identified and short dead-ends are removed; hallway segments which are slightly too short to intersect are extended. The result is a topologically correct hallway network.

The generative part of the L-system consists of following production rules, which are an adaptation and 3D extension of the 2D L-system suggested by Parish & Müller (2001):

- ω: **R**(*ACTIVE*)**?I**(θ_{init},*UNASSIGNED*)
- p1: **R**(*mode*) > **?I**(θ,*state*) : *state==SUCCEED && mode==ACTIVE*
  {***LayoutSetting***(*mode*,θ) sets θ_p[0-4]} → +(θ.*angle*)**F**(θ.*len*)
  B$^{\mathbf{h}}$(*ACTIVE*, θ_p[1]) **B**$^{\mathbf{h}}$(*ACTIVE*, θ_p[2]) **B**$^{\mathbf{v}}$(*INACTIVE*, θ_p[3])
  B$^{\mathbf{v}}$(*INACTIVE*, θ_p[4]) **R**(*ACTIVE*)**?I**(θ_p[0],*UNASSIGNED*)
- p2: **R**(*mode*) > **?I**(θ,*state*) : *state==FAILED* → ε
- p3: **B**$^{\mathbf{h}}$(*mode*,θ) : *mode==ACTIVE* →[**R**(mode)**?I**(θ,*UNASSIGNED*)]
- p4: **?I**(θ,*state*) : *state==UNASSIGNED*
  {***ConsistencyConstraints***(θ) adjusts *state*, θ} → **?I**(θ,*state*)
- p5: **?I**(θ,*state*) : *state!=UNASSIGNED* → ε
- p6: **B**$^{\mathbf{v}}$(*mode*,θ) : *mode==INACTIVE*
  {***ActivationControl*** sets *mode*} → **B**$^{\mathbf{v}}$(*mode*,θ)
- p7: **B**$^{\mathbf{v}}$(*mode*,θ) : *state==SUCCEED && mode==ACTIVE*
  →[**Q**(mode)**?I**(θ,*UNASSIGNED*)]
- p8: **Q**(*mode*) > **?I**(θ,*state*) : *state==SUCCEED && mode==ACTIVE*
  {***LayoutSetting***(*mode*,θ) sets θ_p[0-3]} → +(θ.*angle*)**U**(θ.*len*)
  B$^{\mathbf{h}}$(*ACTIVE*, θ_p[1]) **B**$^{\mathbf{h}}$(*ACTIVE*, θ_p[2]) **B**$^{\mathbf{v}}$(*INACTIVE*, θ_p[3])
  R(*ACTIVE*)**?I**(θ_p[0],*UNASSIGNED*)
- p9: **Q**(*mode*) > **?I**(θ,*state*) : *state==FAILED* → ε

The rules distinguish two different kinds of hallway types: horizontal hallway segments represented in the L-system by **R**(*mode*)**?I**(θ,*state*), and vertical segments like staircases or elevators represented by **Q**(*mode*)**?I**(θ,*state*). **R** and **Q** are rule modules with the attribute *mode={ACTIVE, INACTIVE}*, **?I** is a query insertion module with the attributes *θ=(angle, length, width)* and *state={UNASSIGNED, SUCCEED, FAILED}*.

The axiom ω consists of a horizontal hallway segment. The first three production rules mainly address the production of horizontal hallway segments: p1 creates a horizontal hallway expressed by +(θ.*angle*)**F**(θ.*len*), at the end of which the buds for five branches are inserted pointing to the left, the right, in upward, downward and forward direction. The attributes of the branches are set by the external function *LayoutSetting* according to a specific layout pattern. p2 deletes a **R** module if *ConsistencyConstraints* has failed to fit the corresponding hallway into its geometric environment. p3 generates a new horizontal hallway segment at a branch position. Rules p4 and p5 adjust the attributes of the horizontal and vertical hallway segments set by *LayoutSetting* and decides whether the vertical branches will be further developed or not. If *Consistency-Constraints* is successful in fitting a hallway into its environment, the variable *state* is set to SUCCEED and the hallway can be created. By calling the external function *ActivationControl*, rule p6 ensures that branches pointing in vertical direction are not developed before all horizontal branches have been processed. The vertical growth process is controlled by rules p7 to p9 in analogy to the horizontal growth addressed in p1 to p3.

3.1.2 Split Grammar for Modeling Rooms

Our split grammar fulfills three tasks: (1) partitioning of the 3D building into floors; (2) partitioning of the floors into hallway spaces and non-hallway spaces; (3) partitioning of the non-hallway spaces in rooms. The grammar can be written as a four-tuple G^{rooms}=(*N,T,S,R*) consisting of the non-terminals *N*, the terminals *T*, the axiom *S*, and the production rules *R*.

The non-terminals and terminals of our grammar correspond to basic geometric primitives. The set of non-terminals *N*={*Space*} consists of the single element *Space* representing an arbitrary 3D solid which can be further divided. The axiom *S*=*Space* stands for an empty space (e.g. the body of the building) which is to be partitioned into smaller spaces. The terminals *T* describe solids that are not divisible any further. To distinguish from non-terminals, terminal symbols *space* start with lower case. Both non-terminals and terminals have attributes. They determine the space's geometric extent and probability. The production rules *R* are defined as replacement rules that perform a split, a merge or an instantiation. A split divides a *Space* into two *Space* elements along a partition plane. A merge is the inverse operation combining two adjacent *Space* elements to one. The application of split and merge rules follows the idea of cell decomposition (Kada, 2007) which automatically provides knowledge about neighborhood relations between the spaces and ensures a topologically correct reconstruction. We define six rule types:

- $\mathrm{R}_{\mathrm{i}}^{\mathrm{SingleSplit}} : Space \rightarrow Split^{Space}\left(\boldsymbol{n}_i, d_i\right)$

  $$\text{with } Split^{Space}\left(\boldsymbol{n}_i, d_i\right) := Split_i^{Space} = Space^{\,l}\; Space^{\,r}$$

- $\mathrm{R}_{\mathrm{i}}^{\mathrm{RepeatSplit}} : Space \rightarrow Split_i^{Space^{\,r}} \circ \cdots \circ Split_i^{Space^{\,r}} \circ Split_i^{Space}$
- $\mathrm{R}_{\mathrm{ij...k}}^{\mathrm{StringSplit}} : Space \rightarrow Split_k^{Space^{\,r}} \circ \cdots \circ Split_j^{Space^{\,r}} \circ Split_i^{Space}$
- $\mathrm{R}_{\mathrm{ab...c}}^{\mathrm{MultiSplit}} : Space \rightarrow Split_c^{*} \circ \cdots \circ Split_b^{*} \circ Split_a^{Space}$

  $$\text{with } * = Space \in \text{previously generated Spaces}$$

- $\mathrm{R}^{\mathrm{Merge}} : Space^{\,l} Space^{\,r} \rightarrow Merge^{Space^l, Space^r}$

  $$\text{with } Merge^{Space^l, Space^r} = Space^{\,l} \cup Space^{\,r}$$

- $\mathrm{R}^{\mathrm{Instantiation}} : Space \rightarrow space$

Rule type *SingleSplit* can be applied for all three partitioning tasks mentioned above. It performs a single split operation by replacing the non-terminal *Space* by a left and a right non-terminal *Space* ^{l} and *Space* ^{r}. These are the result of the function $Split^{Space}(\boldsymbol{n}_i,d_i)$. The superscript "Space" denotes that the split is

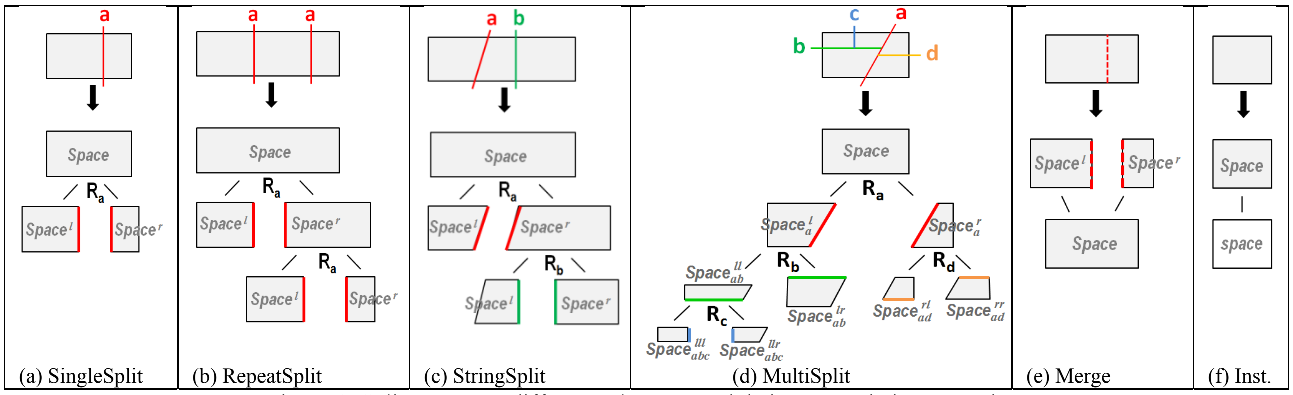

Figure 1. Split grammar: different rule types and their geometric interpretation.

applied to the non-terminal *Space*. Orientation and position of the corresponding partition plane are described by the rule parameters: normal vector ***n*** and a distance value *d*. ***n*** and *d* refer to a local coordinate system which is based on the *Space* to split.

Rule types *RepeatSplit*, *StringSplit* and *MultiSplit* can be used to group functionally related rooms to superior *room units*, and, thus, to explicitly store context information. Usually, rooms are arranged due to their functionality: Rooms having a strong semantic relationship are close to each other and build a unit. For example, in hotels or hospitals a typical room unit consists of a bedroom and a bathroom. Mathematically, such a grouping can be expressed by a concatenation of several *SingleSplits*. When each *SingleSplit* is applied to the right non-terminal *Space* ^{r} produced by the previous *SingleSplit*, the result is a linear sequence of rooms: While *RepeatSplit* generates a sequence of identical room units by repeating a single split operation, *StringSplit* is able to produce a sequence of different room units. Split operations which are applied for modeling non-linear room layouts can be aggregated within the rule type *MultiSplit*. In this case, the split operations can be applied to any of the previously generated Spaces. Based on simple examples, Figure 1 shows how the four split rules, the merge and the instantiation rule can be interpreted geometrically. For sake of brevity, here, a SingleSplit $R_i^{SingleSplit}$ is written as R_i.

The probability of the rules is described by an *a priori probability*, and a *context aware probability*. The a priori probability $P(R_i)$ of the rule R_i is the rule's relative frequency of occurrence. The context aware probability $P(R_j|R_i)$ is a conditional probability which models neighborhood relationships between rooms. For example, $P(R_j|R_i)$=0.5 states that with a probability of 50% rule R_j follows rule application R_i. We implement these probabilities by means of a Markov chain. The nodes of the Markov chain represent the rules. Edges describe neighborhood relationships or transitions between different rules. The probability for a transition from R_i to R_j is given by $\left(\mathrm{P}(R_j)/\mathrm{P}(R_i)\right)\cdot\left(\mathrm{P}(R_i \mid R_j)/\mathrm{P}(R_j \mid R_i)\right)$.

3.2 Grammar Application

The indoor grammar as defined in section 3.1 can be used to automatically generate hypotheses about 3D indoor geometries. For this purpose, the grammar rules are applied within a so-called production process. The starting point is a 3D model of the building's outer shell. The production process is composed of four stages: (1) the application of the L-system to install a 3D hallway network within the building shell; (2) the application of the split grammar to the building shell to generate floors; (3) the application of the split grammar to the floors to generate hallway spaces and non-hallway spaces; (4) the application of the split grammar to the non-hallway spaces to generate rooms. Stage 1 will be described in section 3.2.1, stages 2 to 4 in section 3.2.2.

3.2.1 Application of the L-System

The goal of the L-system is to generate a network of hallways based on which the floors of the building can be segmented into hallway spaces and non-hallway spaces. Before the L-system can be run, an axiom has to be set up which describes one or several initial hallway segments. Such initial hallway segments can be the result of interpreted observation data, or simply represent the centerlines of the building's footprint if no observation data are available. Having determined the axiom, the production rules as defined in section 3.1.1 can be applied. Since the stochastic part of the L-system has been shifted to an external function, the production rules can be processed in sequential order. The production process terminates when no branches are left within the hallway network to be further developed.

3.2.2 Application of the Split Grammar

The application of the split grammar to generate floors, and - within the floors - hallway spaces and non-hallway spaces is trivial. The floor generation simply requires a sequence of *SingleSplits* along horizontal planes at a distance which corresponds to the floor heights. The partitioning of the floors into hallways and non-hallways is also based on *SingleSplits* where the positions of the vertical split planes are defined by the hallway network resulting from the L-system.

In contrast, establishing room configurations within the non-hallway spaces is more complex. Since the split grammar is a stochastic grammar, different room configurations can be produced for one and the same non-hallway space. In order to get the most probable room configuration which fits both the non-hallway space and possibly existing room geometries best, we perform a constraint-augmented random walk on the Markov chain introduced in section 3.1.2. By means of the constraints, geometric restrictions can be considered which are derived from observations. For example, Peter et al. (2013) showed how the position of doors can be inferred from trace data. Based on such information, constraints can be set up which ensure that, e.g., no partition plane will be placed at a door's position. Further constraints considering the positions of windows can be introduced to prevent walls from intersecting window regions.

4. AUTOMATIC GRAMMAR INSTANTIATION BASED ON OBSERVATION DATA

The grammar introduced in section 3 is designed as a template grammar. It defines the syntax of possible rules, and - by means of the modules of the L-system and the non-terminals and terminals of the split grammar - it provides a formal description for those types of indoor geometries which are considered relevant to be represented in 3D indoor models. Thus, the template grammar already contains some basic geometric and semantic knowledge. To describe the individual characteristics of a specific building, the knowledge represented in the template grammar has to be enriched. An instance of an individual indoor grammar contains knowledge about indoor geometries and room arrangements which are characteristic for a specific building or building type. This object knowledge comprises geometric properties (e.g. the size of a room), topological properties (e.g. the connectivity of rooms), as well as semantic aspects (e.g. a functional grouping of rooms). Having such an individual indoor grammar available for a specific building, reliable hypotheses about possible indoor geometries can be generated.

In order to create an instance of an individual grammar, the rules as well as their attributes have to be set up. This instantiation process, which is based on the interpretation of observation data, can be done fully automatically during a parsing process for both the L-system (section 4.1) and the split grammar (section 4.2).

4.1 Instantiation of the L-System

The instantiation process of the L-system highly benefits from the concept of using external functions to uncouple variable components (e.g. attributes, probabilities) from the generative part of the L-system (i.e. the production rules). While the rules can stay fix, only the layout parameters and the control image have to be determined. The layout parameters comprise the lengths, the orientations and the widths of the hallways as well as the probability of occurrence of these parameters; the control image encodes a 2D location-based probability distribution indicating at which 2D position within a floor the development of the hallway network in vertical direction is probable.

The layout parameters and the control image can be derived from various observation data. In previous work (Philipp et al., 2014), we showed how 2D hallway polygons can be automatically derived from traces gathered by foot-mounted MEMS/IMUs, and that these hallway polygons can be used to determine the L-system's layout parameters and the corresponding probabilities. Moreover, traces which are recorded while changing floors are appropriate to detect stairways or elevators (Haala et al., 2011). Based on this information, control images can be generated.

Another option to derive the required layout parameters and the control image is the interpretation of 3D point clouds. In the following, we will present an iterative search algorithm for the automatic extraction of 2D hallway polygons from raw 3D point clouds. The algorithm bases on two assumptions: (1) hallways are likely to run parallel to the main axes of the building's footprint; (2) hallways are connected. According to the first assumption, the main axes of the building's footprint will determine the directions in which the point cloud is traversed to search for points representing hallway walls. According to the second assumption, in each iteration, the search area can be restricted to regions which are connected to previously detected hallways. The algorithm consists of following steps:

- *Step 0 (pre-processing step)*: Generate a 2D map as gray value image where each pixel defines a 2D grid element on the floor plane. (The size of the grid elements depends on the point sampling distance and accuracy requirements.) The gray values of the map represent the number of 3D points that fall into the corresponding grid element when being projected onto the floor plane.
- *Step 1*: Choose the longest axis of the building's footprint that has not been processed before as current search direction, and mark it as "processed".
- *Step 2*: Define the current search area as the region which results from shifting the hallway centerlines, accepted in the previous iteration, like a sweep line across the 2D map in and against the current search direction. (In the first iteration, the search area corresponds to the whole 2D map.)
- *Step 3*: In the current search area: Measure gray value profiles perpendicular to the current search direction. (The distance between the profiles can be chosen according to accuracy requirements.)
- *Step 4*: Search for local maxima in each profile. Resulting pixels refer to grid elements which contain hallway points that are likely to represent hallway walls. In the following, these points will be called hallway point candidates.
- *Step 5*: Estimate lines into the hallway point candidates by means of a RANSAC approach. The resulting lines represent possible hallway walls.
- *Step 6*: Cluster the hallway walls.
- *Step 7*: For each pair of parallel hallway walls whose distance exceeds a pre-defined minimum hallway width, and whose intermediary space does not contain more hallway point candidates than a pre-defined threshold: Derive the centerline.
- *Step 8*: Intersect the centerlines and the minimum 2D bounding box created for the (x,y)-positions of the hallway point candidates. As result, we get centerline segments.
- *Step 9*: Test the centerline segments against the hallway point candidates. Accept only those centerline segments for which the number of hallway point candidates exceeds a certain threshold.
- *Step 10*: If there are unprocessed axes of the building's footprint left, continue with *step 1*.
- *Step 11*: Fit rectangles around all accepted hallway centerlines into the hallway point candidates. This leads to 2D hallway polygons.
- *Step 12*: Analyze the 2D hallway polygons with respect to their length, orientation and width to set the layout parameters of the L-system.

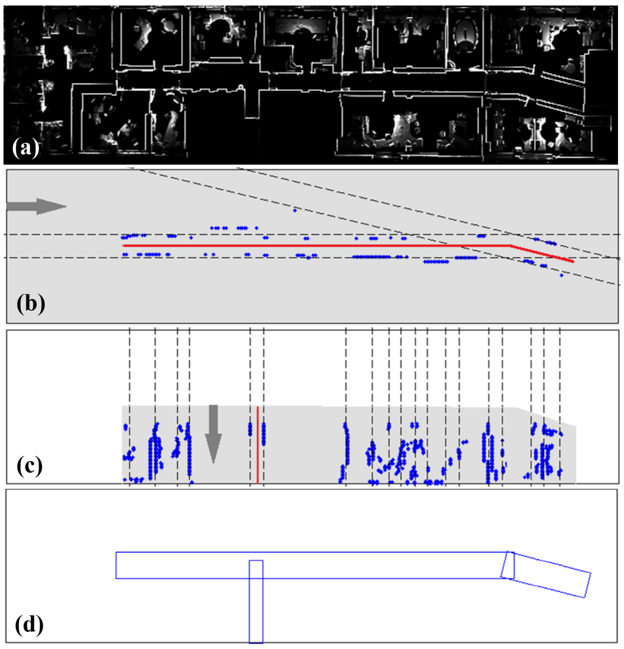

Figure 2. Derivation of 2D hallway polygons from an exemplary 3D point cloud.

Figure 2 illustrates the main steps of the algorithm based on an exemplary data set which represents a 3D point cloud gathered by laser scanning in the 4th floor of an office building. Figure 2a presents the 2D map derived from the 3D point cloud. Figure 2b and Figure 2c contain the result after the first and the second run, each time showing the respective hallway point candidates, the clustered hallway walls (black dashed), and the accepted centerline segments (red). The current search area is shadowed in gray, the search direction is indicated by an arrow. The resulting 2D hallway polygons are given in Figure 2d.

Beside the extraction of 2D hallway polygons based on which the layout parameters of the L-system can be set, 3D point clouds can also be used to detect stairs (see for example Schmittwilken & Plümer, 2010). Knowledge about the occurrence of staircases within a floor is required to set the control image of the L-system.

4.2 Instantiation of the Split Grammar

In order to derive an individual instance of the split grammar presented in section 3.1.2, split rules for following operations have to be set: (1) partitioning of the 3D building shell into floors; (2) partitioning of the floors into hallway spaces and non-hallway spaces; (3) partitioning of the non-hallway spaces into rooms.

Ad 1+2: The first two operations can be fully described by *SingleSplits*. To create instances of this kind of split rule, only the plane parameters of the partition planes, needed to create the respective splits, have to be determined: The horizontal partition planes for partitioning the 3D building shell into floors, can be automatically extracted from the LOD3 building model by searching within the facade plane for horizontal wall regions which are not intersected by windows (Becker, 2009). The parameters of the vertical partition planes for partitioning the floors into hallways and non-hallways are defined by the edges of the 2D hallway polygons as derived from observation data (see section 4.1) or generated by the L-system.

Ad 3: For the third operation, which establishes room configurations in non-hallway spaces, rule type *SingleSplit* is not sufficient. In order to model repetitive room sequences and complex non-sequential room configurations, also *RepeatSplits*, *StringSplits* and *MultiSplits* are required. The split rules are set based on the analysis of already existing wall representations in non-hallway spaces. The main steps of this analysis are:

- *Step 1*: Based on a given set of wall representations located within a non-hallway space, select all walls which are incident with the longest hallway wall. The selected walls represent a linear sequence of walls. Assign all selected walls a high priority; assign all non-selected walls a low priority.
- *Step 2*: For each selected wall: Determine the plane parameters and instantiate a rule of type *SingleSplit*.
- *Step 3*: Evaluate the distances between the walls within the linear sequence of "high priority"-walls and search for repetitive sub-sequences of those distances (i.e. room widths). For each repetitive sub-sequence: Instantiate a rule of type *RepeatSplit* or *StringSplit*.
- *Step 4*: For each pair of consecutive walls within the linear sequence of "high priority"-walls: Collect all "low priority"-walls lying in between. The configuration of the collected walls may be non-linear. Transfer this configuration into the grammar by instantiating a rule of type *MultiSplit*.
- *Step 5*: Repeat steps 1 to 4 until all non-hallway spaces are processed.
- *Step 6*: Determine the number of occurrences of each rule to derive the rule probabilities as described in section 3.1.2.

For the case that no wall representations are available based on which the split rules for establishing room configurations could be set, basic rules of type *SingleSplit* can also be derived from the window structures inherent in the LOD3 building model. For example, following the two assumptions that (1) split walls of high priority are perpendicular to the façade plane (defines the orientation of the split plane), and (2) the smallest room width corresponds to a value a bit higher than the smallest window width (defines the distance value of the split plane), a simple *SingleSplit* can be instantiated as a first guess (see also section 5).

5. GRAMMAR-SUPPORTED TRANSITION FROM LOD3 TO LOD4

The 3D indoor grammar (introduced in section 3) and the concepts to derive individual grammar instances from observation data (presented in section 4) can be used to provide a seamless transition from LOD3 to LOD4 building models. In the following, we present a grammar-based approach for the automatic refinement of an existing LOD3 model, containing 3D façade structures like windows and doors, to a LOD4 model which additionally represents the floor planes and the interior walls of the building. As input data, the approach requires a LOD3 model of the building to be reconstructed, and a 3D point cloud gathered in the interior of the building. The approach is robust as it can cope with raw, unfiltered, and even incomplete point clouds. This is due to the fact that our approach is based on an iterative automatic learning and verification process. The general idea of this iterative process will be described in section 5.1. Main steps and results are illustrated based on a real-world example in section 5.2

5.1 Grammar-based Reconstruction embedded in an Iterative Learning and Verification Process

Starting from an initial low-level grammar automatically derived from the window representations of the LOD3 building model, hypotheses about interior walls can be generated. The hypothesized walls are checked against the available 3D point cloud. The verified geometries form the basis for an automatic update of the initial grammar. By this, the knowledge content of the initial grammar is enriched, leading to a grammar with increased quality. This higher-level grammar can then be applied to predict realistic geometries to building parts where only sparse or even no observation data are available. Hypotheses about room configurations which cannot be checked against observation data are marked as *geometry with low reliability*. Our approach allows for the robust generation of complete 3D indoor models whose quality can be improved continuously as soon as new observation data - gathered in building parts with geometries of low reliability - is fed into the grammar-based reconstruction process.

5.2 Application to Real-World Data

Following data are available for testing the iterative grammar-based reconstruction approach presented in section 5.1:

1) 3D point cloud from laser scanning (Leica HDS3000) gathered in the 4th floor of an office building that comprises in total seven floors. The point cloud covers all rooms and hallways except for the staircase and the lavatories.
2) LOD3 model of the building. The window structures have been inserted manually into a LOD2 model provided by the city surveying office. (Becker (2009) showed, how window structures like these can be reconstructed automatically from mobile mapping data.) The LOD3 building model and the point cloud are given in the same local coordinate system.

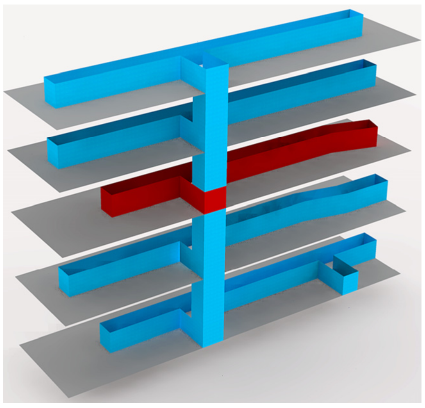
Figure 3. Network of hallways and staircases produced by the initial L-system.

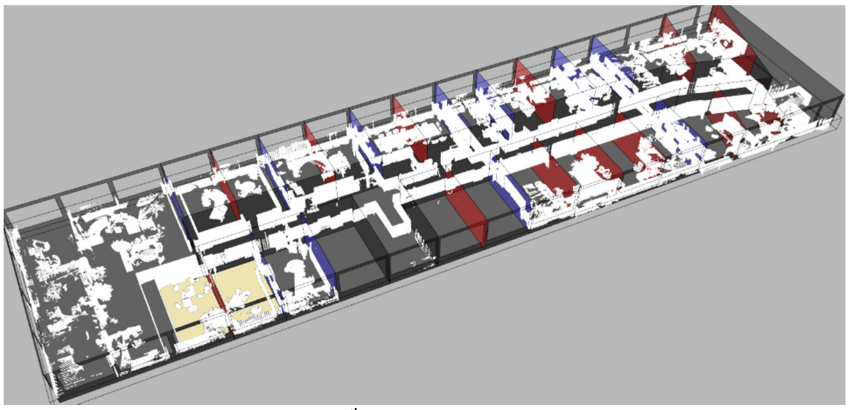
Figure 4. 3D indoor model for the 4th floor (here overlaid with the 3D point cloud) as obtained from applying the initial split grammar.

The approach begins with the instantiation of an initial low-level grammar (see section 0) which can then be applied to start the iterative reconstruction process (see section 5.2.2).

5.2.1 Initialization of an Individual Grammar Instance

The required initial low-level grammar consists of an initial L-system and an initial split grammar. Here, the initial L-system has been derived from the given point cloud as described in section 4.1 based on the same data set. Since this point cloud does not contain any information about the location of stairways, the control image of the L-system, has to be the result of manually selecting a hallway segment and interpreting it as "area containing a staircase". The hallway selected corresponds to the red centerline illustrated in Figure 2c. The initial split grammar has been derived by analyzing the window structures of the LOD3 building model as proposed in section 4.2.

5.2.2 Generation of Hallways and Rooms

At first, the initial split grammar is applied to segment the body of the building into floor spaces. Then, the initial L-system is used to generate a complete hallway network which traverses all floors of the building. The axiom on which the L-system is applied is defined by the centerlines of the 2D hallway polygons extracted from the point cloud in the 4th floor (see Figure 2d). Figure 3 shows how the L-system procedurally expands the initial hallway segments - representing the L-system's axiom (red shaded polygons) - in three dimensions leading to a network of vertical staircases and horizontal hallways. While the network structure in the given data set is rather simple, a more complicated example can be found in our previous work (Philipp et al., 2014). The 2D hallway system modelled there contains several loops and has a branching factor of three.

Based on the hallway network as illustrated in Figure 3, the floors are segmented into hallway- and non-hallway spaces. Afterwards, the initial split grammar is applied to segment the non-hallway spaces of the 4th floor into room configurations. The positions of the proposed split walls are illustrated in Figure 4 as red and blue planes. These wall hypotheses are tested against the point cloud. The split walls which are accepted are the blue ones. From Figure 4 it can be seen that out of all rooms represented in the point cloud only one room (yellow shaded) could not be detected. The left and the right end of the room are bounded by two walls perpendicular to the façade plane. While the right wall is represented in the model as blue split plane, the left one could not be found. The reason is that this wall is not incident with any of the hallway walls and, thus, is interpreted as split wall of low priority (see step 1 of the instantiation process described in section 4.2). Such a split can be reconstructed by means of a split rule of type *MultiSplit*, whose implementation is part of our current work. The geometric accuracy of the reconstructed interior walls directly correlates with the resolution of the 2D map and the thresholds chosen for the extraction of the hallway polygons (see steps 0 and 6 of the instantiation process explained in section 4.1). Since our focus is on a robust detection of floors rather than a high-precision reconstruction, we start with relatively coarse thresholds leading to accuracies for the wall positions in the order of several decimeters. However, accuracy enhancements can be obtained when fitting the detected rooms into the point cloud during an additional adjustment process which will be addressed in our future work.

Continuing based on the 3D reconstruction obtained by applying the *initial low-level grammar* (Figure 4), in a next iteration step, the accepted split walls are used for an update of the initial split grammar by means of the process as described in section 4.2. The resulting *higher-level grammar* is then applied to generate reliable hypotheses about the room configurations for the remaining floors where except for the knowledge about the window positions no observation data are available. The result is exemplarily shown for the floors 2 to 6 in Figure 5. The hypothesized geometries provide a basis which can be tested against arbitrary observation data as soon as available. Thus, the reconstruction in those areas does not need to start from scratch but can be efficiently guided by already existing indoor geometries within a "hypothesis and testing" procedure. For example, Philipp et al. (2014) showed how existing wall representations can be verified against traces collected with foot-mounted MEMS/IMUs. By this approach, even doors can be recognized and integrated into the walls.

6. CONCLUSIONS

We proposed a grammar-based approach for the automatic reconstruction of 3D indoor models for buildings from raw point clouds for "as-built" BIM. The grammar allows for the modeling of buildings whose horizontal, continuous floors are traversed by a hallway system which provides access to the rooms. Landmarks like castles and churches, or small residential houses where this assumption is not given are not supported. Due to the knowledge inherent in the grammar, the approach is *robust* providing realistic 3D geometries even in building areas where observation data are noisy or incomplete. The approach is additionally *flexible*: (1) Since geometric and semantic knowledge about hallway- and room-configurations are automatically derived from observation data or already existing building geometries, the approach automatically adapts to the architectural characteristics of various buildings. (2) Data of different sensors can be used to be fed into the grammar-based reconstruction. For

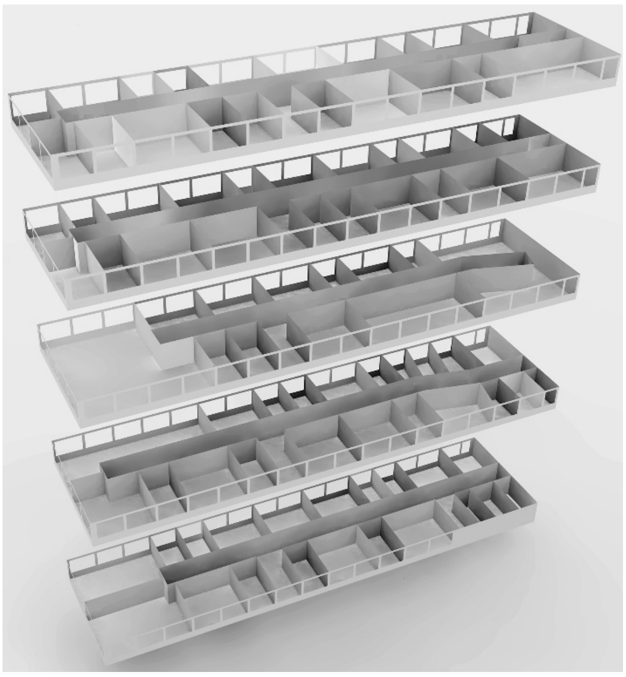
Figure 5. Grammar-based reconstruction for the floors 2 to 6.

example, the layout of hallways can be learned not only from 3D point clouds as described in this paper but also from traces stemming from foot-mounted MEMS/IMU systems as proven in previous work (Philipp et al., 2014). (3) During the iterative process of grammar application and grammar update, the quality of the reconstructed LOD4 model increases continuously.

Our approach provides a means to significantly support as-built BIM creation. The grammar-generated 3D indoor models show potential to be used in many application areas. Moreover, as the robustness of our approach conforms to the rapid development and fast increasing availability of sensors to derive 3D point clouds for everybody, it is not only of interest for experts working in the BIM field but also for the crowd.

ACKNOWLEDGEMENTS

This work is supported by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG), grants FR 823/25-1 and RO 1086/17-1, and the European Social Fund (ESF) as well as the Ministry Of Science, Research and the Arts Baden-Württemberg.

REFERENCES

Adan, A. and Huber, D., 2011. 3D Reconstruction of interior wall surfaces under occlusion and clutter. In *Proc. of 2011 International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, pp. 275-281.

Aliaga, D., Rosen, P. and Bekins, D., 2007. Style grammars for interactive visualization of architecture. *IEEE Transactions on Visualization and Computer Graphics*, 13(4), pp. 786-797.

Becker, S., 2009. Generation and application of rules for quality dependent façade reconstruction. *ISPRS J. Photogrammetry and Remote Sensing*, 64, pp. 640-653.

Bokeloh, M., Wand, M. and Seidel, H.-P., 2010. A connection between partial symmetry and inverse procedural modeling. *ACM Trans. Graph. (TOG)*, 29(4), pp. 104,1-104,10.

Budroni, A., 2013. *Automatic model reconstruction of Manhattan-world scenes from dense laser range data*. PhD Thesis, Deutsche Geodätische Kommission, series C, no. 715, 104 p.

Van Gool, L., Martinovic, A. and Mathias, M., 2013. Towards semantic city models. In D. Fritsch, ed. *Proc. Photogrammetric Week'13*. Wichmann, Berlin/Offenbach, pp. 217-232.

Gröger, G. and Plümer, L., 2010. Derivation of 3D indoor models by grammars for route planning. *Photogrammetrie-Fernerkundung-Geoinformation*, 2010(3), pp. 193-210.

Haala, N., Fritsch, D., Peter, M. and Khosravani, A, 2011. Pedestrian mobile mapping system for indoor environments based on MEMS IMU and range camera. *Archives of Photogrammetry, Cartogr. and Remote Sensing*, 22, pp. 159-172.

Hichri, N., Stefani, C., De Luca, L., and Veron, P., 2013. Review of the "as-built BIM" approaches. In *Int. Arch. Photogr. Remote Sens. Spatial Inf. Sci.*, XL-5/W1, pp. 107-112.

Jenke, P., Huhle, B. and Straßer, W., 2009. Statistical reconstruction of indoor scenes. In *Proc. of 17. Int. Conf. in Central Europe on Comp. Graphics, Vis. and Comp. Vision (WSCG '09)*, 8 p.

Kada, M., 2007. Scale-dependent simplification of 3D building models based on cell decomposition and primitive instancing. In *Proc. Int. Conf. Spatial Inform. Theory*, COSIT '07, pp. 222-237.

Khoshelham, K. and Díaz-Vilariño, L., 2014. 3D modelling of interior spaces: Learning the language of indoor architecture. In *Int. Arch. Photogr. Remote Sens. Spatial Inf. Sci.*, XL-5, pp. 321-326.

Marson, F. and Musse, S.R., 2010. Automatic real-time generation of floor plans based on squarified treemaps algorithm. *Int. Journal of Computer Games Technology*, 2010, pp. 7,1-7,10.

Matsuyama, T. and Hwang, V. S.-S., 1990. *SIGMA - A Knowledge-Based Aerial Image Understanding System*, Perseus Publishing.

Mirahmadi, M. and Shami, A., 2012. A novel algorithm for real-time procedural generation of building floor plans. *Computer Research Repository*, 7p.

Müller, P., Zeng, G., Wonka, P. and van Gool, L., 2007. Image-based procedural modeling of facades. *ACM Trans. Graph.*, 26(3), pp. 85,1-85,9.

Müller, P., Wonka, P., Haegler, S., Ulmer, A. and van Gool, L., 2006. Procedural modeling of buildings. *ACM Trans. Graph.*, 25(3), pp. 614-623.

Parish, Y.I.H. and Müller, P., 2001. Procedural modeling of cities. In *Proc. 28. Ann. Conf. Comp. Graph. and Interactive Techniques*. SIGGRAPH '01, pp. 301-308.

Peter, M., Khosravani, A.M. and Fritsch, D., 2013. Refinement of coarse indoor models using position traces and a low-cost range camera. In *Proc. of 4th Int. Conf. on Indoor Positioning and Indoor Navigation*, pp. 787-795.

Philipp, D., Baier, P., Dibak, C., Dürr, F., Rothermel, K., Becker, S., Peter, M. and Fritsch, D., 2014. MapGENIE: Grammar-enhanced indoor map construction from crowd-sourced data. In *Proc. 12. IEEE Int. Conf. on Pervasive Computing and Communications (PerCom 2014)*, pp. 1-9.

Prusinkiewicz, P. and Lindenmayer, A., 1990. *The algorithmic beauty of plants*, Springer New York.

Schmittwilken, J. and Plümer, L., 2010. Model-based reconstruction and classification of facade parts in 3D point clouds. In *Proc. of the Symp. on Photogrammetric Computer Vision and Image Analysis (PCV 2010)*, 6p.

Stilla, U. and Michaelsen, E., 1997. Semantic modelling of man-made objects by production nets. In A. Gruen et al., eds. *Automatic extraction of man-made objects from aerial and space images (II)*. Birkhäuser, Basel, pp. 43-52.

Tang, P., Huber, D., Akinci, B., Lipman, R. and Lytle, A., 2010. Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques. *Automation in Construction*, 19(7), pp. 829-843.

Wonka, P., Wimmer, M., Sillion, F. and Ribarsky, W., 2003. Instant architecture. In *ACM SIGGRAPH 2003 Papers*, pp. 669-677.