VOXEL-BASED INDOOR RECONSTRUCTION FROM HOLOLENS TRIANGLE MESHES

P. Hübner*, M. Weinmann, S. Wursthorn

Institute of Photogrammetry and Remote Sensing, Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
(patrick.huebner, martin.weinmann, sven.wursthorn)@kit.edu

Commission IV, WG IV/5

KEY WORDS: Indoor Reconstruction, HoloLens, Triangle Meshes, Voxels, 3D, Classification

ABSTRACT:

Current mobile augmented reality devices are often equipped with range sensors. The Microsoft HoloLens for instance is equipped with a Time-of-Flight (ToF) range camera providing coarse triangle meshes that can be used in custom applications. We suggest to use these triangle meshes for the automatic generation of indoor models that can serve as basis for augmenting their physical counterpart with location-dependent information. In this paper, we present a novel voxel-based approach for automated indoor reconstruction from unstructured three-dimensional geometries like triangle meshes. After an initial voxelisation of the input data, rooms are detected in the resulting voxel grid by segmenting connected voxel components of ceiling candidates and extruding them downwards to find floor candidates. Semantic class labels like 'Wall', 'Wall Opening', 'Interior Object' and 'Empty Interior' are then assigned to the room voxels in-between ceiling and floor by a rule-based voxel sweep algorithm. Finally, the geometry of the detected walls and their openings is refined in voxel representation. The proposed approach is not restricted to Manhattan World scenarios and does not rely on room surfaces being planar.

1. INTRODUCTION

Current head-worn Augmented Reality (AR) devices like the Microsoft HoloLens[1] hold great potential for enriching indoor environments with the in-situ visualisation of location-dependent information, e.g. from digital building information models (Hübner et al., 2018). While suchlike building models are currently gaining in prevalence with the increasing use of Building Information Modeling (BIM) techniques in the planning and construction stages of building projects (Jung and Lee, 2015), already existing, older buildings frequently lack such a digital representation that could be used in such indoor AR scenarios (Lu and Lee, 2017).

Mobile indoor AR devices are however often equipped with range sensors to facilitate 1) tracking and relocalisation within indoor environments and 2) convincing placement and interaction of virtual content with the physical surrounding. The HoloLens for instance is equipped with a Time-of-Flight (ToF) range camera providing range images and preprocessed triangle meshes that can be used in custom applications. The geometric accuracy of these meshes was found to be in the range of few centimetres, however drift effects on scales larger than single rooms can occur (Hübner et al., 2019, 2020; Khoshelham et al., 2019). We suggest to use these data for the automatic generation of digital models of indoor environments that can serve as basis for augmenting their physical counterpart with location-dependent informative content in an indoor AR setting.

While the automated reconstruction of indoor building models from unstructured three-dimensional geometries is an active field of research (Ma and Liu, 2018), prevalent approaches use dense point clouds acquired by LiDAR sensors or range cameras. To the best of our knowledge, there is currently no attempt on indoor reconstruction using sparse triangle meshes as provided by the HoloLens system.

In this paper, we present a novel indoor reconstruction approach for automatically deriving indoor building models as voxel representation from sparse triangle meshes. The proposed approach is not restricted to Manhattan World scenarios and does not rely on room surfaces being planar. It can furthermore be easily transferred to using input data in the form of dense point clouds in the scope of future work.

After briefly summarising related work in Section 2, we describe our voxel-based method for the reconstruction of indoor environments from sparse triangle meshes and the applied evaluation procedure in Section 3. After presenting evaluation results in Section 4 and discussing them in Section 5, we provide concluding remarks and suggestions for further research in Section 6.

2. RELATED WORK

Recently, a range of notable work on indoor reconstruction was published which is briefly summarised in this section. Tran and Khoshelham (2019) for instance proposed to detect rooms in LiDAR point clouds via a space subdivision approach by RANSAC planes, first horizontally for separating storeys and then vertically per storey. In each storey, the resulting cell complex generated by the intersecting planes is arranged to the most probable composition of rooms by a stochastic approach, where the likelihood of a cell being part of a room is determined by the amount of points constituting its surface. Points obstructing room interior space are penalised.

Yang et al. (2019) proposed to detect rooms in a cell complex of intersecting lines by an energy minimisation approach. Wall surfaces constituting the cell complex are derived in this case by detecting and refining closed contours in 2D sections shortly below ceiling height, where the amount of furniture can be expected to be minimal. This approach is capable of reconstructing rooms with horizontally curved walls.

Ochmann et al. (2019) focused on indoor reconstruction relying on RANSAC for plane detection. Rooms are segmented from

*Corresponding author
[1] https://www.microsoft.com/en-us/hololens

the detected planes by Markov clustering based on the mutual visibility of plane patches. The detected room surface planes are subsequently intersected in 3D space to reconstruct volumetric wall and slab objects via an integer linear programming approach.

Nikoohemat et al. (2018) presented an approach to reconstruct indoor models from point clouds captured by mobile LiDAR-based indoor mapping systems while also incorporating trajectory information e.g. for separating storeys and stairwells. Planar surface patches are detected and successively merged depending on coplanarity and distance. The resulting plane patches are arranged in an adjacency graph and subjected to a rule-based process to reconstruct and refine room surface geometry. Wall openings are distinguished from occlusion-caused absence of points by a voxel-based ray-tracing approach emanating from the trajectory and by checking if the trajectory itself passes through an opening. Rooms are discerned by a voxel-based space partitioning, again incorporating trajectory information.

Xie et al. (2019) also focused on detecting planar patches and subsequently refining them by merging. Then vertical patches are selected based on their normal direction as wall surfaces and contour lines are extracted in 2D sections in a similar way as proposed by Yang et al. (2019). Here, however, multiple contours are extracted over the whole height range of the room to account for vertical changes in wall geometry like protrusions or recesses. The refined contours are then assembled to 3D room surfaces.

Díaz-Vilariño et al. (2019) presented an indoor reconstruction approach, that also relies on region growing for extracting planar surfaces. The planes are subsequently classified in obstacles and room surfaces, with the latter being further refined by intersecting the individual planes with each other. For each wall surface, rectangular openings are detected by applying the Generalized Hough Transformation on 2D raster representations of the walls. The reconstructed indoor models are further used in an indoor path-finding framework.

Like Nikoohemat et al. (2018), Flikweert et al. (2019) also use mobile LiDAR point clouds while also taking the trajectory into account as additional source of information. Wall surfaces are detected in the point cloud by applying the Medial Axis Transform. Doors are then detected in voxelized walls with regard to the voxelized trajectory. Rooms are finally segmented by region growing among floor voxels while stopping on floor voxels within the detected doors.

Gorte et al. (2019) also use a 3D voxel representation of the input data for floor detection. They however do not reconstruct complete indoor models but focus on detecting navigable floor space for indoor path-finding to building exits. Floor voxels are detected as non-empty voxels with a certain amount of vertical and horizontal free space above them. Region growing with a threshold on the maximum height difference between neighbouring floor voxels generates connected components of navigable floor voxels extending over stairs. Here, indoor voxel models prove to be a suitable form of data representation for distance calculation for navigation tasks.

3. METHODOLOGY

In this section, we present our novel method for reconstructing voxel models of indoor environments from unstructured 3D data with normal directions. After stating necessary data preparation steps, we provide an in-depth explanation of our proposed reconstruction algorithm. Subsequently, we elaborate on our evaluation procedure.

The proposed approach is summarised in Figure 1 with Table 1 detailing the respective voxel colours. 3D data of indoor environments are transformed to a voxel representation and subsequently a model of the respective indoor environment is reconstructed in voxel space. To this aim, voxels are assigned to rooms and semantic classes. While the method is applicable for unstructured 3D data with absolute normals and thus could also be applied to point clouds provided they have normals, we focus in this study on triangle meshes captured with the Microsoft HoloLens.

3.1 Data Preparation

While our proposed method is not restricted to Manhattan World scenarios, where all room surfaces have to be aligned with the coordinate axes, the results are cleaner and visually more appealing when planar wall surfaces are aligned with the coordinate axes. Thus, it can make sense to align the model along the coordinate axes as a preprocessing step. This is, however, optional and not necessary for the reconstruction method presented here to work. Furthermore, our method does not rely on wall surfaces to be planar but can also deal with curved wall surfaces.

For now, however, we presuppose, that walls are vertical. Vertically slanted walls with an inclination of more than $45°$ are valid though. These are reconstructed as ceiling surfaces. We furthermore assume that the upward direction is known and corresponds to one of the coordinate axes.

As a preliminary processing step, the input dataset is converted to a voxel representation. To this aim, a certain range of space encompassing the input data is subdivided by a three-dimensional grid of cubical, non-overlapping cells of uniform size, i.e. voxels. Each voxel intersecting a geometric primitive of the input data (i.e. a point of a point cloud or a triangle of a triangle mesh as is the case here) is marked as non-empty. These non-empty voxels are classified according to the normal directions of the intersecting geometric primitives. If the majority of the normal vectors of intersecting primitives points upwards or downwards within a range of $\pm 45°$, the voxel is classified as 'Normal Up' or 'Normal Down', respectively. Otherwise, non-empty voxels are classified as 'Normal Horizontal'.

Voxels can thus be characterised as having one of four possible states: 'Empty', 'Normal Up', 'Normal Down' or 'Normal Horizontal'. Hence, the voxel representation can be stored memory-efficiently as a three-dimensional array of byte values. The voxel size as parameter of the voxelisation process is set to $5\,\mathrm{cm}$ in the scope of this paper, a reasonable value in consideration of the typical dimensions of indoor environments and the resolution of the HoloLens triangle meshes. The same voxel size has been used e.g. by Gorte et al. (2019) for the automated extraction of navigable space in indoor environments.

The resulting voxel grid serves as input for the reconstruction algorithm presented in the following sections. The aim of the proposed algorithm is to segment this voxel grid in rooms and to classify voxels belonging to a room as 'Ceiling', 'Floor', 'Wall', 'Wall Opening', 'Empty Interior' or 'Interior Object'.

3.2 Room Detection

In a first processing step, rooms are detected in the voxel model by segmenting ceiling candidates. In subsequent refinement steps, ceilings and floors are reconstructed as voxel representation for each room.

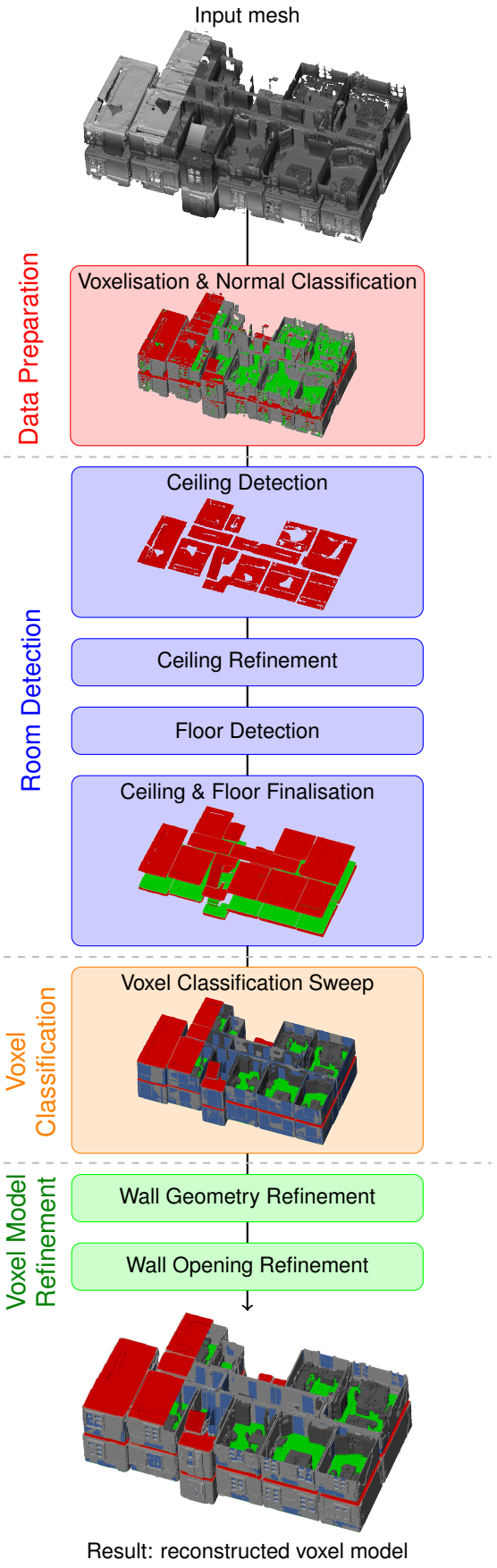


Figure 1. Processing workflow from input mesh to final reconstructed voxel model. The colour scheme of the normal classification is described in Table 1(a). The other depictions are coloured according to Table 1(b). A part of the ceiling is removed for the sake of visibility.

Table 1. Colour schemes for voxel classes.

a) Normal Direction

Colour	Class Label
	Normal Down
	Normal Up
	Normal Horizontal

b) Indoor Reconstruction

Colour	Class Label
	Ceiling
	Floor
	Wall
	Wall Opening
	Interior Object

3.2.1 Ceiling Detection Initially, ceiling segments are detected as seeds for room candidates. This is achieved by segmenting 'Normal Down' voxels to connected components based on a 3D-26-neighbourhood via 3D region growing. Beforehand, a rule-based voxel sweep is performed by iterating section-wise from bottom to top through the voxel grid along the upward direction and switching 'Normal Down' voxels that have a 'Normal Horizontal' voxel directly below them to 'Normal Horizontal' themselves. This ensures that the ceiling segments resulting from the region growing algorithm represent distinct room candidates divided by walls.

Ceiling segments are discarded as room candidates if they have a horizontal coverage of less than $0.5\,\mathrm{m}^2$. In our algorithm, parameters like this one are generally set as metric values in order to be independent of the applied resolution of the voxel grid. The respective values are chosen in consideration of typical dimensions of indoor environments.

3.2.2 Ceiling Refinement The remaining ceiling candidates are subsequently refined. In this refinement process, horizontal holes in the voxel segments representing ceilings are detected. These holes are eventually closed later on if they satisfy certain criteria trying to ensure that only holes are being closed, that are caused by incomplete acquisition of the geometric primitives the voxel representation is derived from, e.g. caused by occlusion. Actual holes in ceiling surfaces caused e.g. by columns or corners pointing convexly inside the room, on the other hand, should not be closed.

For the detection of holes in ceiling segments, the ceiling segments are transformed to a horizontal two-dimensional pixel grid covering the whole horizontal extension of the respective ceiling segment. Pixels are marked as non-empty if a voxel of the ceiling segment occupies its position. The ceiling pixel is assigned the height in voxel grid integer coordinates of the lowest ceiling segment voxel occupying its position.

Hole pixels are then detected within this two-dimensional grid by searching for empty pixels that are in-between two non-empty pixels along the four main directions resulting from a 2D-8-neighbourhood. Hole pixels need not be directly neighbouring their enclosing non-empty pixels, i.e. holes can have an extent of multiple pixels. The detected hole pixels are subsequently segmented to hole segments based on a 2D-4-neighbourhood.

Furthermore, each hole pixel is assigned a height value in integer voxel grid coordinates. This height value is determined by 2D ray-tracing from ceiling pixels that are positioned in a 2D-4-neighbourhood of hole pixels across the hole in the four main directions resulting from a 2D-8-neighbourhood. If a 2D ray ends in another ceiling pixel, the height of hole pixels along the ray is interpolated linearly between the height values of the two ceiling pixels. Otherwise, the height of hole pixels is set directly from the ceiling pixel the ray started from. The height value of each hole pixel is kept as a moving average over all rays and subsequently rounded to an integer value. Finally, a smoothing of the height

values is applied across all hole pixels belonging to a common hole segment.

3.2.3 Floor Detection Once all hole pixels have been assigned a height value, it is decided per hole segment if the hole can be closed, i.e. if the voxels corresponding to hole pixels and their respective height values should be included as ceiling voxels in the respective ceiling segment. To make this decision, knowledge about the supposed vertical extent of the room under the respective ceiling voxel is necessary. Hence, for each voxel corresponding to a ceiling or hole pixel and its respective height, a corresponding floor voxel candidate needs to be found.

To this aim, another two-dimensional pixel grid of the same extent as the respective ceiling pixel grid is created for each ceiling voxel segment. The voxel grid is traced through downwards from each ceiling or hole voxel position. If a 'Normal Up' voxel is encountered before reaching the bottom of the voxel grid or a 'Normal Down' voxel belonging to another ceiling segment, the respective floor pixel grid position is marked as floor candidate and assigned the height value of the 'Normal Up' voxel.

These floor candidate pixels are subsequently segmented in floor candidate pixel segments based on a 2.5D-8-neighbourhood per 2D floor grid. The 2.5D-8-neighbourhood is realised by 2D region growing with a threshold of $18\,\mathrm{cm}$ (a common height of stair steps) as maximally allowed height difference between neighbouring floor candidate pixels.

Floor candidate segments with a horizontal extent of less than $0.5\,\mathrm{m}^2$ are discarded, unless there are no floor candidate segments with a horizontal extent larger than this in the respective floor pixel grid. From the remaining floor candidate segments, the segment containing the lowest height is selected as final floor for the respective room. All pixels belonging to other segments are set to 'Empty'.

The selected value of $18\,\mathrm{cm}$ as maximum height difference on floors proved to result in floor segments that can extend over stairs and ramps while mostly avoiding to spread over to surfaces on top of furniture like tables or chairs.

For each floor pixel grid, all empty pixels corresponding to a ceiling pixel or hole pixel in the respective corresponding ceiling pixel grid need to have a height value assigned. Their height is determined via the same procedure as used for the height of the hole pixels in the ceiling pixel grids as detailed beforehand.

3.2.4 Ceiling and Floor Finalisation On the basis of the floor reconstruction, the decision which ceiling holes are to be closed can now be made by analysing the voxels below hole voxels in the height range restricted by the height values in the respective floor grid pixels. If a horizontal hole extent overlaps with another already reconstructed room within this height range, it cannot be closed. If this is not the case and at least $75\,\%$ of the horizontal hole extent is occupied by a non-empty voxel in the input voxel grid within the height range between ceiling and floor, the hole is to be closed. Otherwise, the hole can potentially still be closed, unless the vertical extent of the border between hole and room is filled to at least $75\,\%$ in the input voxel grid, indicating the presence of a wall.

At this state, ceilings and floors of all detected rooms are determined as connected voxel components with each ceiling voxel having a corresponding floor voxel somewhere below it. Subsequently, a smoothing process is applied to the height of ceilings and floors per room. Furthermore, wall voxels are initialized along the border of ceilings as a closed contour of voxels neighbouring the ceiling segment on the outside.

3.3 Voxel Classification

During the complete reconstruction process detailed here, the state of the voxel grid is stored in a three-dimensional array of integer arrays. Under certain circumstances, our algorithm allows for voxels to belong to two different classes at once for the same room as well as to belong to multiple rooms at once with different classes. For instance, a voxel can be classified as 'Ceiling' as well as as 'Wall' at once for the same room to represent the fact, that the edge between the actual ceiling and wall surfaces runs through this voxel. This is the case for the wall voxels initialized along the contours of ceilings. Similarly, voxels can belong simultaneously to the classes 'Floor' and 'Wall'.

Furthermore, voxels can belong to more than one room. This is the case, if the surfaces of neighbouring rooms are covered by the same voxel. The frequency of this occurrence depends on the voxel size as well as on the width of walls and horizontal slabs of the buildings represented by the data. In those cases where a voxel belongs to more than one room, its respective classes can only occur in certain specific combinations. While a voxel can e.g. represent the walls of two neighbouring rooms or ceiling for one room and floor for another directly above it, it for example cannot represent floor for more than one room at once. Also voxels belonging to the room interior (classes 'Empty Interior' and 'Interior Object') can only belong to one room, as rooms can only share voxels along their borders.

For classifying voxels besides the already reconstructed floors and ceilings, the voxel grid is traversed section-wise from top to bottom, while voxels are assigned to rooms and classes depending on the state of the voxel above it. In doing so, all voxels below a ceiling voxel get assigned to the same room the ceiling voxel belongs to, until a floor voxel of the respective room is encountered. When the ceiling voxel does not also have the class label 'Wall' for the same room, the voxels between it and the floor will be assigned to the classes 'Empty Interior' or 'Interior Object', depending on the respective voxel being empty or not in the input data. Voxels below ceiling voxels that are simultaneously wall voxels of the same room, on the other hand, are classified as 'Wall' or 'Wall Opening' based on the same criterion. When walls finally hit their corresponding floor voxel, that voxel gets assigned the class label 'Wall' additionally to it being classified as 'Floor'.

3.4 Voxel Model Refinement

After applying the voxel sweep algorithm described previously, voxels have assigned class labels and room affiliations. The resulting indoor voxel model is further improved in subsequent refinement sweeps.

Preliminarily, horizontal two-dimensional normal directions in integer voxel grid coordinates are determined pointing towards the room interior for all wall voxels (class 'Wall' or 'Wall Opening') based on a 2D-4-neighbourhood. As is the case with class labels, these normal directions are managed on a per-room basis. Thus, a voxel can have two sets of normal directions if it is a wall voxel neighbouring two rooms.

3.4.1 Wall Geometry Refinement In a first refinement pass, missing wall sections are detected and completed by searching for voxels belonging to the room interior (classes 'Empty Interior' and 'Interior Object') that are horizontally neighbouring voxels that do not belong to the same room based on a 2D-4-neighbourhood. These neighbouring voxels not belonging to the same room can either belong to other rooms or no room at all.

Such situations can occur on discontinuities in height of horizontally neighbouring ceiling or floor voxels. This can occur e.g. on

vertical windows in slanted ceilings or on the floor where large discontinuities in height are possible despite of the threshold on height difference of neighbouring voxels in the region growing of floor segments. This can happen, when parts of the floor on different height levels are connected by stairs or ramps (e.g. in the case of staircases or platforms like stages). In these cases, interior voxels are converted to wall voxels (class 'Wall' or 'Wall Opening' depending on the voxel being empty in the input data) or to ceiling or respectively floor depending on the height of the discontinuity being larger or smaller than the aforementioned threshold of $18\,\mathrm{cm}$.

So far, all reconstructed walls have a thickness of one voxel. This results from the fact, that they have been initialised as a contour with the width of one voxel along the border of the ceilings and were then just traced vertically downwards. However, also vertical walls as presupposed by this algorithm can encompass elements with a certain extent perpendicular to the wall surface, e.g. applications such as bordures, light switches, window ledges or window recesses that are normally considered as part of the wall and not as individual furniture objects (which would belong to the class 'Interior Object'). To include suchlike elements in our voxel reconstruction of walls, a wall refinement procedure is applied for each pre-existing wall voxel (class 'Wall' or 'Wall Opening'). New wall voxels arising in the course of this procedure are not recursively processed by it.

First, for each wall voxel, we sweep through the neighbouring voxels along the reversed normal directions of the voxel, i.e. going away from the room to the outside. Here, non-empty voxels that do not yet belong to a room are searched within a distance of up to $15\,\mathrm{cm}$. If there are any non-empty voxels within this search distance, they are added to the wall of the respective room and the normal directions of the initial voxel are copied. If there are also empty voxels in-between, these are also added with class label 'Wall Opening'.

Next, we apply a similar procedure going from the initial voxel along its normal directions up to $15\,\mathrm{cm}$ towards the inside of the room. Here, however, we immediately stop if we encounter a voxel labeled as 'Empty Interior'. Only a continuous succession of 'Interior Object' voxels can potentially be added to the wall. Before doing so, however, we check if this continuous succession of voxels goes further than the search distance of $15\,\mathrm{cm}$. If this is the case, we assume that they belong to a large furniture object like e.g. a table and do not add them to the wall.

3.4.2 Wall Opening Refinement In a next step, we refine the occurrence of wall opening labels. This again is done by traversing the voxel grid along the normal directions towards the room interior and away from it for all pre-existing wall voxels (i.e. all wall voxels that already existed before the last refinement step). Thus, for each pre-existing wall voxel, we get stacks of wall voxels along the normal directions. If any of these voxels has the label 'Wall', we set all 'Wall Opening' voxels in the stack to 'Wall'.

Furthermore, for all voxels that are still labeled as 'Wall Opening' after applying this refinement step, we again traverse the voxel grid along the normal directions going towards the room interior with a search distance of $70\,\mathrm{cm}$ to check if there are surfaces of large furniture objects occluding this part of the wall. If this is the case, the whole wall voxel stack is set to 'Wall' as well.

3.5 Evaluation

As the indoor reconstruction method proposed in this work does not process the geometric primitives comprising the input datasets themselves but voxel representations derived from them, ground truth data for evaluation purposes should in the end also be given as voxel representation. Thus, a possible approach would be to manually label voxel representations of input data with ground truth information. However, to be able to automatically evaluate the influence of the voxel size in the scope of future work, we chose not to label our ground truth data in voxel space but to construct ground truth data in the metric space of the input data and pass it through the voxelisation step as well.

As our aim is to not only semantically classify the geometric primitives comprising the input data, but also to geometrically reconstruct indoor environments in the voxel representation, the generation of ground truth data must exceed simple labelling of the geometric primitives. The input data in the form of triangle meshes may contain gaps caused by occlusion or incomplete mapping of the represented indoor environments. The reconstruction algorithm, however, is expected to fill these gaps by labelling the voxels there as room surface. Thus, suchlike gaps must be manually closed by geometries in the ground truth data.

We hence approached ground truth data generation by manually cutting the triangle meshes comprising the input data in different parts and labelling them. First, the meshes are manually split in rooms and then on per-room basis split in semantic classes as 'Floor', 'Ceiling' and 'Wall'. The resulting meshes are then completed by manually constructing planes to close gaps in the data, e.g. where walls are occluded by furniture. Furthermore, openings in the meshes that should explicitly be detected as openings by the reconstruction algorithm such as window openings or door openings are also manually constructed as planes.

The resulting labelled ground truth meshes are then voxelized as described in Section 3.1, while their labels are passed on to the resulting voxels. For evaluation purposes, test data as well as ground truth data are both voxelized by the same grid, i.e. for determining the grid extensions, we use a bounding box including both datasets. Thus, we ensure that voxel coordinates are directly comparable.

Some rules are applied to handle situations, where ground truth meshes with conflicting labels are intersecting the same voxel. For instance, if a voxel intersects 'Wall' meshes as well as 'Wall Opening' meshes, it should be labelled as 'Wall'. Furthermore, the ground truth voxel model should use the same data representation as the one created by the algorithm to be evaluated, i.e. voxels can belong to multiple classes and multiple rooms.

As the manual labelling of furniture as 'Interior Object' and the construction of volumetric geometries representing 'Empty Interior' would be too time-consuming, these classes were excluded from the evaluation. Voxels belonging to these classes are for evaluation purposes treated as empty voxels not belonging to any room, as they are represented in the ground truth data.

As the labelling of room numbers may differ between test data and ground truth data and, furthermore, the segmentation of rooms may differ between the datasets, a mapping between rooms of both datasets has to be determined. On the one hand, this serves to evaluate the room segmentation of the proposed algorithm and, on the other hand, this is used for evaluating the voxel classification.

To this end, a weighted mapping between rooms from both datasets is derived by analysing voxels, that in both datasets are classified as 'Ceiling'. If such a voxel is assigned in the ground truth dataset as belonging to room X and in the test data as belonging to room Y, the weight counter of room mapping $X \rightarrow Y$ is incremented. Room mappings with negligible weight can be discarded.

Table 2. Dataset characteristics.

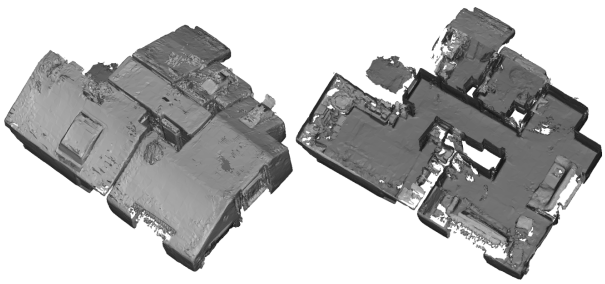
Dataset	Office	Attic
Spatial Extent [m]	13×21×8	8×9×3
Time for Acquisition [min]	29	16
Mesh Vertices [$\cdot 10^6$]	2.87	0.08

In the evaluation procedure, we currently only check for corresponding voxels to hold exactly similar states in due consideration of the above-mentioned room mapping and the treatment of room interior voxels as not belonging to a room. This means that we do currently not consider cases, where a voxel has the correct state for one room but an incorrect one for another room, as partly correct.

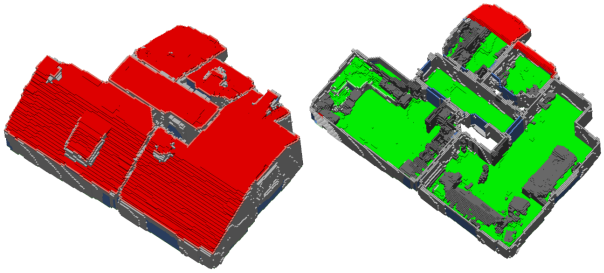
4. RESULTS

In the evaluation process, we use two datasets. Both triangle meshes were captured with a Microsoft HoloLens (version 1). The first dataset ('Office') represents an indoor office environment with multiple rooms on two storeys including furniture. The input mesh as well as the reconstructed voxel model including voxels of the class 'Interior Object' are depicted in Figure 1. The second dataset ('Attic') represents an attic with slanted ceilings comprised of five rooms used as storage area. The triangle mesh and the resulting voxel reconstruction are depicted in Figure 2. Data characteristics such as spatial extent, resolution and time for acquisition are listed in Table 2.

For the 'Office' dataset, a ground truth voxel representation derived from manually labeled and constructed ground truth meshes is depicted in Figure 3(a), while Figure 3(b) depicts our reconstruction with voxels of the classes 'Interior Object' and 'Empty Interior' omitted. To demonstrate that our reconstruction algorithm is not restricted to Manhattan World scenarios, we also rotated the input mesh as well as the ground truth mesh by 30° around the up-axis. The resulting voxel models are depicted in Figures 3(c) and 3(d), respectively.

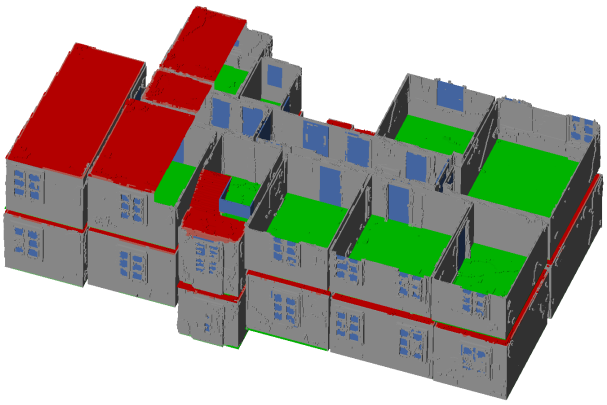


(a) Triangle mesh.

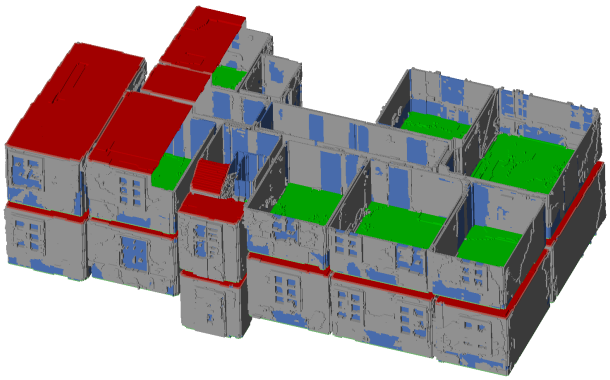


(b) Reconstructed voxel model, where voxels are colourised according to Table 1(b).

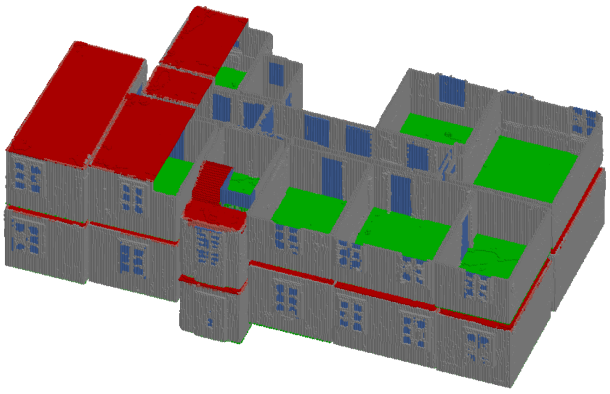
Figure 2. The dataset 'Attic'.



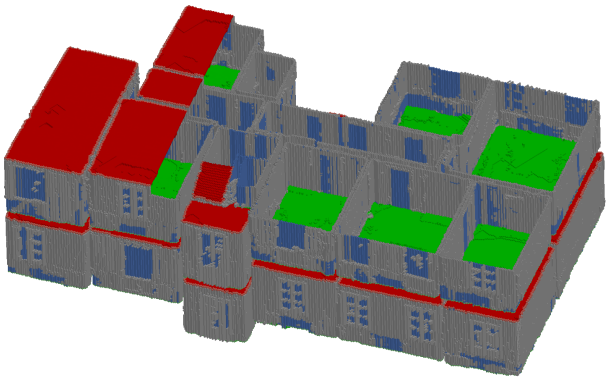
(a) Ground truth for the axes-aligned dataset.



(b) Reconstruction result for the axes-aligned dataset.



(c) Ground truth for the dataset rotated by 30° around the up-axis.



(d) Reconstruction result for the dataset rotated by 30° around the up-axis.

Figure 3. Voxelized ground truth data and reconstruction results for the 'Office' dataset. Voxels are colourised according to Table 1(b). In each case, a part of the ceiling is removed for the sake of visibility.

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-V-4-2020-79-2020 | © Authors 2020. CC BY 4.0 License.

Table 3. Evaluation results (Vx.: Voxels, NE: Non-Empty, GT: Ground Truth, RC: Reconstruction).

Dataset	Office	Office Rotated	Attic
Vx. [$\cdot 10^6$]	18.89	30.26	1.73
NE Vx. in RC [%]	5.24	3.23	6.8
Rooms GT	25	25	5
Rooms RC	30	29	5
Wrong Rooms from GT	1	1	0
Correct Vx. [%]	95.18	96.72	92.91
Correct Vx. in RC NE [%]	51.77	53.70	44.04
Correct Vx. in GT and RC NE [%]	91.66	91.82	84.53

Table 4. The meaning of the colours in Figure 4 (GT: Ground truth, RC: reconstruction)

Colour	Label
	Voxel classified correctly
	Voxel not empty in GT and RC, but wrong class
	Voxel empty in GT, but not in RC
	Voxel empty in RC, but not in GT

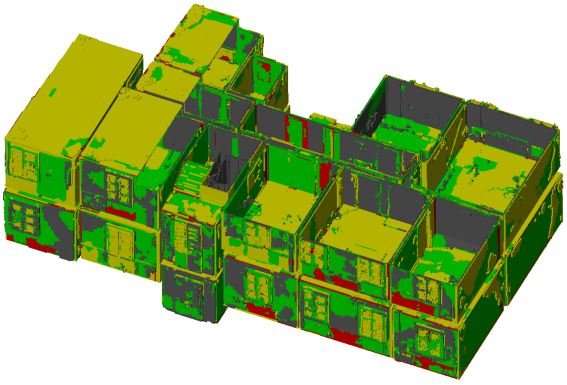
The evaluation results for these datasets are detailed in Table 3. In both cases of the 'Office' dataset (axes-aligned and rotated mesh), all rooms except for the stairwell were detected correctly. The stairwell being defined as one room in the ground truth model was in both cases split into multiple rooms by the reconstruction algorithm. In the case of the 'Attic' dataset, all five rooms were correctly detected.

For all datasets, the high overall ratio of correctly classified voxels is heavily biased by an enormous predominance of correctly classified empty voxels. The low ratio of correctly classified voxels in proportion to non-empty voxels in the reconstructed model, however, is also misleading as will be discussed in the following section. The ratio of correctly classified voxels in proportion to voxels that are not empty in the reconstruction as well as in the ground truth, on the other hand, is again over 90 % for the 'Office' dataset and around 85 % for the 'Attic' dataset. The results for the axes-aligned and the rotated version of the 'Office' dataset are quite comparable, both visually as well as concerning the evaluation results presented in Table 3.

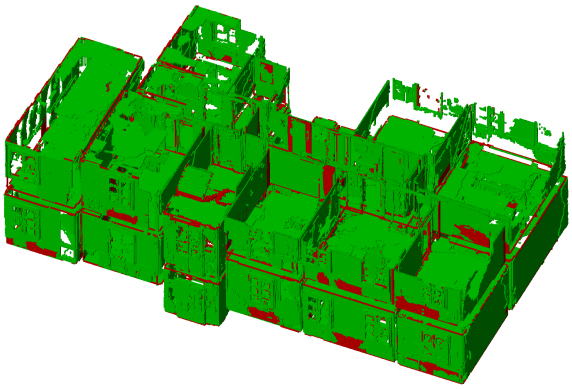
Figure 4 gives an impression of the spatial distribution of wrongly classified voxels for the axes-aligned 'Office' dataset. Voxels being correctly identified as non-empty but having the wrong class label are mainly constituted by wall voxels missing a second class label for ceiling or floor and false wall openings that should have been classified as 'Wall'. Furthermore, it is apparent, that the low ratio of correctly classified voxels in relation to non-empty voxels in ground truth or reconstruction is mainly caused by layers of missing or superfluous voxels along the room surfaces.

5. DISCUSSION

As already mentioned, the low ratio of correctly classified voxels in proportion to the total count of voxels being not empty in the reconstructed model seems to indicate a quite low quality of the classification results. However, as Figure 4 indicates, missing voxels in the reconstruction are in large parts situated as layers along room surfaces as is the case with voxels that should be empty



(a) All cases as listed in Table 4.



(b) Only voxels that are not empty in the reconstruction as well as in the ground truth model.

Figure 4. Spatial distribution of correctly and wrongly classified voxels for the reconstruction of the axes-aligned model from Figure 3(b). The voxels are colourized according to Table 4. A part of the ceiling is removed for the sake of visibility.

according to the ground truth, but are not in the reconstruction. The high amount of voxels that should be empty according to the ground truth along wall surfaces can be attributed to the geometric refinement of the walls presented in Section 3.4.1 which results in a thickening of the voxel representation of the reconstructed walls. Layers of missing voxels, on the other hand, mainly occur on top of ceiling and floor voxel surfaces, which indicates that our approach seems to systematically underestimate the height of these horizontal slabs.

Generally, it needs to be discussed, if an evaluation procedure accounting only for the presence or absence of values on exact voxel positions is adequate in a reconstruction scenario such as this. However, it would rather be more appropriate to try to quantify the displacement between entities such as walls or slabs between ground truth data and reconstruction. This, however, presupposes a meaningful segmentation of such entities and the correct allocation of them between reconstruction and ground truth. Generally, a meaningful comparison between different model representations or mapping results representing the same indoor environment is not a trivial matter. Valuable discussions on this topic can be found e.g. in (Chen et al., 2018) or (Khoshelham et al., 2018).

As indicated by Figure 4(b), there is clearly still potential for improvement in the distinction between actual wall openings and the absence of geometric primitives caused by occlusion. While checking for large occluding furniture objects directly in front of wall opening voxels works quite well in many cases, occlusions from tables where the occluding object is not in front but above the

wall opening are not yet accounted for. Some cases of erroneous wall openings could also potentially be prevented by checking if the adjacent wall of a neighbouring room shows a corresponding opening. If this is not the case, it could be reasoned that the respective wall opening can be closed.

The voxel representation of indoor environments used in this work can be considered as an intermediate form of data representation, that enables a straight-forward detection and segmentation of building elements such as walls and rooms. More compact, prevalent forms of indoor models such as the boundary representation can later on be derived from voxel models in the scope of future work. However, in some application scenarios, voxel models can have advantages over more compact forms of indoor models. Its explicit, discrete, volumetric representation of empty indoor space with clearly defined neighbourhood relations between voxels can facilitate tasks like determining the shortest exit route from arbitrary positions inside the indoor environment as demonstrated by Gorte et al. (2019). In the case of more compact forms of indoor models, solving a suchlike task would be more complicated.

6. CONCLUSION

In this work, we presented a novel approach for reconstructing models of indoor environments in the form of voxel representations from unstructured three-dimensional geometries like triangle meshes. This process encompasses the voxelisation of the input data. Rooms are detected in the resulting voxel grid by segmenting connected voxel components of ceiling candidates and extruding them downwards to find floor candidates. Semantic class labels like 'Wall', 'Wall Opening', 'Interior Object' and 'Empty Interior' are then assigned to the room voxels in-between ceiling and floor by a rule-based voxel sweep algorithm. Finally, the geometry of the detected walls is refined in the voxel representation.

We demonstrated by means of quantitative evaluation, that the presented algorithm holds potential for the reconstruction of indoor environments from sparse and noisy data of complicated and clutter-rich three-dimensional indoor environments. Further work should be directed in merging over-segmented rooms and extracting three-dimensional topological relations between the reconstructed rooms such as adjacency and accessibility through sufficiently large wall openings. Furthermore, the conversion of the resulting voxel model into more common forms of building data representation such as surface models or volumetric objects would enable the further processing of the reconstructed models with prevalent BIM tools.

A further goal for future research is to make these indoor models derived from data acquired by a mobile augment reality device accessible in the context of indoor augmented reality by enriching them with additional space-related information that can then be visualised directly in the location that it refers to. An open research question in this context is how a mobile AR device can be automatically localised within large-scale indoor environments only on the basis of an abstracted indoor model without relying on artificial markers or other kinds of infrastructure.

References

Chen, J., Mora, O. E. and Clarke, K. C., 2018. Assessing the Accuracy and Precision of Imperfect Point Clouds for 3D Indoor Mapping and Modelling. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences IV-4/W6, pp. 3–10.

Díaz-Vilariño, L., Boguslawski, P., Khoshelham, K. and Lorenzo, H., 2019. Obstacle-Aware Indoor Pathfinding Using Point Clouds. ISPRS International Journal of Geo-Information 8(5), pp. 233:1–18.

Flikweert, P., Peters, R., Díaz-Vilariño, L., Voûte, R. and Staats, B., 2019. Automatic Extraction of a Navigation Graph Intended for IndoorGML from an Indoor Point Cloud. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W5, pp. 271–278.

Gorte, B., Zlatanova, S. and Fadli, F., 2019. Navigation in Indoor Voxel Models. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W5, pp. 279–283.

Hübner, P., Clintworth, K., Liu, Q., Weinmann, M. and Wursthorn, S., 2020. Evaluation of HoloLens Tracking and Depth Sensing for Indoor Mapping Applications. Sensors 20(4), pp. 1021:1–24.

Hübner, P., Landgraf, S., Weinmann, M. and Wursthorn, S., 2019. Evaluation of the Microsoft HoloLens for the Mapping of Indoor Building Environments. In: Dreiländertagung der DGPF, der OVG und der SGPF in Wien, Österreich – Publikationen der DGPF, Vol. 28, pp. 44–53.

Hübner, P., Weinmann, M. and Wursthorn, S., 2018. Marker-Based Localization of the Microsoft HoloLens in Building Models. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-1, pp. 195–202.

Jung, W. and Lee, G., 2015. The Status of BIM Adoption on Six Continents. International Journal of Civil, Structural, Construction and Architectural Engineering 9(5), pp. 406–410.

Khoshelham, K., Tran, H. and Acharya, D., 2019. Indoor Mapping Eyewear: Geometric Evaluation of Spatial Mapping Capability of HoloLens. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-2/W13, pp. 805–810.

Khoshelham, K., Tran, H., Díaz-Vilariño, L., Peter, M., Kang, Z. and Acharya, D., 2018. An Evaluation Framework for Benchmarking Indoor Modelling Methods. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-4, pp. 297–302.

Lu, Q. and Lee, S., 2017. Image-Based Technologies for Constructing As-Is Building Information Models for Existing Buildings. Journal of Computing in Civil Engineering 31(4), pp. 04017005/1–14.

Ma, Z. and Liu, S., 2018. A Review of 3D Reconstruction Techniques in Civil Engineering and their Applications. Advanced Engineering Informatics 37, pp. 163–174.

Nikoohemat, S., Peter, M., Oude Elberink, S. and Vosselman, G., 2018. Semantic Interpretation of Mobile Laser Scanner Point Clouds in Indoor Scenes Using Trajectories. Remote Sensing 10(11), pp. 1754:1–23.

Ochmann, S., Vock, R. and Klein, R., 2019. Automatic Reconstruction of Fully Volumetric 3D Building Models from Point Clouds. ISPRS Journal of Photogrammetry and Remote Sensing 151, pp. 251–262.

Tran, H. and Khoshelham, K., 2019. A Stochastic Approach to Automated Reconstruction of 3D Models of Interior Spaces from Point Clouds. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W5, pp. 299–306.

Xie, L., Wang, R., Ming, Z. and Chen, D., 2019. A Layer-Wise Strategy for Indoor As-Built Modeling Using Point Clouds. Applied Sciences 9(14), pp. 2904:1–27.

Yang, F., Zhou, G., Su, F., Zuo, X., Tang, L., Liang, Y., Zhu, H. and Li, L., 2019. Automatic Indoor Reconstruction from Point Clouds in Multi-Room Environments with Curved Walls. Sensors 19(17), pp. 3798:1–19.