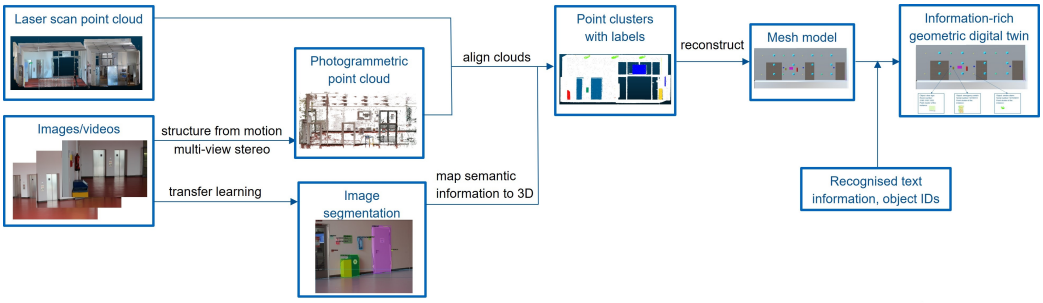
Graphical Abstract

Enriching geometric digital twins of buildings with small objects by fusing laser scanning and AI-based image recognition

Yuandong Pan, Alexander Braun, Ioannis Brilakis, André Borrmann



Highlights

Enriching geometric digital twins of buildings with small objects by fusing laser scanning and AI-based image recognition

Yuandong Pan, Alexander Braun, Ioannis Brilakis, André Borrmann

- Laser scanning and photogrammetric technologies are fused in the pipeline.
- Semantic information in images is mapped to 3D laser-scanned point clouds.
- Small objects of different classes are detected in the creation of digital twins.
- Text information is recognised and used to enrich digital twins.

Enriching geometric digital twins of buildings with small objects by fusing laser scanning and AI-based image recognition

Yuandong Pan[a,b,1,*], Alexander Braun[a], Ioannis Brilakis[b,c], André Borrmann[a,b]

[a]*School of Engineering and Design, Technical University of Munich, Arcisstr. 21, Munich, 80333, Germany*
[b]*Institute for Advanced Study, Technical University of Munich, Lichtenbergstrasse 2a, Munich, 85748, Germany*
[c]*Department of Engineering, University of Cambridge, , 7a JJ Thomson Avenue, Cambridge, CB2 1PZ, United Kingdom*

Abstract

This paper addresses the challenge of enriching geometric digital twins of buildings, with a particular emphasis on capturing small but important entities from the electrical and the fire-safety domain, such as signs, sockets, switches, smoke alarms, etc. Unlike most previous research that focussed on structural elements and processed laser point clouds and images separately, we propose a novel method that fuses laser scanning and photogrammetry methods to capture the relevant objects, recognise them in 2D images and then map these to a 3D space. The considered object classes include electrical elements (light switch, light, speaker, socket, elevator button), safety elements (emergency switch, smoke alarm, fire extinguisher, escape sign), plumbing system elements (pipes), and other objects with useful information

*Corresponding author
[1]Part of the research was conducted while visiting the University of Cambridge.

(door sign, board). Semantic information like class labels is extracted by applying AI-based image segmentation and then mapped to the 3D point cloud, segmenting the point cloud into point clusters. We subsequently fit geometric primitives to the point clusters and extract text information by AI-based text detection and recognition. The final output of our proposed method is an information-rich digital twin of buildings that contains geometric information, semantic information such as object categories and useful text information which is valuable in many aspects, like condition monitoring, facility maintenance and management. In summary, the paper presents a nearly fully-automated pipeline to enrich a geometric digital twin of buildings with details and provides a comprehensive case study.
Keywords: digital twin, deep learning, object detection, text recognition, 3D reconstruction
PACS: 0000, 1111
2000 MSC: 0000, 1111

1. Introduction

This research is about enriching Geometric Digital Twins (GDTs) with small objects. By enriching, we refer here to the process of adding more categories of objects to the GDTs of basic elements in a building. By geometric digital twins, we refer here to a digital twin with geometric data only. A digital twin of a building here is defined as a regular-updated digital replica of a physical building that can represent the current condition of the building [1]. By small objects, we refer here to the elements that are smaller in scale in comparison with structural elements (like walls, floors, ceilings). In this

paper, we focus on enriching geometric digital building twins by adding these elements. Meanwhile, instead of only segmenting point clouds, we extract text information such as object IDs to recognise object instances.

Generating a geometric digital twin of an existing asset is a process that consists of the following steps: (1) capturing raw visual and spatial data in the form of RGB imagery and laser-scanned point clouds; (2) detecting geometric objects and geometric relationships of objects in the raw data. Step 1 of this process is significantly more automated than step 2 and requires much fewer labour hours [2]. The cost and effort needed to complete step 2 for most assets appear to counteract the perceived value of the resulting GDTs. Step 2 can be broken down into the detection of large objects (such as ceilings, floors, walls) and small objects (such as fire extinguishers, smoke alarms) by their scale. Several recent methods have been proposed for the former ([3], [4], [5]), and have been validated to robustly automate this task. However, no method has yet been proposed for the latter. This is the challenge that this paper aims to focus on.

Apart from those relatively large structural elements, small elements (such as fire alarms, emergency switches) should also be included in an enriched building twin, these being helpful for facility managers. In the Repair and Maintenance (R&M) activities of a building, Mechanical, Electrical and Plumbing (MEP) costs usually constitute the largest share of total costs [6]. Therefore, a building twin would be more valuable if it were to contain those elements that are frequently required in facility management processes. In addition, facility management involves more accurate data about the floor plans, space utilization, asset location, and technical plants [7]. Text infor-

mation such as room numbers and serial numbers (IDs) next to assets that can identify the corresponding assets (as shown in Figure 1) is very helpful, especially when managing large facilities. These IDs exactly represent the corresponding object instances in an asset and make the link between physical assets and digital twins much clearer. Therefore, it is valuable to add the information to an enriched digital twin of buildings. Unfortunately, this work is currently mostly manual work.

In summary, the great manual effort required to create an enriched digital twin is too costly when compared with the perceived value of the resulting model. For these reasons, there is a high demand for a higher degree of automation in the generation of an information-rich digital building twin.

In this paper, the authors propose a novel framework to enrich a geometric building twin by fusing point cloud processing and object detection in images. The proposed method of information enrichment can be used to complete as-built models generated by other methods of creating geometric digital twins of structural elements. In particular, this paper presents the following contributions:

a) Because the performance of detecting small-scale elements directly in point clouds is significantly lower than in images, unlike most previous methods that exclusively use point clouds as input, the approach presented here extracts semantic information from images by deep learning and then maps the extracted semantic information to laser-scanned point clouds.

b) While most of the previous approaches only detect primary elements (like ceilings, walls, floors, windows and doors), our proposed method includes small but highly relevant objects in the energy and the fire-safety

sub-systems that are essential for maintaining and monitoring buildings (like smoke alarms, emergency switches);

c) In order to create an information-rich building twin, other useful information (text and numbers) is detected in images by applying optical character recognition (OCR) technologies to detect object IDs and recognise object instances. Some examples are shown in Figure 1. The detected machine-encoded texts include the room number on the door sign, as well as numbers or text corresponding to the detected objects, which helps to identify the object instance in the physical asset.

(a) Room number on the door sign

(b) Serial number on the fire switch

(c) Serial number next to the smoke alarm

Figure 1: Text information in a building

The rest of this paper is organised as follows: research background including state of the art is reviewed in Section 2; the proposed pipeline is introduced in Section 3 in detail; experiments and implementation details are shown in Section 4; conclusions and future work are discussed in Section 5.

2. Background

In this paper, the authors aim to enrich a geometric digital building twin. Apart from structural objects such as ceilings, floors and walls, a rich building twin should also contain other small but important objects, for example objects from the energy and fire-safety sub-systems such as smoke alarms, emergency switches, etc. In our previous research [3], we have already reconstructed ceilings, floors, and walls of buildings by initially detecting the void space inside rooms. These structural elements do not fall within the scope of this paper. Compared to structural elements in a building, other components are usually small in size and have different geometry properties, which makes it hard to apply the same methods to detect those small-scale elements. Therefore, 2D information from images and 3D information from laser-scanned point clouds are connected and integrated into the proposed approach. We believe that this combination provides a significant advantage over using the laser-scanned point cloud alone, especially for detecting small-scale components in a building. In addition, text information, including serial numbers and IDs, can also be extracted from 2D images, and the detected information can be used to enrich the digital twin further.

Recent research into small objects detection is discussed in Section 2.1. As object detection and text recognition in images are achieved by deep learning in our approach, recent research in both fields is introduced in Section 2.2 and 2.3 respectively. Finally, research gaps are summarised in Section 2.4.

2.1. Secondary object reconstruction in buildings

With regard to elements located on wall surfaces, such as sockets and light switches, in [8], the authors designed a robot that can recognise doors, door handles, and sockets to achieve the door task and plugging task. The electrical outlet pattern is detected in camera images by feature detection, and a laser scanning sensor is used to find the pose of a wall. In [9], the authors detect light switches and sockets in orthographic 2D images by a random forest classifier. They use a feature descriptor pool to measure the probability of the detection. A method was designed in [10] that allows a mobile robot to get on/off an elevator in a multistory building. An algorithm is presented for recognising elevator buttons, where the input image is first converted to a binary image, and then the candidates of buttons and floor numbers are filtered out and ambiguous candidates are rejected by applying a neural network. While most of these methods are used to help robots recognise specific objects in the environment and perform a given task, little work has been done in the AEC domain. In [6], the authors proposed a method to detect objects such as switches, ducts and signs in a coloured point cloud. Depending on whether the objects have geometric discontinuities or colour discontinuities in the wall area, potential regions of interest are computed in depth images and colour images with regard to the wall plane, respectively. The region of interest is then matched to a predefined depth model database and a predefined colour model database that contain object classes in the scene.

With regard to elements mounted on the ceiling such as lighting, in [11], the authors proposed a recognition method based on thermal-mapped point

clouds for building elements consisting of electrical systems and heating, ventilation, and air-conditioning (HVAC) components. Assuming the temperatures of these elements are different from other parts of the ceiling, the points of corresponding elements can be extracted from the point cloud. In [12], the authors used two steps to recognise objects in thermal-mapped point clouds: segmentation with thermal information and classification with geometric information. The target objects are light fixtures on the ceilings, monitors on the wall and humans in the environment. In [13], the authors extract the ceiling plane first and then convert the laser-scanned point cloud to an image of the ceiling. Fluorescent lightings and circular low-energy bulbs are detected from the image by Harris corner detector and Hough transformation. In [14], a method to detect tunnel luminaires from the point cloud is proposed. In this approach, they use assumptions that are only valid in the tunnel, for example luminaires are located at higher points at the side of the tunnel and have brighter colour patterns than their surroundings.

With regard to identifying pipes, in [15], the authors proposed a method to detect pipe spools in a cluttered point cloud. The method used curvature estimation, points clustering, and feature matching to extract pipe spool objects. In an office building, pipes are rarely visible because they are usually located inside the walls or behind suspended ceilings. In [16], the authors proposed a neural network to segment RGBD images into 13 building component classes which include classes of small components such as duct, plumbing, conduit, etc. In [17], the authors used deep learning to detect and differentiate between different pipes in laser scanning point clouds of industrial facilities.

2.2. Object detection networks and transfer learning

In computer vision, object detection refers to identifying an object and precisely estimating its location [18]. One of the most widely used algorithms in object detection is Region-based Convolutional Neural Network (RCNN) (RCNN) [19]. In RCNN, regions of interest are identified first and then classified by Convolutional Neural Network (CNN) to detect objects in the regions. Since original RCNN is relatively slow, some variants of RCNN have been proposed, like fast-RCNN [20], mask-RCNN [21].

In the AEC domain, researchers have also applied and proposed different network architectures to achieve their research objectives, for example defect and damage detection ([22], [23], [24]), worker detection on construction sites ([25], [26], [27]).

A neural network can be trained from scratch on a specific dataset. However, in order to achieve optimal results, it requires a large training set as well as substantial processing time [28]. Therefore, transfer learning [29] is proposed to overcome the problems and improve performance. Transfer learning is a process where a neural network is pre-trained on a related larger dataset and re-trained on a user-specific dataset. Currently, there are several large, publicly available datasets that are used to pre-train a neural network, such as ImageNet [30], which contains more than one million images for training, the Pascal VOC 2012 dataset that contains more than 20,000 images [31], the COCO dataset contains more than 300,000 images [32] with 2.5 million instances.

2.3. Text detection and recognition

In a building, some elements contain texts and numbers that are also valuable for facility management, such as room numbers on a door sign. In large facilities, entities of some electrical elements (such as smoke alarms, emergency switches) usually have a unique serial number in order to clearly label entities and make facility management more efficient. It is also very helpful to attach this information to the objects in the building twin, recognising and identifying objects at an instance level. There are usually two steps to extracting the information from images: text detection and text recognition.

With regard to text detection, neural networks that are used in object detection can also be used to detect text in an image, such as Mask-RCNN [21] because text area can also be considered a type of object. Researchers have also proposed neural networks that aim to detect text in an image, like [33], [34], [35], [36], [37]. These networks were proposed to detect arbitrary-shaped text in an image and can be trained on large, publicly available datasets like ImageNet [30].

With regard to text recognition, some neural networks have been proposed to recognise regular and irregular text in an image, like [38], [39], [40], [41]. These networks can be trained on text image datasets, such as the SynthText dataset [42], which contains approximately 800 thousand synthetic scene-text images, the COCO-Text dataset [43] with more than 60 thousand real images and around 239 thousand annotated text instances.

In the field of building reconstruction, only a few previous works deal with text detection and recognition, and these focus on CAD drawings. In [44],

the authors used Optical Character Recognition (OCR) technology to extract text information from CAD drawings and then added detected information to the as-is digital model of buildings. In [45], the authors applied OCR to extract the object information from the images of structural drawings (i.e., grids, columns and beams) and generate Industry Foundation Class (IFC) models for buildings.

2.4. Research gaps

We summarise the research gaps in enriching a geometric digital twin of buildings as follows:

a) Previous work focuses solely on structural elements and does not consider other smaller but still valuable objects in a building. While some researchers detect geometric and colour discontinuities to find specific classes of small objects in images, these approaches do not apply AI-based methods to enhance the performance of detection in point clouds. Moreover, most previous work dealt with only some classes of objects, and there is still a lack of comprehensive object categories when creating a building twin. The reason is that, unlike structural elements, visible small object classes differentiate much in different facilities.

b) Most previous work used only point clouds to achieve object detection and reconstruction. 3D deep learning networks for point cloud segmentation perform well for structural elements but much worse for smaller objects, as shown in Table 3. Because methods of object detection in 2D images are more mature and can provide better performance than those in 3D point clouds, there is a potential performance improvement when concatenating the information from various input sources. But there is still a lack of a

straightforward way to map information in images to point clouds.

c) While text information like object IDs attached to corresponding objects is also important in a rich building twin, none of the previous works considered adding text information, while such information can usually be extracted only in 2D images. There is still a lack of creating a comprehensive information-rich building twin which contains geometric and semantic information.

3. Proposed solution

3.1. Scope

In our previous research [3], we already reconstructed structural elements, so that these do not fall within the scope of this paper. In this paper, we propose a novel approach that processes information from images as well as point clouds together. Our methods focus on 12 important and relatively small-scale elements (compared to walls, ceilings, floors) in buildings: light switch, emergency switch, light, smoke alarm, escape sign, speaker, fire extinguisher, socket, pipe, board, door sign, elevator button, trash bin.

3.2. Overview

The overall process of the proposed method is illustrated in Figure 2. The inputs for our proposed method are point clouds acquired by laser scanners and videos or images captured in the same area of a building. It should be noticed that we also collect an annotated image dataset that contains the target objects. But these images are only used to train a deep learning model and are not required in the reconstruction pipeline. The outputs are

point clusters with labels and a mesh model for each element that is found. All points in one point cluster have an identical label. The overall goal is to create a comprehensive digital building model represented by mesh geometry and enriched with semantic information of the detected elements. To achieve this, we map information in 2D images onto a 3D laser-scanned point cloud. We start by detecting objects in images or videos by applying the transfer learning technique. The next step is to construct a photogrammetric point cloud and align this point cloud to the laser-scanned point cloud. Subsequently, the semantic information from 2D images or videos is projected onto the 3D point cloud. After finding a best-fitting label for each point, we obtain the output point clusters of different objects. In the final step, we fit a pre-defined mesh model to each found instance.

3.3. Object detection in image

In this step, we aim to detect the 12 element classes listed in Section 3.1 from images or videos. Recently, Deep Neural Networks (DNN) [46], especially the introduction of RCNN [19], have proven effective in object detection in 2D images [47]. But we still need to prepare our own dataset because those publicly available datasets, like Imagenet [30], one of the largest online available image datasets, does not contain all of the categories we need. Even if some of the target categories are present in Imagenet, such as fire alarms and fire extinguishers, there are no labelled instances available. Therefore, we cannot detect the target objects in images or videos that were captured in buildings by publicly available pre-trained models because these models are trained on a dataset lacking the categories we require. The available networks must be re-trained for our application domain. In the conducted

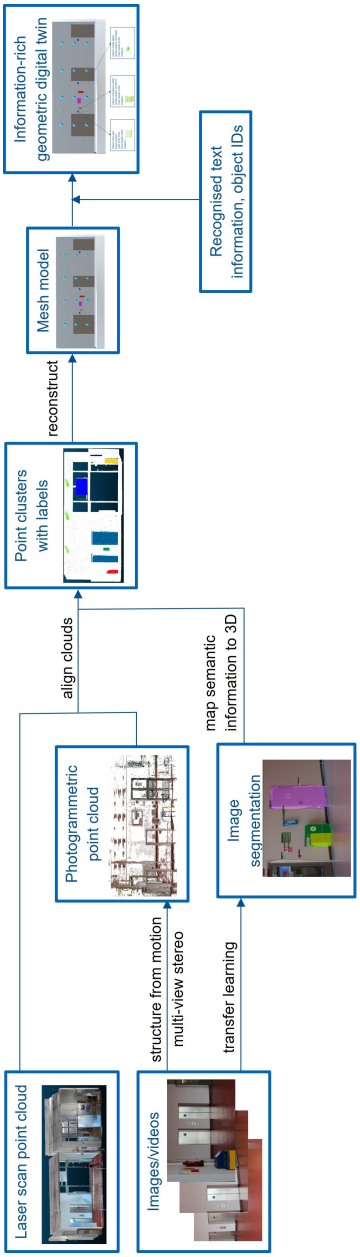


Figure 2: The overall procedure of the proposed method

research, we prepared our own dataset by manually labelling images that we captured in public buildings, more precisely office buildings on the inner-city

campus of the Technical University of Munich (TUM).

In practice, there is no required minimum number of images when training a neural network. In Imagenet [30], categories like fire/smoke alarm and fire bell contain hundreds of labelled images. If we follow the similar setup that each category has hundreds of images, thousands of images are required for a dataset with 12 classes, which leads to a huge amount of labelling work. Considering the vast human effort to label these images manually, we decided to use transfer learning techniques. As its name implies, transfer learning [29] means using the knowledge learned previously to solve new, but related problems. When starting with a pre-trained model that has already been trained on thousands of images, we do not need as many images as if we were training a network from scratch because the model has already "seen" and "learnt" from lots of images.

Object detection in images results in finding a bounding box for a detected instance. Obviously, some regions within the bounding box do not belong to this instance, especially when the object is not a rectangle or inclined in the image. Since we want to map semantic information obtained in 2D images to the 3D point cloud in further steps, we need to reduce this kind of error here and apply image segmentation instead of instance detection. To this end, we use a variant of CNN called Mask RCNN [21] that detects objects in images by generating a mask for each instance. By doing so, we can find a more precise contour of the object instance than the mere bounding box. Some results of image segmentation and bounding box prediction of various objects are illustrated in Figure 3.

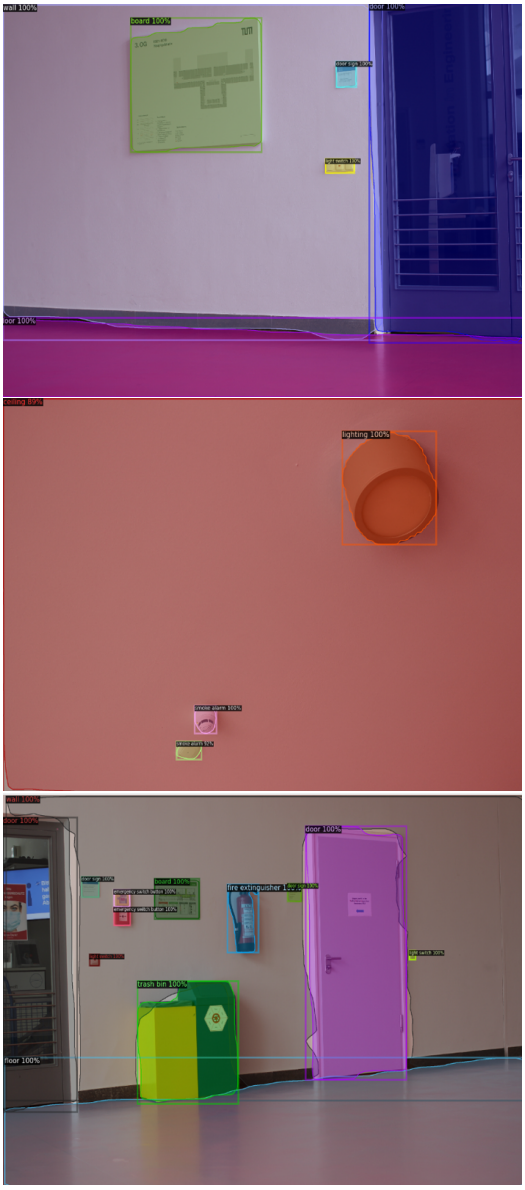


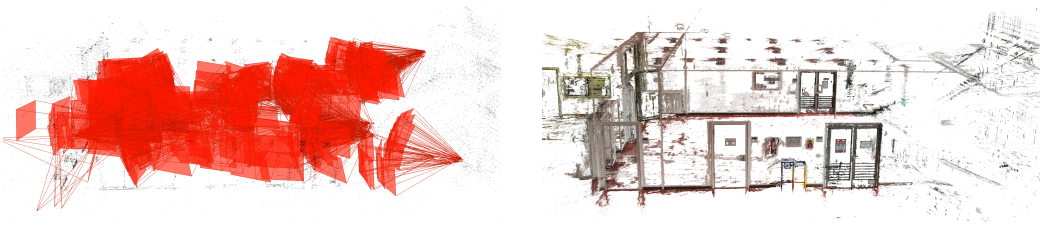
Figure 3: The object detection result by image segmentation mask and bounding box

3.4. Creating a photogrammetric point cloud

In [48], the authors used the photogrammetric point cloud to connect images and Building Information Modeling (BIM) models. Similarly, in our

proposed approach, the photogrammetric point cloud acts as the bridge that connects 2D information in images with 3D information in the laser-scanned point cloud. In the photogrammetric process, the extrinsic and intrinsic camera parameter matrices of pictures are estimated. Images or videos are supposed to be taken from different viewpoints within the area and cover as much information as possible. In our approach, we apply COLMAP [49] [50], an open-source Structure-from-Motion (SfM) and Multi-View Stereo (MVS) software, to reconstruct photogrammetric point clouds. The input of SfM is a set of overlapping images taken from different viewpoints. It starts with feature detection and extraction, continues with feature matching and geometric verification, and then reconstructs the object in 3D space, including the reconstructed intrinsic and extrinsic camera parameters of all images. MVS takes the output of SfM to compute depth and normal information for pixels in all images and creates a dense point cloud of the scene.

The estimated camera poses (position and orientation) of each image and the reconstructed sparse photogrammetric point cloud are illustrated in Figure 4. As we can see, the edges are reconstructed quite well, while plane faces of elements like walls, ceilings, and floors are missing. This is because almost no features can be detected and extracted on these weakly textured surfaces, like a planar white wall, in the SfM process. However, these weakly textured surfaces can be captured quite well by laser scanners. This is one of the reasons why we propose the use of both laser-scanned point clouds and images to create sufficiently detailed and complete digital twins. In this way, we can acquire all of the required information by using both techniques to capture buildings.



(a) Camera poses in sparse model **(b)** Reconstructed dense dense point cloud

Figure 4: An example of estimated camera poses and the reconstructed point cloud

3.5. Point clouds alignment

Laser scanners measure the distance by transmitting light and sensing the return from objects [51] so that laser-scanned point clouds represent the actual scale of the environment. In contrast, photogrammetric point clouds extract information from 2D images – they do not represent the actual scale in world units unless additional information is considered, such as the size of an object. To perform the necessary registration of the two point clouds, we align the photogrammetric point cloud with the laser-scanned point cloud so that the photogrammetric point cloud also represents the environment in its actual size.

The photogrammetric point cloud is transformed to the coordinate of laser-scanned point cloud by

$$\mathbf{Q} = \mathbf{MP}, \tag{1}$$

where $\mathbf{P}$ denotes the point set of the photogrammetric point cloud, $\mathbf{Q}$ denotes the point set of the photogrammetric point cloud transformed to the coordinate of the laser-scanned point cloud, $\mathbf{M}$ denotes the transformation matrix that transforms points from the coordinate of the photogrammetric

point cloud to the coordinate of the laser-scanned point cloud.

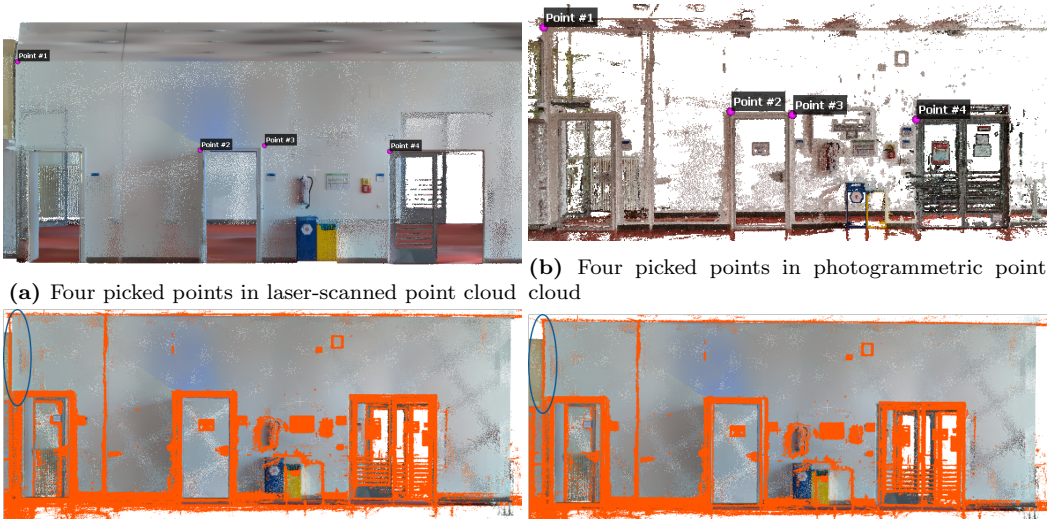
4×4 transformation matrices are widely used to represent non-linear transformations in 3D space. In our approach, we use two steps to determine the 4×4 transformation matrix: the rough alignment step and the refinement step. In the rough alignment step, we use 4 pairs of points from the photogrammetric point cloud and laser-scanned point cloud to compute the roughly estimated transformation matrix from photogrammetric point cloud coordinate to laser-scanned point cloud coordinate, denoted by $\mathbf{M_1}$. In this step, we only need to select points roughly and get a rough alignment result. These point pairs can be chosen at random, and could be any key points in point clouds, such as room and door corners, the centre of an object, etc. After rough alignment, we use the Iterative Closest Point (ICP) algorithm [52], to refine the alignment and obtain the refinement transformation matrix $\mathbf{M_2}$. The overall transformation matrix $\mathbf{M}$ can be computed by

$$\mathbf{M} = \mathbf{M_2}\mathbf{M_1}. \tag{2}$$

The photogrammetric point cloud can then be transformed to the coordinates of the laser-scanned point cloud by applying Equation 1. This alignment process is illustrated in Figure 5. When comparing the marked area in Figure 5c with that in Figure 5d, it is clear that the refinement step improves the alignment result.

3.6. Find visible laser scanning points in each image

In this step, we determine whether a point from the laser-scanned point cloud is visible in each image that is used to reconstruct the photogrammetric



(a) Four picked points in laser-scanned point cloud

(b) Four picked points in photogrammetric point cloud

(c) Rough alignment result (Photogrammetric point cloud is set to red color)

(d) Refined alignment result (Photogrammetric point cloud is set to red color)

Figure 5: The alignment process of photogrammetric and laser-scanned point cloud

point cloud. Because the photogrammetric point cloud and the laser-scanned point cloud are aligned already, the estimated parameters (extrinsic and intrinsic camera parameters) from the reconstruction process are also mapped into 3D space. The extrinsic camera matrix and intrinsic parameter matrix are known for each image or frame of a video. Based on the matrices, we can find which points are visible at each camera position and captured in the corresponding image.

As the transformation matrix that transforms points from a photogrammetric point cloud coordinate to a laser-scanned point cloud coordinate is $\mathbf{M}$, any point $\mathbf{p} = \begin{bmatrix} x_0, y_0, z_0 \end{bmatrix}^{\mathbf{T}}$ in the original laser-scanned point cloud $\mathbf{S}$

can be transformed to the coordinate of the photogrammetric point cloud by

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ d_1 \end{bmatrix} = \mathbf{M^{-1}} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix}, \tag{3}$$

where $\Big[x_0, y_0, z_0, 1\Big]^T$ is the homogeneous coordinates of this point $\mathbf{p}$, $\mathbf{M^{-1}}$ is the inverse matrix of $\mathbf{M}$, and $\Big[x_1, y_1, z_1, d_1\Big]^T$ is the new calculated homogeneous coordinates of the point in the coordinate of photogrammetric point cloud. Normalization is then applied by dividing each vector component by d_1,

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \frac{1}{d_1} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ d_1 \end{bmatrix}, \tag{4}$$

where $\Big[x_2, y_2, z_2, 1\Big]^T$ is the normalized homogeneous coordinate vector of point $\mathbf{p}$ in the coordinate of photogrammetric point cloud.

The next step is to transform every point from the coordinate of the photogrammetric point cloud to the camera coordinate of the image. In this paper, we use $\mathbf{N}$ to denote the whole image set that is used to reconstruct the photogrammetric point cloud, $\mathbf{n}_i$ to denote the i^{th} image in the image set $\mathbf{N}$. For one single image $\mathbf{n}_i$, $\mathbf{M}^i_{ext}$ and $\mathbf{M}^i_{int}$ denote the corresponding camera extrinsic and intrinsic parameter matrices. The extrinsic parameter matrix

can be defined as

$$\mathbf{M}_{ext}^{i} = \begin{bmatrix} & \mathbf{R}_i & & \mathbf{T}_i \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{5}$$

where $\mathbf{R_i}$ is the 3×3 rotation matrix $\mathbf{R_i} = \begin{bmatrix} r_{11}^i & r_{12}^i & r_{13}^i \\ r_{21}^i & r_{22}^i & r_{23}^i \\ r_{31}^i & r_{32}^i & r_{33}^i \end{bmatrix}$, and $\mathbf{T_i}$ is the 3×1 translation matrix $\mathbf{T_i} = \begin{bmatrix} t_1^i \\ t_2^i \\ t_3^i \end{bmatrix}$ of the image $\mathbf{n}_i$.

The intrinsic parameter matrix can be represented by

$$\mathbf{M}_{int}^{i} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{6}$$

where f_x and f_y are the effective focal length of the camera measured in units of image pixels in the horizontal and vertical directions, c_x and c_y are the pixel coordinates of the principal point. Additionally, s denotes the skew coefficient for the camera. This is zero if the image axis is perpendicular to the image plane. It should be noticed that no distortion is assumed here. 3D points can be then transformed in camera coordinates by

$$\begin{bmatrix} x_3 \\ y_3 \\ z_3 \\ 1 \end{bmatrix} = \mathbf{M}_{out}^{i} \begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11}^i & r_{12}^i & r_{13}^i & t_1^i \\ r_{21}^i & r_{22}^i & r_{23}^i & t_2^i \\ r_{31}^i & r_{32}^i & r_{33}^i & t_3^i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} \tag{7}$$

and subsequently transformed to the image plane by computing

$$\begin{bmatrix} x_4 \\ y_4 \\ z_4 \end{bmatrix} = \mathbf{M}^i_{int} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_3 \\ y_3 \\ z_3 \end{bmatrix}, \tag{8}$$

where x_3, y_3, z_3 are coordinates in the camera coordinate, and x_4, y_4, z_4 are the perspective projected coordinates on the image coordinate. By homogeneous coordinate normalisation, we obtain the image coordinates of the projected point in the image plane:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{z_4} \begin{bmatrix} x_4 \\ y_4 \\ z_4 \end{bmatrix}, \tag{9}$$

where u and v are the pixel coordinates in the horizontal and vertical direction in the image plane.

By using the Equations 3 to 9, all points in the original laser-scanned point cloud can be projected into the image plane. However, there are points in the cloud that are not in the field of view of the given camera pose and intrinsic parameters. Assuming the dimension of the image in pixels is $W \times H$, if a point (x_0, y_0, z_0) in the original laser-scanned point cloud and its projected point in the image plane (u, v) can be seen in the image, the point should follow these conditions:

$$0 \leq u \leq W, 0 \leq v \leq H. \tag{10}$$

(a) An image captured in an area of a hallway

(b) The same area in the laser-scanned point cloud

(c) Transform the point cloud to the camera frame (camera at the origin)

(d) Visible points from laser-scanned point cloud at the camera pose

Figure 6: The process of finding visible points in an image (ceiling points in the point cloud are removed for better visualisation)

The process of checking the visibility of laser-scanned points for one image is illustrated in Figure 6. As we can see in subfigure (d), the visible area shown in the laser-scanned point cloud is identical to the image scene.

Up to this step, the visibility of a point is only determined by the camera parameters. That means that as long as the points fulfil Condition 10, they are considered visible points, which makes the camera see "through" the wall. As shown in Figure 7, it is obvious that some points should not be visible, like points behind the surface of the wall.

We use the raycasting method [53] to remove those points that should not

Figure 7: Top view of visible points at camera position in Figure 6. Points behind the wall (within the red dash line) are actually not visible from the camera pose.

be seen at the current camera position. However, rays might pass through the point cloud without intersecting any points because point clouds are actually discrete points in 3D space. Therefore, point clouds are usually voxelised before raycasting [54]. Figure 8 shows how raycasting works in a voxelised point cloud. Rays shoot from the camera position to each point in the point cloud. While a dark blue voxel means there are points within the

voxel, a light blue voxel indicates no points in the voxel. If a ray starting from the camera does not pass through any other dark blue voxels, its target point is visible at the camera position. In contrast, if a ray passes through at least one other dark voxel before reaching the target point, this target point is occluded by other voxels in between.

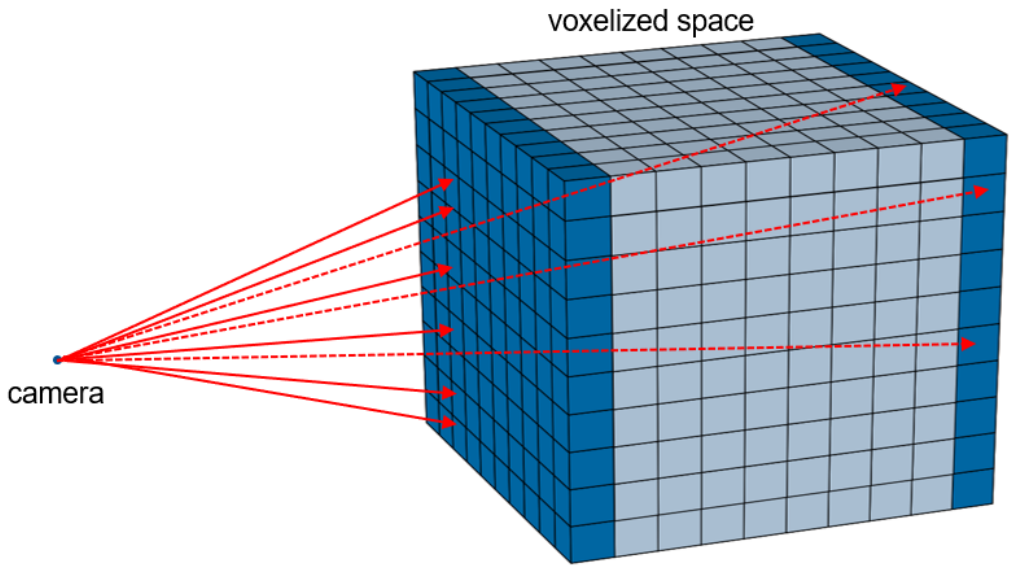


Figure 8: Raycasting method in a voxelized point cloud. There are points in dark blue voxels but no points in light blue voxels. Rays of dotted lines starting from the camera intersect other dark blue voxels before reaching the target voxel. These target voxels are occluded by the voxels between the camera and themselves.

The remaining visible points after applying the raycasting method to the point cloud are shown in Figure 9. In the raycasting process, the voxel size has an enormous impact on performance. A further discussion on finding the best voxel size is presented in Section 4.3.

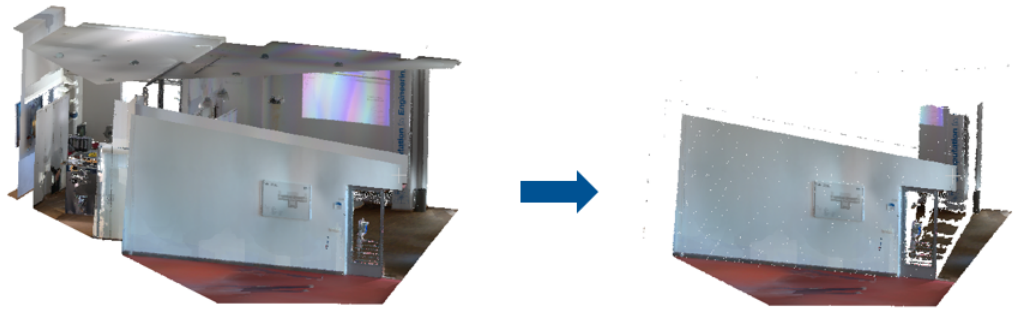


Figure 9: Apply raycasting to the visible points at the camera position

3.7. Map 2D semantic information to a 3D space

In this step, the semantic information detected from 2D images or videos in Section 3.3 is mapped to the 3D space. We use Mask-RCNN [21] to detect objects in images, and the result for each detected instance (like a board, a smoke alarm, etc.) is a mask. The mask is a matrix that is exactly the same size as the input image, but has only two values, 0 and 1. While pixels with a value of 0 are background, pixels with a value of 1 are where the detected instance is located in the image. As shown in Figure 10c, 10e, and 10g, when a mask is applied to an image, only the image area that belongs to the detected area can be seen.

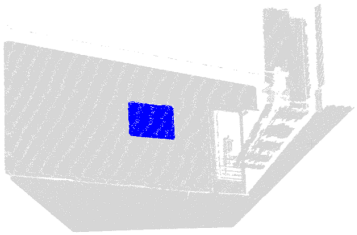
In the previous step, all visible points (x_0, y_0, z_0) in 3D space are already transformed to 2D coordinates (u, v) in the image plane. At this step, we check that every point in the image plane is in the predicted segmentation mask or the background area. Points located in the instance mask of three categories are shown in Figure 10c, 10e, and 10g for example.

Because we use images/videos to reconstruct the photogrammetric point cloud, many images have overlapping areas. In order to record semantic infor-

(a) Image segmentation result

(b) Visible points for the camera

(c) A detected board instance



(d) The board points in 3D

(e) A door sign in image

(f) The door sign points in 3D

(g) A light switch in image

(h) The light switch in 3D

Figure 10: Image segmentation masks and corresponding points in 3D of different instances

mation from all images, an $M \times N$ matrix $\mathbf{L}$ is used to accumulate predicted information from all images, where M denotes the number of categories and N denotes the number of points in the laser-scanned point cloud. If the k^{th}

point's projection in the image plane is within a mask of category j, the term $\mathbf{L}_{j,k}$ in the matrix $\mathbf{L}$ would be increased by 1, where $1 \leq j \leq M$ and $1 \leq j \leq L$.

One point in the laser-scanned point cloud is usually visible in multiple images, and the predicted labels from these images might be different. Therefore, it is necessary to retain all information and find the best-fitting label prediction for each point in later steps. The pseudocode of the method proposed in Section 3.5 to 3.7 is shown in Algorithm 1.

Algorithm 1 The mapping algorithm from 2D to 3D.

Input:
One point $\mathbf{s}_k \in \mathbf{S}$, laser-scanned point cloud set $\mathbf{S}$;
Image set used to reconstruct the photogrammetric point cloud $\mathbf{N}$;
For image $\mathbf{n}_i \in \mathbf{N}$, camera extrinsic and intrinsic parameter matrices $\mathbf{M}^i_{ext}$ and $\mathbf{M}^i_{int}$;
Predicted segmentation mask $\mathbf{m}^i_j \in \mathbf{K^i}$ for image $\mathbf{n}_i$, category j, $\mathbf{K^i}$ denotes all predicted masks for image $\mathbf{n}_i$;
Transformation matrix from photogrammetric point cloud to laser-scanned point cloud $\mathbf{M}$;
Function to check whether a point is visible at a camera position $\alpha()$;
Function to check whether a point belongs to a mask $\beta()$;
Initialize:
Matrix used to count labels for all points in point cloud $\mathbf{L} \leftarrow \mathbf{O}$;
Algorithm:
for $\mathbf{s}_k \in \mathbf{S}$ **do**
  Point in the coordinate of photogrammetric point cloud $\mathbf{p}_k = \mathbf{M}^{-1} \times \mathbf{s}_k$
  for $\mathbf{n}_i \in \mathbf{N}$ **do**
    Point in image plane $\mathbf{c}_k = \mathbf{M}^i_{int} \times \mathbf{M}^i_{ext} \times \mathbf{p}_k$
    if $\alpha(\mathbf{c_k})$ is **FALSE then**
      continue
    end if
    for $\mathbf{m}^i_j \in \mathbf{K^i}$ **do**
      if $\beta(\mathbf{c_k})$ is **TRUE then**
        count label j for point k once, $\mathbf{L}_{j,k} = \mathbf{L}_{j,k} + 1$
      end if
    end for
  end for
end for
return $\mathbf{L}$

3.8. Find best-fitting labels for all points

As described in the previous section, we need to find a best-fitting label for each point in 3D from the $M \times N$ label matrix $\mathbf{L}$.

Two values are used to determine the best label for each point. For

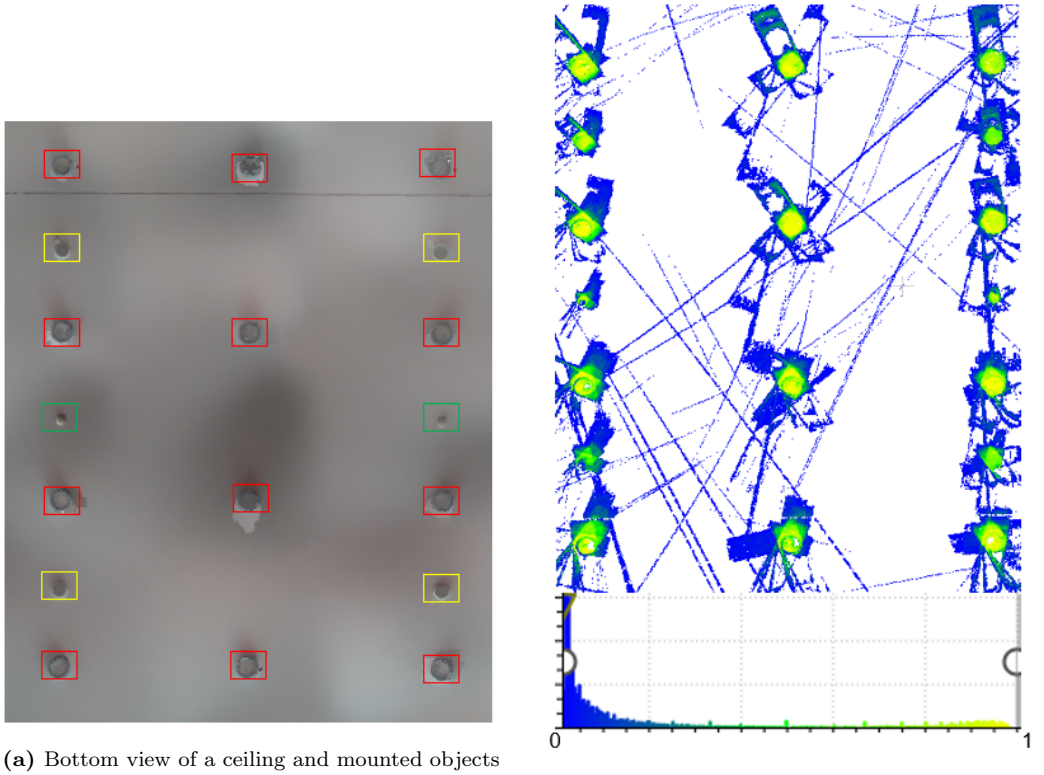
one point $\mathbf{p}_i$ in the laser-scanned point cloud, N_i is the number of images where the point can be seen, $\mathbf{L}_{j,i}$ is the number of images where the point is within the predicted mask of category j. But it should be noted that N_i is not equal to the sum of N_i^j for all categories because a point could also be located in the "background" area instead of the mask area. Basically, a point in the 3D point cloud would be assigned to the label with the maximum occurrence from different images when it is predicted diversely in different images. Furthermore, we use two values to represent how certain the label assigned to the i^{th} point $\mathbf{p}_i$ is:

$$U_i = \max_{1 \leq j \leq M} \mathbf{L}_{j,i} / N_i, \tag{11}$$

$$V_i = \max_{1 \leq j \leq M} \mathbf{L}_{j,i} / \sum_{j=1}^{M} \mathbf{L}_{j,i}. \tag{12}$$

Because the pixels at the border of the predicted mask area can probably be mapped to an object's surrounding points that do not belong to the object (for example, some points on the ceiling are predicted as points of a smoke alarm), these wrongly predicted points need to be removed. Unlike the points of an object, these neighbouring points do not appear in all images of the object. Moreover, some of them may only appear in one image, but are predicted as object points. Therefore, it is not enough to rely solely on prediction accuracy from all images. The value U_i is used to filter the surrounding points out and we illustrate how it works in Figure 11.

Figure 11a is a part of the point cloud that shows the ceiling and three kinds of objects (lighting, speaker, smoke alarm) mounted to it from the



(a) Bottom view of a ceiling and mounted objects (red box: lighting, yellow box: speaker, green box: smoke alarm).

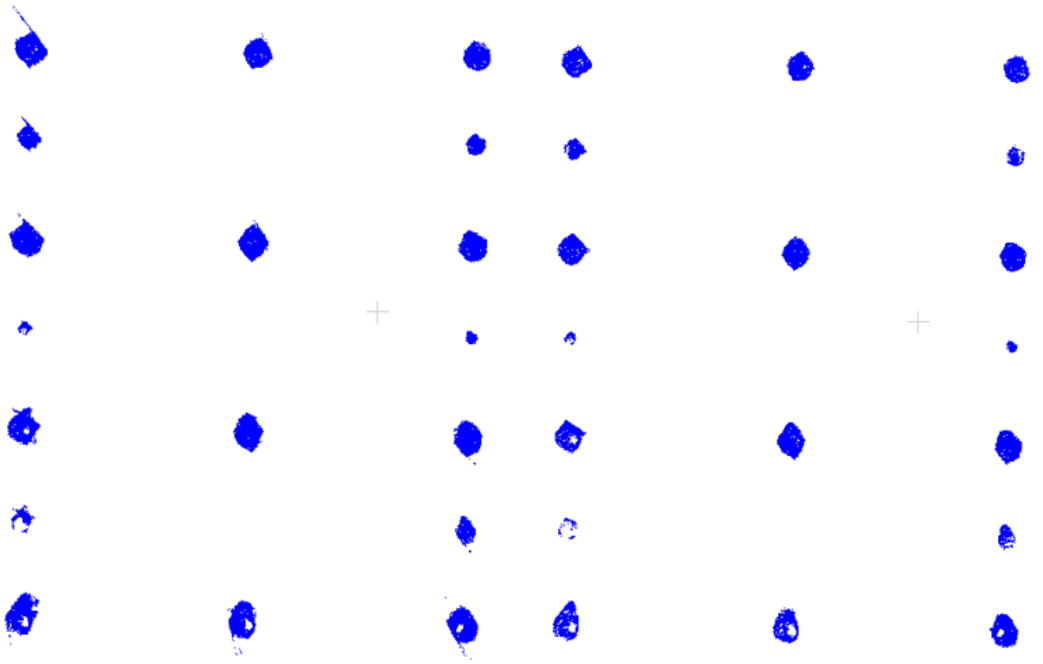
(b) Mapping prediction to point cloud and distribution of U_i

Figure 11: The distribution of U_i for a part of the point cloud of a ceiling

bottom view. Figure 11b shows the distribution of U_i. Many points on the ceiling are predicted as a point of the object because the prediction is mapped from 2D images that are taken from different views.

Most of the surrounding points (ceiling points) are distributed in the low-value range of U_i. Figure 12a and Figure 12b show the points left after filtering out those points with the criteria $U_i > 0.5$ and $U_i > 0.7$. Objects' points can be extracted from their neighbouring points on the ceiling.

Unlike U_i, which aims to remove surrounding points of an object, V_i is used to show how certain we are when assigning a class label with a point.



(a) Remaining point cloud after filtering $U_i > 0.5$ **(b)** Remaining point cloud after filtering $U_i > 0.7$

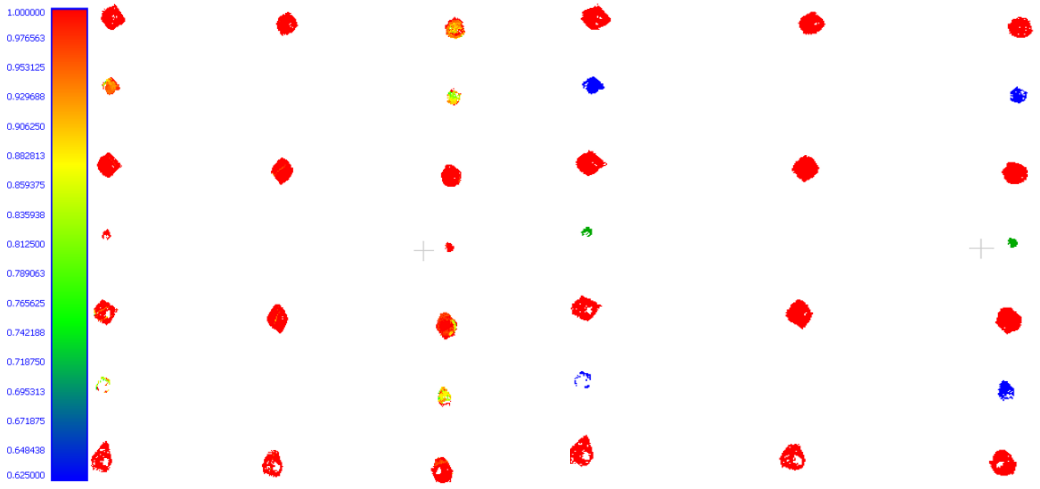
Figure 12: The remaining point cloud by filtering out ceiling points

Figure 13a shows the distribution of how certain we are when assigning the label that occurs mostly as the class of the point for the same area. In this case, it is quite certain that the assigned labels are correct as most points are located in the range close to 1. Figure 13a shows points in different colours according to their assigned labels.

3.9. Fit shape to each point cluster

In this step, we fit a geometric shape to each extracted point cluster. Different object types are reconstructed by varying strategies.

For small objects mounted on the ceiling and wall (like smoke alarms, sockets, switches), the extracted point clusters from the previous section are projected on the plane of the ceiling or wall. By then fitting simple geometric



(a) The distribution of V_i (assigning the corresponding label in V_i to the point)

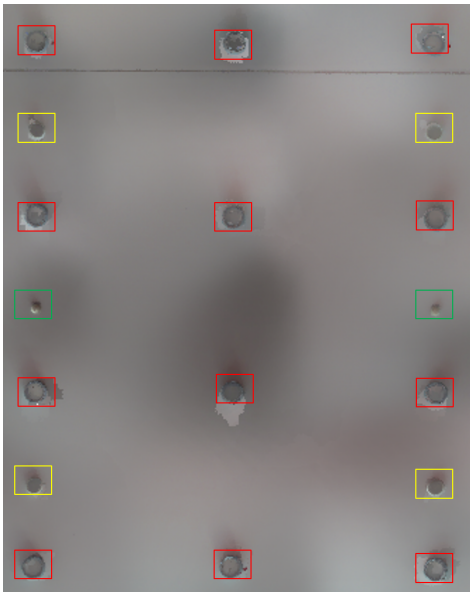
(b) Points of different classes (red: light, blue: speaker, green: smoke alarm)

Figure 13: The distribution of V_i and the extracted points of different classes

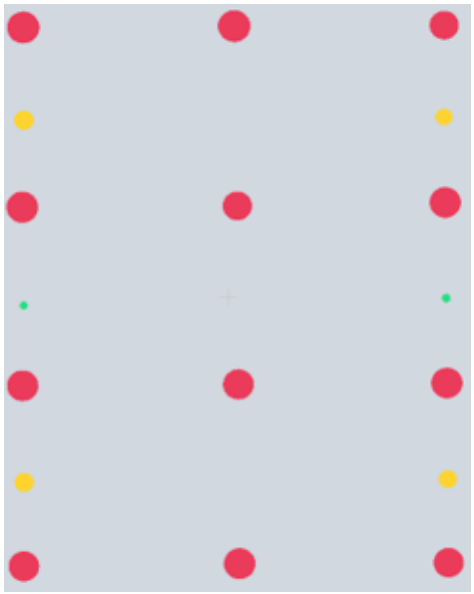
shapes (like circles and rectangles) in the wall or ceiling plane, the location and size in the 2D plane can be found. The reason we choose to fit geometric shapes in 2D planes rather than in 3D point clouds is: a) Some surfaces of the elements might not be captured when capturing buildings with a laser scanner. It is hard to fit geometric shapes in the 3D point cloud directly, especially for small elements (like smoke alarms) that lack points on their surface. b) Some elements are commonly standardised elements (sockets, light switches, smoke alarms) whose instances are identical across the entire facility. Fitting shapes in the 2D plane can also reduce the computing cost.

The random sample consensus (RANSAC) algorithm [55] is used to fit circles for cylindrical objects (such as a light, speaker, smoke alarm) and rectangles for "cuboid-like" objects (socket, switch, door sign, board, elevator button). We then extrude the 2D shapes from the wall or ceiling plane by default thickness (if available) or estimate the thickness of the object in the

3D point cluster by finding the maximum distance to the plane. The fitting circles of three classes of objects (light, speaker, smoke alarm) on the ceiling plane are shown in Figure 14 and corresponding extruded cylinders are shown in Figure 15 by way of example.



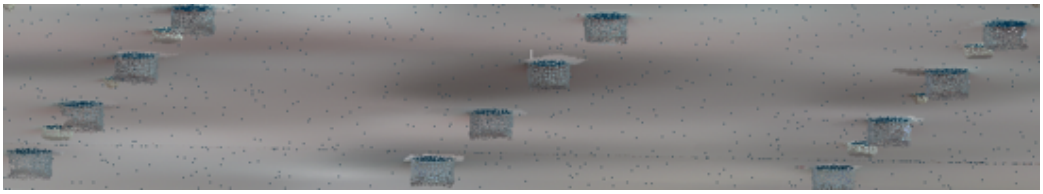
(a) Bottom view of part of ceiling (Red box: lighting, yellow box: speaker, green box: smoke alarm)



(b) Fitting result on ceiling plane (red: lighting, yellow: speaker, green: smoke alarm)

Figure 14: Bottom view of part of a ceiling and fitting result

With regard to pipes and fire extinguishers that are usually cylindrical, RANSAC is used to fit a cylinder to the point cluster and find its dimension and position. The extracted cylinder of a fire extinguisher is illustrated in Figure, 16 for example. As shown in Figure 16c, only one cylinder is reconstructed in this step, based on the major part of the fire extinguisher body. A more detailed structure of the fire extinguisher body and hose pipe would be ignored.



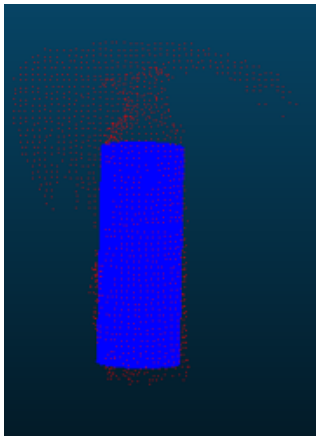
(a) Part of a ceiling in 3D space

(b) The fitting result in 3D space (red: lighting, yellow: speaker, green: smoke alarm)

Figure 15: Part of a ceiling and the fitting result in 3D space

(a) A fire extinguisher in point cloud

(b) Point cluster of the fire extinguisher



(c) Fit a cylinder to the cluster

Figure 16: Part of a ceiling and the fitting result in 3D space

3.10. Text detection and recognition

In this step, text information attached to objects is extracted from images. As shown in Figure 1, text information for facility management is available on or next to dedicated objects in a building, like the room number on a

door sign (shown in Figure 1a), the serial number on an emergency switch (shown in Figure 1b), the serial number next to a smoke alarm (shown in Figure 1c). Apart from detecting and recognising texts, the aim of this step is also to link the detected information to the corresponding objects.

With regard to text detection, text can be located in the object area as well as next to the object (like numbers next to the smoke alarm in Figure 1). No valid result could be found for the second case if detecting text only within the object area. In order to solve this problem, we enlarge the predicted object area by increasing its width and length by 50%, assuming related texts to the object are within the enlarged region. The text detection network model with differentiable binarization [36], pre-trained on [42], is applied within the enlarged area and outputs the corresponding text bounding boxes.

With regard to text recognition, the text recognition network model for irregular text [56] is applied to detected text bounding boxes. The recognised text is the information related to the corresponding object that contains or is close to the text area. The text detection and recognition result of a door sign and an emergency switch is illustrated in Figure 17. Most texts can be recognised correctly, especially those numbers that are very useful for building management.

Although the network we used is designed and trained to work with multi-oriented texts, the recognition result would suffer if texts were not horizontally-oriented. Non-horizontally-oriented texts usually occur in the images of the ceiling because it is hard to make sure the texts in all images are horizontally-oriented when holding a camera to collect images. In order to solve this problem, we inserted an intermediate step between text detection

(a) Text detection result on door sign

(b) Text recognition result on door sign

(c) Text detection result on emergency switch

(d) Text recognition result on emergency switch

Figure 17: Text detection and recognition result

and text recognition. In this step, the detected text bounding box would be rotated to the position where its longer side is horizontal by assuming texts are oriented along the longer side. Two angles (clockwise and counterclockwise) can rotate the bounding box to the horizontal position and produce two new bounding boxes. One of the angles would flip the text. The two new bounding boxes are then the input for the text recognition step. The

flipped texts can be discarded by the lower prediction score, and the results are shown in Section 4.2.3.

In summary, the input to the proposed processing pipeline are images/videos and point clouds. Point clusters with semantic information are created by mapping semantic information detected by deep learning to the 3D point cloud. The 3D mesh model is reconstructed by fitting geometric shapes to point clusters and then enriched by useful information that is valuable for maintaining the building by detecting and recognising text information on or close to objects.

4. Implementation and result

4.1. Implementation

The proposed processing pipeline is implemented in a software prototype written in C++ and Python and is tested in the point cloud collected in the Chair of Computational Modeling and Simulation at the Technical University of Munich (TUM) with the help of NAVVIS (www.navvis.com). The annotated dataset used for transfer learning contains more than 1000 instances, including 120 boards, 124 door signs, 34 elevator buttons, 52 emergency switches, 34 fire extinguishers, 30 escape signs, 357 lights, 94 light switches, 45 pipes, 137 smoke alarms, 123 sockets, and 91 speakers. These images are taken in different areas of the buildings in the city centre campus at TUM.

In point cloud processing, the PCL library [57] is used to implement the proposed algorithm. Object detection in images is done with Detectron2 [58]. In our experiment, we use the pre-trained Mask-RCNN model [21] provided by Facebook [58] that has been trained on the COCO dataset (more than

Technology	Language and library used	automatic or manual
Object detection in image by Transfer learning (Section 3.3)	Python, Detectron2 [58]	automatic
Creating photogrammetric point clouds (Section 3.4)	Python, COLMAP [49] [50]	automatic
Point clouds alignment (Section 3.5)	None	manual
Extract visible points (Section 3.6)	C++, PCL library [57]	automatic
Map 2D information to 3D space (Section 3.7)	C++	automatic
Find best-fitting labels (Section 3.8)	C++	automatic
Fit shape to point clusters (Section 3.9)	C++, PCL library [57]	automatic
Text detection and recognition (Section 3.10)	Python, MMOCR [59]	automatic

Table 1: Implementation details of each step

100k images) [32] and retrained on our annotated dataset. The photogrammetric point cloud is created by using COLMAP [49] [50]; text detection and recognition are implemented by means of the MMOCR tool [59]. The detailed implementation information, including the used technologies and frameworks, is listed in Table 1.

4.2. Results

In this section, we present the results of our experiments from three aspects, point cloud segmentation result, reconstruction result and, text recognition result. We use the mean Intersection over Union (mIoU), one of the common used evaluation metrics for semantic segmentation, to evaluate the performance of all 12 classes of small objects. Then we show the qualitative

Method	mIoU
PointNet [62]	47.6
SPG [63]	62.1
DGCNN [64]	56.1
RSNet [65]	56.5
PointCNN [66]	65.4
KPConv [61]	69.6
Point transformer [67]	73.5

Table 2: Segmentation mIoUs on S3DIS dataset (evaluated with 6-fold cross-validation)

result of the reconstructed model and evaluate the quantitative results of three classes (smoke alarm, light, speaker) in the facility. At last, we compare the text recognition result with and without the method proposed of rotating text boxes in Section 3.10.

4.2.1. Point cloud segmentation result

In our proposed pipeline, 2D semantic information detected from images is mapped to a 3D point cloud to identify the respective point clusters. The result is in the same format as that of point cloud segmentation of 3D deep learning. We compared the segmentation results of our proposed approach with those of 3D deep learning. In this regard, the S3DIS dataset [60] contains the point cloud of the indoor environment that is similar to the point cloud captured on the TUM campus. As shown in Table 2, KPConv [61] is one of the best-performing network architectures with the mIoU around 70%.

We choose KPConv for the experiments with the annotated laser-scanned point clouds captured at TUM and consider these are the reference values for further comparisons. We trained our model with two different downsam-

Model	wall	ceiling	floor	smoke alarm	light
KPConv (3cm)	89.0	96.5	97.6	29.1	69.4
KPConv (5cm)	88.2	96.2	97.8	18.6	65.2

Table 3: Segmentation mIoUs of related classes in our point cloud

pling sizes: 3cm and 5cm. As shown in Table 3, it is plain to see that the performance for large objects (wall, ceiling, floor) is much better than that for smaller objects. This result is consistent with that of the S3DIS dataset [60]. For a small object like a smoke alarm in particular, the performance is quite low, which means the current state-of-the-art network is not suitable for segmenting small objects. There are two possible explanations: a) the input point cloud resolution is too low for neural networks to understand small objects; b) small objects have much fewer points compared to larger ones (like a ceiling, floor, and wall), and the unequal class distribution means this has to be compensated during training, which could sacrifice the performance of some classes.

The performance of our proposed approach for different classes is shown in Table 4. As we can see, compared with the state-of-the-art network that only uses point clouds as input, our approach with additional image input provides a significant improvement in the common classes which are available in the image as well as the point cloud (smoke alarm from 29.1% to 48.6%, light from 69.4% to 79.9%).

4.2.2. Reconstruction result

One example of the information-rich digital twin that is created by applying our processing pipeline is illustrated in Figure 18. The digital twin is

board	door sign	elevator button	emergency switch	fire extinguisher	escape sign	light	light switch	pipes	smoke alarm	socket	speaker
68.0	67.0	80.8	62.2	85.7	70.1	79.9	47.6	39.1	48.6	61.1	64.5

Table 4: Segmentation mIoUs of small objects in our point cloud

a comprehensive model which includes geometric information (reconstructed 3D geometric models), semantic information (point clusters of object instances with labels and useful text information).

In Table 5, 6 and 7 we compare the dimension result for some objects in three categories from one area against the corresponding manually created model from the laser-scanned point cloud. As most of the absolute deviations of the radius are less than $0.01m$, the performance is quite good, given the resolution of the point cloud we used is $0.005m$. The relative deviations of smoke alarm diameters are relatively larger than those of the other two classes because the smoke alarms are smaller, which means an absolute deviation in a similar range results in a larger relative deviation value.

4.2.3. Text recognition result

In our experiments, the text recognition network model [56] works well if the text in an image is horizontally oriented and performs worse if the text is not horizontal. The comparison of recognition results for texts attached to two objects is shown in Figure 19.

In order to improve the recognition result, we introduce a method of rotating the detected bounding boxes in Section 3.10. The corresponding

No.	radius	ground truth	deviation (abs.)	deviation (rel.%)
1	0.116	0.110	0.006	5.5
2	0.110	0.110	0	0
3	0.118	0.110	0.008	7.3
4	0.110	0.110	0	0
5	0.118	0.110	0.008	7.3
6	0.121	0.110	0.011	10.0
7	0.116	0.110	0.006	5.5
8	0.117	0.110	0.007	6.4
9	0.118	0.110	0.008	7.3
10	0.117	0.110	0.007	6.4
11	0.121	0.110	0.011	10.0
12	0.113	0.110	0.003	2.7

Table 5: Light radius comparison between model created from our approach and manually created model: (m)

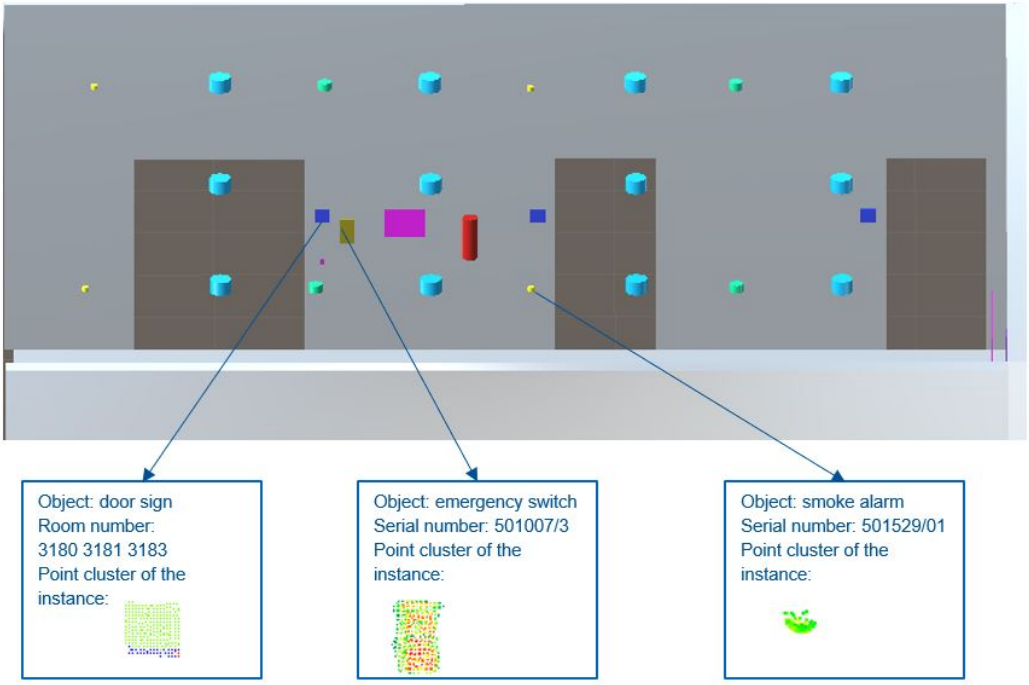
No.	radius	ground truth	deviation (abs.)	deviation (rel.%)
1	0.072	0.070	0.002	2.9
2	0.063	0.070	0.007	10.0
3	0.068	0.070	0.002	2.9
4	0.073	0.070	0.003	4.3

Table 6: Speaker radius comparison between model created from our approach and manually created model: (m)

No.	radius	ground truth	deviation (abs.)	deviation (rel.%)
1	0.030	0.035	0.005	14.3
2	0.032	0.035	0.003	8.6
3	0.025	0.035	0.010	28.6
4	0.028	0.035	0.007	20.0
5	0.027	0.035	0.008	22.9

Table 7: Smoke alarm radius comparison between model created from our approach and manually created model: (m)

(a) Input point cloud (ceiling removed for visualisation)

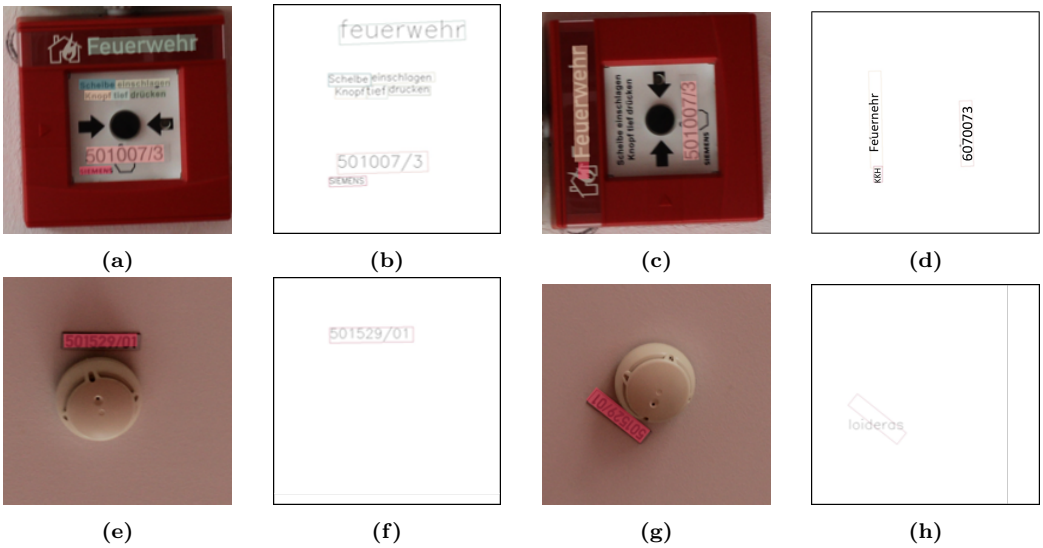


(b) The created information-rich building twin

Figure 18: Input point cloud and the created elements of the building twin

result is shown in Figure 20, for example.

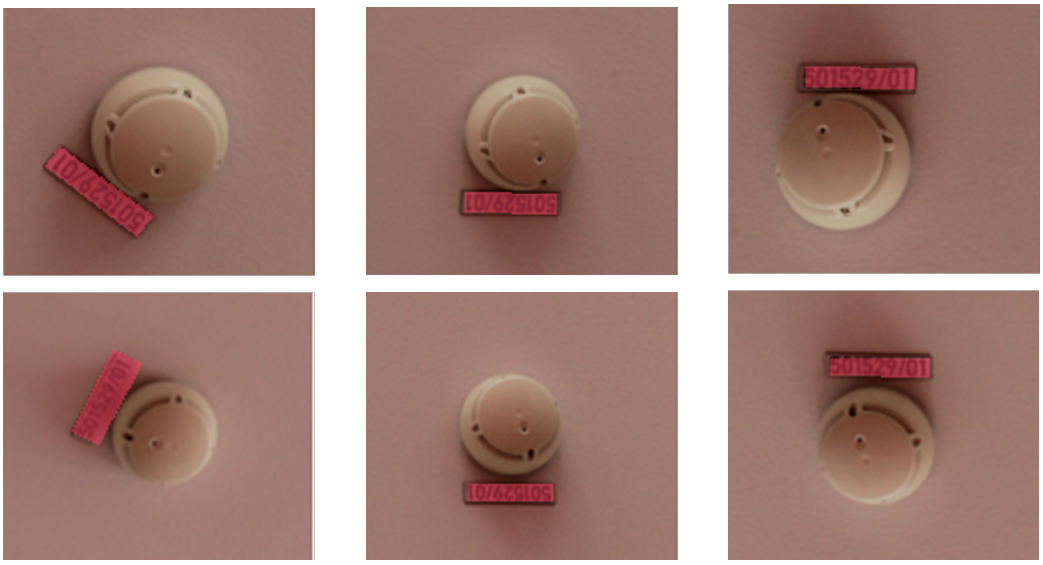
In order to discard the prediction of flipped texts, prediction scores are checked. The recognised texts and corresponding prediction score of four



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 19: Comparison of recognition results between non- and horizontally-oriented text



(a) Two images of the same object

(b) Counterclockwise rotation to the horizontal position

(c) Clockwise rotation to the horizontal position

Figure 20: Rotating the detected box to a horizontal position

Image Nr.	Text	Score
1	501529/01	**0.99995**
2	LO/SEZSLOS	0.78154
3	501529/01	**0.99824**
4	LO/62SLOS	0.84252

Table 8: Recognised text and prediction score

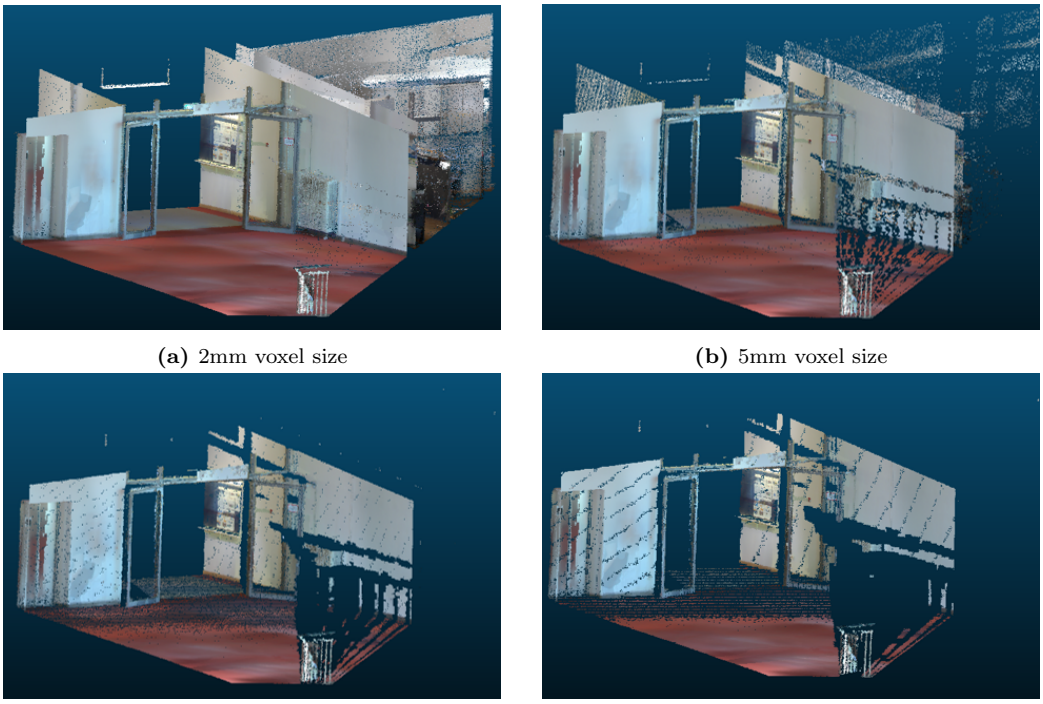
horizontal bounding boxes in Figure 20 are listed in the Table 8. It is plain to see that two prediction scores (Nr.2 and Nr.4) are significantly lower than the other two (Nr.1 and Nr.3), which means the level of certainty is lower. And this lower prediction score comes from the flipped text. Therefore, it is very easy to identify the correct direction of text by analysing the prediction score. The texts from high score predictions are then chosen as the extracted text information if these predictions provide identical results (as in Table 8, where they both predict "501529/01"). If high score predictions are in conflict with each other, which usually happens when multiple images for the same object are available, all predicted texts are stored with their prediction scores. So the final decision is left up to the human user.

4.3. Parameter study

In Section 3.6, we use the ray-casting method to remove points that should not be visible at the given camera position. The aim of ray-casting is to make points visible in the real world that can also be seen in the point cloud. At the same time, it should not "look through" the wall either, seeing points that should be occluded. Therefore, the voxel size in Figure 8 is essential.

Figure 21 shows a comparison of four different voxel size: 2mm, 5mm, 1cm, 2cm. As we can see, rays can still go through the wall with a resolution

of 2mm and 5mm, which makes the scene behind the wall visible. With a resolution of 2cm, the handrail and its fence cause too much occlusion, making a relatively large part of the wall that should not be occluded invisible. In this case, the voxel size of 1cm provides the best result. Moreover, the test point cloud resolution is also 1cm in Figure 21. This is not a coincidence, because a 1cm resolution point cloud means the distance between neighbouring points is around 1cm. Therefore, it is appropriate that the voxel size chosen for ray-casting is the same as the resolution of a point cloud, so that rays do not pass through a surface and at the same time avoid unnecessary occlusions.



(a) 2mm voxel size

(b) 5mm voxel size

(c) 1cm voxel size

(d) 2cm voxel size

Figure 21: Ray-casting result with different voxel sizes

4.4. Discussion

As shown in Section 4.2, the proposed pipeline provides convincing results in creating geometric digital twins of buildings from laser-scanned point clouds and images. Meanwhile, the method could be applied to other facilities if the environment is captured by a laser scanner and a camera. However, it should be noted that the photogrammetric process only works if a sufficient amount of images were taken differently from different viewpoints. It is hard to say a minimum required number of images for the photogrammetric process because it depends on different aspects, such as the facility size, number of objects, the camera lens, etc. But according to the authors' experience, more images from various viewpoints usually improves the reconstruction result.

In addition, we also test the photogrammetric process with images and frames extracted from videos. In our experiment, photogrammetric point clouds created by video frames are usually noisier than those from camera images. Furthermore, a camera with a higher resolution and larger field of view can also contribute to a higher-quality point cloud, which usually requires a longer computation time. As the photogrammetric process is only used to register images to laser-scanned point clouds, the strategies of increasing the quality of photogrammetric point clouds and reducing the cost are not in the scope of this paper.

If the photogrammetric process in the pipeline fails, all the other parts can proceed as the same. But an alternative way to provide a camera's intrinsic and extrinsic parameters should be included, for example, using the referenced images taken by modern laser scanners that have cameras during

data capturing, manually recording camera poses and calibrating parameters.

Furthermore, there are still other limitations to our methods. Firstly, the object detection step can provide good results for standard objects like fire extinguishers, smoke alarms, etc. But it performs worse with objects that vary greatly in different environments, such as lights on the ceiling. More training pictures are required to solve this problem. Secondly, although we have already enlarged the number of reconstructed categories in the indoor environment, many other objects are still missing, such as desks, bookshelf, etc. These elements are also valuable in an information-rich building twin.

5. Conclusion

In conclusion, we propose a novel pipeline to enrich the geometric digital twin of buildings with small objects along with useful text information. It can be used to enrich and complete as-built models generated by other methods of creating digital twins. The contributions of the paper are as follows:

a) Unlike most previous work that used only laser scanning or photogrammetric technologies, we fuse both to enhance information input. Semantic information detected by deep learning in image recognition is mapped into a 3D point cloud to obtain point clusters of different classes;

b) We put emphasis on the object classes in building twins that represent electrical elements (light switch, light, speaker, socket, elevator button), safety elements (emergency switch, smoke alarm, fire extinguisher, escape sign), plumbing system elements (pipe), and other objects with useful information for facility management (door sign and boards);

c) Apart from geometric and semantic information, we apply text detec-

tion and recognition technology to extract useful text information such as serial numbers and object IDs for related objects;

d) The whole processing pipeline is almost completely automated. The only step that requires manual work is registering the photogrammetric and laser-scanned point cloud, which can be easily achieved by off-the-shelve software products.

In future, we want to collect more data and continue adding more classes (like furniture) to the building twin. While we only fit simple geometric shapes (like a cylinder) to the extracted point clusters at present, more complex shapes or CAD models can be considered as a potential improvement for the building twin. Furthermore, we would also combine 3D deep learning in the point cloud and 2D deep learning in images in one framework that can probably improve the segmentation performance.

6. Acknowledgements

The work in this paper was funded by the Institute for Advanced Study (IAS) at the Technical University of Munich. The dataset we used in this paper was collected on the main campus of the Technical University of Munich with the help of NAVVIS (https://www.navvis.com/). In addition, we would like to thank NVIDIA Applied Research Accelerator Program for their support by providing high-performance hardware.

Appendix A. Appendix

References

[1] I. Brilakis, Y. Pan, A. Borrmann, H.-G. Mayer, F. Rhein, C. Vos, E. Pettinato, S. Wagner, Built environment digital twining (2019). URL https://publications.cms.bgu.tum.de/reports/2020_Brilakis_BuiltEnvDT.pdf,l access April 04, 2022

[2] E. Agapaki, I. Brilakis, Instance segmentation of industrial point cloud data, Journal of Computing in Civil Engineering 35 (6) (2021) 04021022. doi:https://doi.org/10.1061/(ASCE)CP.1943-5487.0000972.

[3] Y. Pan, A. Braun, A. Borrmann, I. Brilakis, Void-growing: a novel scan-to-BIM method for manhattan world buildings from point cloud, in: Proceedings of the 2021 European Conference on Computing in Construction, Vol. 2 of Computing in Construction, University College Dublin, Online, 2021, pp. 312–321. doi:https://doi.org/10.35490/ec3.2021.162.

[4] H. Tran, K. Khoshelham, A. Kealy, L. Díaz-Vilariño, Shape grammar approach to 3d modeling of indoor environments using point clouds, Journal of Computing in Civil Engineering 33 (1) (2019) 04018055. doi:https://doi.org/10.1061/(ASCE)CP.1943-5487.0000800.

[5] S. Ochmann, R. Vock, R. Klein, Automatic reconstruction of fully volumetric 3d building models from oriented point clouds, ISPRS journal of photogrammetry and remote sensing 151 (2019) 251–262. doi:https://doi.org/10.1016/j.isprsjprs.2019.03.017.

[6] A. Adán, B. Quintana, S. A. Prieto, F. Bosché, Scan-to-bim for 'secondary' building components, Advanced Engineering Informatics 37 (2018) 119–138. doi:https://doi.org/10.1016/j.aei.2018.05.001.

[7] C. D'Urso, Information integration for facility management, IT Professional 13 (6) (2011) 48–53. doi:https://doi.org/10.1109/MITP.2011.100.

[8] W. Meeussen, M. Wise, S. Glaser, S. Chitta, C. McGann, P. Mihelich, E. Marder-Eppstein, M. Muja, V. Eruhimov, T. Foote, et al., Autonomous door opening and plugging in with a personal robot, in: 2010 IEEE International Conference on Robotics and Automation, IEEE, 2010, pp. 729–736. doi:https://doi.org/10.1109/ROBOT.2010.5509556.

[9] U. Krispel, H. L. Evers, M. Tamke, R. Viehauser, D. Fellner, Automatic texture and orthophoto generation from registered panoramic views, The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 40 (5) (2015) 131. doi:https://doi.org/10.5194/isprsarchives-XL-5-W4-131-2015.

[10] J.-G. Kang, S.-Y. An, W.-S. Choi, S.-Y. Oh, Recognition and path planning strategy for autonomous navigation in the elevator environment, International Journal of Control, Automation and Systems 8 (4) (2010) 808–821. doi:https://doi.org/10.1007/s12555-010-0413-3.

[11] P. Kim, J. Chen, Y. K. Cho, Building element recognition with thermal-mapped point clouds, in: Proceedings of the 34th International Symposium on Automation and Robotics in Construction, 2017, pp. 872–878. doi:10.22260/ISARC2017/0122.

[12] P. Kim, J. Chen, Y. K. Cho, Robotic sensing and object recognition from thermal-mapped point clouds, International Journal of Intelligent Robotics and Applications 1 (3) (2017) 243–254. doi:https://doi.org/10.1007/s41315-017-0023-9.

[13] L. Díaz-Vilariño, H. González-Jorge, J. Martínez-Sánchez, H. Lorenzo, Automatic lidar-based lighting inventory in buildings, Measurement 73 (2015) 544–550. doi:https://doi.org/10.1016/j.measurement.2015.06.009.

[14] I. Puente, H. González-Jorge, J. Martínez-Sánchez, P. Arias, Automatic detection of road tunnel luminaires using a mobile lidar system, Measurement 47 (2014) 569–575. doi:https://doi.org/10.1016/j.measurement.2013.09.044.

[15] T. Czerniawski, M. Nahangi, C. Haas, S. Walbridge, Pipe spool recognition in cluttered point clouds using a curvature-based shape descriptor, Automation in Construction 71 (2016) 346–358. doi:https://doi.org/10.1016/j.autcon.2016.08.011.

[16] T. Czerniawski, F. Leite, Automated segmentation of rgb-d images into a comprehensive set of building components using deep learning, Advanced Engineering Informatics 45 (2020) 101131. doi:https://doi.org/10.1016/j.aei.2020.101131.
URL https://www.sciencedirect.com/science/article/pii/S1474034620301026

[17] E. Agapaki, I. Brilakis, Cloi-net: Class segmentation of industrial facil-

ities' point cloud datasets, Advanced Engineering Informatics 45 (2020) 101121. doi:https://doi.org/10.1016/j.aei.2020.101121.

[18] C. Szegedy, A. Toshev, D. Erhan, Deep neural networks for object detection, in: C. Burges, L. Bottou, M. Welling, Z. Ghahramani, K. Weinberger (Eds.), Advances in Neural Information Processing Systems, Vol. 26, Curran Associates, Inc., 2013.
URL https://proceedings.neurips.cc/paper/2013/file/f7cade80b7cc92b991cf4d2806
last access April 04, 2022

[19] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 580–587. doi:https://doi.org/10.1109/CVPR.2014.81.

[20] R. Girshick, Fast r-cnn, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 1440–1448. doi:https://doi.org/10.1109/ICCV.2015.169.

[21] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2961–2969. doi:https://doi.org/10.1109/ICCV.2017.322.

[22] Y. Jiang, D. Pang, C. Li, A deep learning approach for fast detection and classification of concrete damage, Automation in Construction 128 (2021) 103785. doi:https://doi.org/10.1016/j.autcon.2021.103785.

[23] Y. Tan, R. Cai, J. Li, P. Chen, M. Wang, Automatic detection of sewer defects based on improved you only look

once algorithm, Automation in Construction 131 (2021) 103912. doi:https://doi.org/10.1016/j.autcon.2021.103912.

[24] Y. Wu, Y. Qin, Y. Qian, F. Guo, Automatic detection of arbitrarily oriented fastener defect in high-speed railway, Automation in Construction 131 (2021) 103913. doi:https://doi.org/10.1016/j.autcon.2021.103913.

[25] I. Jeelani, K. Asadi, H. Ramshankar, K. Han, A. Albert, Real-time vision-based worker localization & hazard detection for construction, Automation in Construction 121 (2021) 103448. doi:https://doi.org/10.1016/j.autcon.2020.103448.

[26] H. Son, C. Kim, Integrated worker detection and tracking for the safe operation of construction machinery, Automation in Construction 126 (2021) 103670.
URL https://doi.org/10.1016/j.autcon.2021.103670

[27] N. D. Nath, A. H. Behzadan, S. G. Paal, Deep learning for site safety: Real-time detection of personal protective equipment, Automation in Construction 112 (2020) 103085. doi:https://doi.org/10.1016/j.autcon.2021.103670.

[28] Z. Kolar, H. Chen, X. Luo, Transfer learning and deep convolutional neural networks for safety guardrail detection in 2d images, Automation in Construction 89 (2018) 58–70. doi:https://doi.org/10.1016/j.autcon.2018.01.003.

[29] S. J. Pan, Q. Yang, A survey on transfer learning, IEEE Transactions

on knowledge and data engineering 22 (10) (2009) 1345–1359.
URL https://doi.org/10.1109/TKDE.2009.191

[30] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE conference on computer vision and pattern recognition, Ieee, 2009, pp. 248–255. doi:https://doi.org/10.1109/CVPR.2009.5206848.

[31] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, A. Zisserman, The pascal visual object classes (voc) challenge, International journal of computer vision 88 (2) (2010) 303–338. doi:https://doi.org/10.1007/s11263-009-0275-4.

[32] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C. L. Zitnick, Microsoft coco: Common objects in context, in: European conference on computer vision, Springer, 2014, pp. 740–755. doi:https://doi.org/10.1007/978-3-319-10602-1_48.

[33] S. Long, J. Ruan, W. Zhang, X. He, W. Wu, C. Yao, Textsnake: A flexible representation for detecting text of arbitrary shapes, in: European conference on computer vision, 2018, pp. 20–36. doi:https://doi.org/10.48550/arXiv.1807.01544.

[34] W. Wang, E. Xie, X. Song, Y. Zang, W. Wang, T. Lu, G. Yu, C. Shen, Efficient and accurate arbitrary-shaped text detection with pixel aggregation network, in: IEEE/CVF International Conference on Computer Vision, 2019, pp. 8439–8448. doi:https://doi.org/10.48550/arXiv.1908.05900.

[35] W. Wang, E. Xie, X. Li, W. Hou, T. Lu, G. Yu, S. Shao, Shape robust text detection with progressive scale expansion network, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 9336–9345. doi:https://doi.org/10.48550/arXiv.1903.12473.

[36] M. Liao, Z. Wan, C. Yao, K. Chen, X. Bai, Real-time scene text detection with differentiable binarization, Proceedings of the AAAI Conference on Artificial Intelligence (2020) 11474–11481doi:https://doi.org/10.48550/arXiv.1911.08947.

[37] Y. Zhu, J. Chen, L. Liang, Z. Kuang, L. Jin, W. Zhang, Fourier contour embedding for arbitrary-shaped text detection, in: IEEE conference on computer vision and pattern recognition, 2021. doi:https://doi.org/10.48550/arXiv.2104.10442.

[38] B. Shi, X. Bai, C. Yao, An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 39 (11) (2017) 2298–2304. doi:https://doi.org/10.1109/TPAMI.2016.2646371.

[39] H. Li, P. Wang, C. Shen, G. Zhang, Show, attend and read: A simple and strong baseline for irregular text recognition, Proceedings of the AAAI Conference on Artificial Intelligence 33 (01) (2019) 8610–8617. doi:https://doi.org/10.1609/aaai.v33i01.33018610.

[40] F. Sheng, Z. Chen, B. Xu, Nrtr: A no-recurrence sequence-to-

sequence model for scene text recognition, in: 2019 International Conference on Document Analysis and Recognition (ICDAR), IEEE Computer Society, Los Alamitos, CA, USA, 2019, pp. 781–786. doi:https://doi.org/10.1109/ICDAR.2019.00130.

[41] X. Yue, Z. Kuang, C. Lin, H. Sun, W. Zhang, Robustscanner: Dynamically enhancing positional clues for robust text recognition, in: Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIX, Springer-Verlag, Berlin, Heidelberg, 2020, p. 135–151. doi:https://doi.org/10.1007/978-3-030-58529-7_9.

[42] A. Gupta, A. Vedaldi, A. Zisserman, Synthetic data for text localisation in natural images, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2315–2324. doi:https://doi.org/10.1109/CVPR.2016.254.

[43] A. Veit, T. Matera, L. Neumann, J. Matas, S. Belongie, Coco-text: Dataset and benchmark for text detection and recognition in natural images, arXiv preprint arXiv:1601.07140 (2016). doi:https://doi.org/10.48550/arXiv.1601.07140.

[44] Q. Lu, L. Chen, S. Li, M. Pitt, Semi-automatic geometric digital twinning for existing buildings based on images and cad drawings, Automation in Construction 115 (2020) 103183. doi:https://doi.org/10.1016/j.autcon.2020.103183.

[45] Y. Zhao, X. Deng, H. Lai, Reconstructing bim from 2d structural

drawings for existing buildings, Automation in Construction 128 (2021) 103750. doi:https://doi.org/10.1016/j.autcon.2021.103750.

[46] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, Advances in neural information processing systems 25 (2012) 1097–1105.
URL https://papers.nips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-A
last access April 14 2022

[47] Z.-Q. Zhao, P. Zheng, S.-t. Xu, X. Wu, Object detection with deep learning: A review, IEEE transactions on neural networks and learning systems 30 (11) (2019) 3212–3232. doi:https://doi.org/10.48550/arXiv.1807.05511.

[48] A. Braun, A. Borrmann, Combining inverse photogrammetry and bim for automated labeling of construction site images for machine learning, Automation in Construction 106 (2019) 1–13. doi:https://doi.org/10.1016/j.autcon.2019.102879.

[49] J. L. Schönberger, E. Zheng, M. Pollefeys, J.-M. Frahm, Pixelwise view selection for unstructured multi-view stereo, in: European Conference on Computer Vision (ECCV), 2016. doi:https://doi.org/10.1007/978-3-319-46487-9_31.

[50] J. L. Schönberger, J.-M. Frahm, Structure-from-motion revisited, in: Conference on Computer Vision and Pattern Recognition (CVPR), 2016. doi:https://doi.org/10.1109/CVPR.2016.445.

[51] T. Oguchi, S. H. Yuichi, T. Wasklewicz, Chapter seven - data sources, in: M. J. Smith, P. Paron, J. S. Griffiths (Eds.), Geomorphological Mapping, Vol. 15 of Developments in Earth Surface Processes, Elsevier, 2011, pp. 189–224. doi:https://doi.org/10.1016/B978-0-444-53446-0.00007-0.
URL https://www.sciencedirect.com/science/article/pii/B9780444534460000070

[52] P. J. Besl, N. D. McKay, Method for registration of 3-d shapes, in: Sensor fusion IV: control paradigms and data structures, Vol. 1611, International Society for Optics and Photonics, 1992, pp. 586–606. doi:https://doi.org/10.1109/34.121791.

[53] S. D. Roth, Ray casting for modeling solids, Computer graphics and image processing 18 (2) (1982) 109–144. doi:https://doi.org/10.1016/0146-664X(82)90169-1.

[54] S. Laine, T. Karras, Efficient sparse voxel octrees–analysis, extensions, and implementation, Tech. rep., NVIDIA Corporation (2010).
URL https://www.nvidia.com/docs/IO/88972/nvr-2010-001.pdf, last access April 17, 2022

[55] M. A. Fischler, R. C. Bolles, Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, Communications of the ACM 24 (6) (1981) 381–395. doi:https://doi.org/10.1145/358669.358692.

[56] H. Li, P. Wang, C. Shen, G. Zhang, Show, attend and read: A simple and strong baseline for irregular text recognition, in: Proceedings of the

AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 8610–8617. doi:https://doi.org/10.1609/aaai.v33i01.33018610.

[57] R. B. Rusu, S. Cousins, 3d is here: Point cloud library (pcl), in: IEEE International Conference on Robotics and Automation (ICRA), IEEE, Shanghai, China, 2011. doi:https://doi.org/10.1109/ICRA.2011.5980567.

[58] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, R. Girshick, Detectron2, `https://github.com/facebookresearch/detectron2`, last access April 17, 2022 (2019).

[59] Z. Kuang, H. Sun, Z. Li, X. Yue, T. H. Lin, J. Chen, H. Wei, Y. Zhu, T. Gao, W. Zhang, K. Chen, W. Zhang, D. Lin, Mmocr: A comprehensive toolbox for text detection, recognition and understanding, arXiv preprint arXiv:2108.06543 (2021). doi:https://doi.org/10.48550/arXiv.2108.06543.

[60] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, S. Savarese, 3d semantic parsing of large-scale indoor spaces, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1534–1543. doi:https://doi.org/10.1109/CVPR.2016.170.

[61] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, L. J. Guibas, Kpconv: Flexible and deformable convolution for point clouds, in: Proceedings of the IEEE/CVF Interna-

tional Conference on Computer Vision, 2019, pp. 6411–6420. doi:https://doi.org/10.48550/arXiv.1904.08889.

[62] C. R. Qi, H. Su, K. Mo, L. J. Guibas, Pointnet: Deep learning on point sets for 3d classification and segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 652–660. doi:https://doi.org/10.48550/arXiv.1612.00593.

[63] L. Landrieu, M. Simonovsky, Large-scale point cloud semantic segmentation with superpoint graphs, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 4558–4567. doi:https://doi.org/10.1109/CVPR.2018.00479.

[64] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, J. M. Solomon, Dynamic graph cnn for learning on point clouds, Acm Transactions On Graphics (tog) 38 (5) (2019) 1–12. doi:https://doi.org/10.1145/3326362.

[65] Q. Huang, W. Wang, U. Neumann, Recurrent slice networks for 3d segmentation of point clouds, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 2626–2635. doi:https://doi.org/10.48550/arXiv.1802.04402.

[66] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, B. Chen, Pointcnn: Convolution on x-transformed points, Advances in neural information processing systems 31 (2018) 820–830. doi:https://doi.org/10.48550/arXiv.1801.07791.

[67] H. Zhao, L. Jiang, J. Jia, P. H. Torr, V. Koltun, Point transformer, in: Proceedings of the IEEE/CVF Interna-

tional Conference on Computer Vision, 2021, pp. 16259–16268. doi:https://doi.org/10.48550/arXiv.2012.09164.