Classification of Indoor Point Clouds Using Multiviews

Vladeta Stojanovic
Hasso Plattner Institute
University of Potsdam, Germany
vladeta.stojanovic@hpi.de

Matthias Trapp
Hasso Plattner Institute
University of Potsdam, Germany
matthias.trapp@hpi.de

Rico Richter
Hasso Plattner Institute
University of Potsdam, Germany
rico.richter@hpi.de

Jürgen Döllner
Hasso Plattner Institute
University of Potsdam, Germany
juergen.doellner@hpi.de

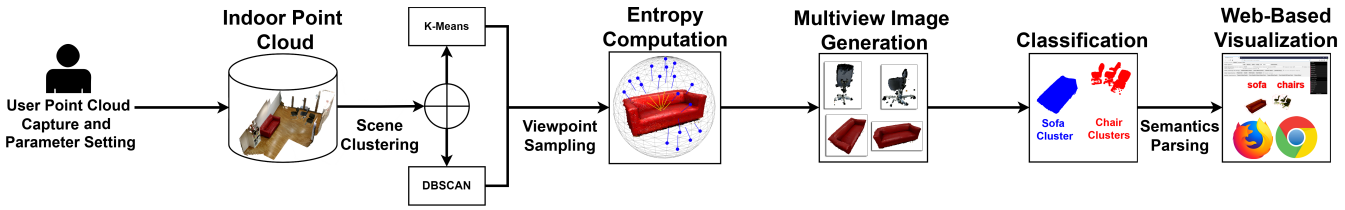


Figure 1: Classification of indoor point clouds, using the viewpoint entropy of a virtual perspective camera view of a 3D point cluster. The entropy is obtained as a vector parallelism measure between the randomly sampled cluster bounding sphere vertices (used as virtual camera positions and directions), and point cluster normal vectors. This approach allows for semantic-enrichment of indoor point cloud scenes, implemented as a light-weight service-oriented Web3D software component.

ABSTRACT

We present an approach for classifying 3D point clouds of indoor environments using Convolutional Neural Network (CNN)-based image analysis for entropy-selected 3D views. Prior to classification, the 3D point clouds are clustered using either Density-Based Spatial Clustering of Applications with Noise (DBSCAN) or *k*-Means algorithms. We then use randomly sampled 3D point normal vectors as an entropy descriptor to calculate the direction and position of the virtual camera, which is placed around these clusters. It synthesizes 2D images of a given cluster from multiple views with positions and directions that have highest visual entropy. We then proceed to classify the images using a retrained CNN. We test our approach for classifying common office furniture items using both synthetic and actual 3D point clouds of typical indoor office spaces. The empirical results demonstrate that our approach is suited towards classifying specific indoor furniture objects. We also describe how our approach can be implemented as a lightweight component within a service-oriented system that is used for visualization and classification of 3D point clouds using a given Web3D tool. The resulting semantically-enriched 3D point clouds can further be used for digital indoor environment representations, with further use as base data for Building Information Model (BIM), Facility Management (FM), and Digital Twin (DT) applications.

CCS CONCEPTS

• **Computing methodologies** → *Computer graphics*; **Visibility**; *Neural networks*; • **Human-centered computing** → *Visualization systems and tools*;

KEYWORDS

Viewpoint Entropy, Multiview Classification, Indoor Point Clouds, Semantic Enrichment

1 INTRODUCTION

Classification of indoor 3D point clouds is becoming increasingly important as photogrammetry and laser-based scanning acquisition technologies are rapidly adopted to capture the physical state of all kinds of *as-built* environments. In particular, 3D point clouds can represent the physical features of indoor environments (e.g., room representations, furniture objects, machinery, and any other static features of the operational areas) at varying Level-of-Detail (LOD). Insofar, indoor 3D point clouds serve as source data for *as-is* BIM generation [Teicholz et al. 2013], and provide a basis for DT representations [Stojanovic et al. 2018b]. While 3D point clouds may feature either RGB or intensity color-mapped representations, because they are mainly interpreted visually by users, the semantics are not explicitly contained in the data. Manually segmenting 3D point clouds and assigning semantics tends to be a time consuming, error prone, and costly process. In contrast, our approach classifies 3D point clouds of indoor environments using a retrained CNN based on image analysis for entropy-selected 3D views that allows for object detection and adding semantics to 3D point clouds used, e.g., for further reconstruction and representation tasks (Fig. 1). It aims to enable flexible and fast classification on indoor point clouds within the context of Web3D, utilizing lightweight service-oriented web-application components for routine visualization, classification, and semantic enrichment of such data. We have implemented the presented cluster, viewpoint entropy selection, and point cloud visualization methods using Three.js [Cabello et al. 2010]. However,

our approach is adaptable for integration and extension of most Web3D-based tools, specifically those that can support 3D point cloud visualization.

Motivation. Image-based classification relies on using a CNN model to classify image data, based on given training data used as examples for classification categories. Since 3D point clouds are predominantly captured from physical *as-built* built environments [Qu et al. 2014], a CNNs can be trained using photographs of specific indoor features or objects (e.g., furniture, machinery, or structural features), and used to classify 2D images synthesized from 3D point clouds. These photographs may feature items in various positions and projections. If specific objects are required to be classified, then images containing optimal views of given objects are best suited for training. The CNN trained to classify specific images can be applied to classify 2D images of 3D point clusters containing expected items. However, the more items are partially occluded, the more the correct classification of item types decreases. To improve the classification approach, multiple 2D images of a given 3D point cluster need to be classified. By using optimal viewpoints for taking images of 3D point clusters, we can reduce the amount of time and data required for classification, while increasing the overall classification accuracy of specific item clusters in the given scene.

Research Contributions. Our approach automatically classifies indoor 3D point clouds, based on classifying automatically determined multiple 2D images synthesized from 3D point clusters. The views are selected such that the most useful amount of visual information, i.e., those with highest entropy, is shown [Castelló et al. 2006]. Our method to select viewpoints is based on an entropy measure, i.e., a measure of normal vector parallelism. Randomly sampled inverted normal vectors from the bounding sphere of a given 3D point cluster are taken as camera positions and directions based on their parallelism in terms of direction to the pre-computed 3D point cluster normal vectors. The 2D images of 3D point clusters with optimal viewpoints are then synthesized and classified, and the classification results are averaged for each cluster, and streamed back into the 3D point cloud for semantic enrichment. In particular, we focus on detecting common office furniture items (e.g., chairs, sofas and tables); the corresponding 3D point clouds are generated by commodity mobile scanning devices applying photogrammetric methods for 3D point cloud acquisition. We also evaluate the use of k-means and DBSCAN for clustering of points and discuss their advantages, comparative performance and limitations for specific clustering tasks of indoor 3D point clouds.

We also discuss how this approach can be implemented as a data processing service for 3D point clouds. Experimental results are presented regarding the classification approach using different clustering methods. We have tested our approach using synthetic datasets, and real-world indoor 3D point clouds using the Stanford dataset [Armeni et al. 2017]. We compare the classification correctness and limitations for classifying *as-is* 3D point clouds with RGB information containing common office furniture. Finally, the classification accuracy of our approach is compared against another similar multiview classification method described by Stojanovic et al. [2018a].

2 RELATED WORK

Semantic Enrichment of 3D Point Clouds. Two approaches for semantic enrichment of 3D point clouds are presented by Armeni et al. [2016] and Armeni et al. [2017]. The first paper proposes a method where raw 3D point clouds are parsed into 3D space volumes, which are aligned into a canonical reference coordinate system. This approach relies on assuming the structural similarity of disjoint indoor spaces—taking into account the fairly detectable spatial histogram signature of void spaces between representative 3D point clusters of rooms. Once the rooms have been detected, other features such as furniture can be detected by using a sliding box representation to capture and inspect any remaining point clusters using voxelized elements of the sliding box for geometric classification. The second approach uses joint 2D and 3D scene data for semantics generation. This includes the 2D RGB images of the scene, 2.5D images including depth and surface normal vectors, and the 3D point cloud and reconstructed meshes. The semantics for a 3D point cloud are generated using the same approach as in the first paper, but these semantics are then aligned with all of the associated scene data types.

Recent research by Runceanu and Haala [2018] has focused on semantic enrichment of indoor 3D point clouds for BIM reconstruction, making use of region growing segmentation based on point normal vector and color similarity. Segmented regions allow for training a Random Forest classifier, which is then applied to detect segments and to assign corresponding semantics to them. With advances in machine learning, new approaches for semantic enrichment rely on using 2D and 3D CNNs for classification of built environment features [Ioannidou et al. 2017]. The 2D and 3D CNNs classify 3D point clouds effectively for semantic enrichment of both indoor [Dietze et al. 2017] and outdoor scenes [Hackel et al. 2017; Huang and You 2016; Wolf et al. 2019]. We extend the previous work of Stojanovic et al. [2018a], by evaluating a method for improving the generation of multiviews (instead of using perspective projections captured in cubemap images) to increase the probability of correct classification of indoor 3D point clouds. Apart from parsing semantics, interactive viewing and annotation are also requirements for multiple stakeholder engagement; such as web-based 3D visualization for outdoor 3D point clouds described by Discher et al. [2018a].

Multiview Classification. The two established methods for classifying 3D data include 3D mesh feature and 2D image-based classifications. The former applies a 3D CNN trained on either on mesh features such as surface curvature, or the voxelized representation, of a given 3D object to distinguish between geometric features (examples include Qi et al. [2017] and Wang et al. [2017a]). The later applies a 2D CNN image classifier, and uses consecutively captured 2D images of 3D objects for classification. However, in terms of both training and classification, a 2D image-based approach is still considered to be faster and more suitable [Wang et al. 2017b]. Earlier research on 3D shape classification based on multiview classification was presented by Su et al. [2015]. The authors described two different multiview classification setups for capturing 2D images of 3D objects. The first setup assumes that the 3D object being captured placed on a given axis and 12 cameras are placed around the model at even angle spacing. The second setup assumes

that the 3D model is not orientated towards a specific axis and uses an icosahedron placed around the model to capture different views (using the vertices of the icosahedron as camera locations). At all times, the direction of the camera is oriented towards the 3D object. The classification results show that 2D CNN classification on the multiview images outperformes the 3D shape classification methods at that time.

In Kanezaki et al. [2018], a dodecahedron determines the camera view sampling locations from all directions towards the 3D object with the elevation angle of the camera being changed using a given angle interval. The use of such geometric primitives for view sampling has influenced our approach of using bounding spheres for viewpoint generation of segmented 3D point clusters. In terms of classifying 3D point clouds using multiview classification, more recently multiview approaches expand on this research by combining multiview classification with 3D point clouds CNNs [You et al. 2018]. Another notable approach is based on partial and downsampled multiview 3D point clusters of indoor spaces instead of 2D images to perform automatic classification and segmentation [Zhu et al. 2018]. While multiview classification has distinct advantages over other 3D shape classification methods, the approach disregards the attributes of the 3D model itself and relies on generating a set number of consecutive images [Huang et al. 2018]. From current multiview approaches, the direct correlation between the captured 3D viewpoint and the classification result is observed.

Viewpoint Entropy. Selecting optimal viewpoints to visually inspect and comprehend 3D shapes has been an active area of computer graphics research for many decades, though methods for selecting optimal viewpoints for 3D point clouds are not discussed. Early approaches to viewpoint selection made use of human participants for selecting optimal viewpoints of generic 3D models [Blanz et al. 1999]. Soon the idea of selecting the most optimal viewpoint shifted towards an automated approach, making use of visual entropy for selection of "best views". The computation of viewpoint entropy has been described by Vázquez et al. [2001] as the amount of visual information available for a single view based on the relative area of the projected 3D geometry faces over the sphere of virtual camera view directions centered in the viewpoint. This probability measure is based on information theory entropy measurement. The authors modify the value of the main probability variable to relate to the measure of the angle from the viewpoint direction to the angle of a given mesh face normal vector. The selection of surface features for measuring the viewpoint entropy is related to the mesh saliency of a given 3D object [Lee et al. 2005; Yamauchi et al. 2006]. This refers to the quantifiable features of a given 3D object that can be viewed, such as its surface curvature, which in turn effects user perception or entropy calculation for selecting a specific view of the object.

There is a wide selection of viewpoint entropy calculation methods based on different saliency measures, with reviews of methods performed by Bonaventura et al. [2018] and Dutagaci et al. [2010]. Each viewpoint entropy method is suitable for specific use cases and needs to be adapted for specific 3D object attributes (e.g., volumetric model viewpoint selection [Takahashi et al. 2005]). Other notable research by Riemenschneider *et al.* describes an approach for predicting a best view for semantic labeling of building facade components, using an entropy measure based on sparse geometric features [Riemenschneider et al. 2014]. Per-point normal vectors approximate the surface of the local point proximity. These can be computed efficiently by analyzing the local neighborhood of a point [Mitra and Nguyen 2003], and are used to orientate the point primitive according to the represented surface. The neighborhood of a point can be defined based on a sphere or a number of nearest neighbors (knn). It is computed using the covariance matrix of the neighbors and corresponding eigenvectors and eigenvalues [Hoppe et al. 1992]. Alternatively, a more robust normal estimation approach can also be used [Mura et al. 2018].

We make use of point normal vectors related to randomly-sampled viewpoints from an object-centred bounding sphere to establish the optimal viewpoints for multiview classification. A 3D bounding sphere is optimal for approximating the randomly distributed sampling positions for entropy calculation, as it allows a 3D object to be viewed from multiple and evenly distributed viewing directions. The random sampling and computation of viewpoint entropy measurements is inspired by the research from Lauri et al. [2015], where a Monte Carlo tree-search simulation is applied to optimize a system for finding the optimal views for object recognition.

3 APPROACH

Our approach is based on the use of clustering methods for segmenting spatial regions of RGB 3D point clouds with respect to furniture objects. The point cloud is first acquired by the user, using common RGB point cloud acquisition methods (e.g., photogrammetry). Our current approach also requires the user to provide a segmented point cloud with pre-computed normal vectors. Additionally, the user has to adjust the desired clustering parameters (e.g., number of clusters based on number of furniture objects in the scene), as well as the viewpoint entropy sampling parameters. The pre-segmentation is required to remove the walls, floors, and ceiling point clusters — this can either be performed automatically or manually by the user. The normal vectors of the segmented point cloud can then be pre-computed, using a planar local surface approximation model (with a preferred normal orientation along the positive Y-axis). We use the open-source software CloudCompare to accomplish these pre-processing tasks [Girardeau-Montaut 2011].

For each generated furniture 3D point cluster we generate a bounding sphere with 994 vertices, using the default 3D sphere object included with the Three.js framework (Section 3.4). The bounding sphere encloses each furniture object point cluster. We use the shuffle method by Fisher et al. [1949] to shuffle an array copy of the generated bounding sphere vertices, whose position and inverted normal vectors we then randomly sample a portion of as a set. We use this set for sampling potential camera directions and positions. We then compare the inverted sphere vertex normal vectors from this set to each of the corresponding 3D point cluster normal vectors. For each of the 3D point cluster normal vectors that are close to being parallel to the camera direction vector (e.g., if the *dot* product is between 0.75 to 1.0), we mark as an optimal view (Fig. 2). We use this as the main entropy measure for the captured viewpoints. The generated multiview images are then classified, where they are given a classification score (probability

of being either a sofa, chair or table object), and the average classification value is then computed for that given cluster. The entire implementation is used as part of a classification component for a service-oriented prototype application for semantic enrichment of indoor 3D point clouds. This process is repeated for each cluster in the 3D point cloud.

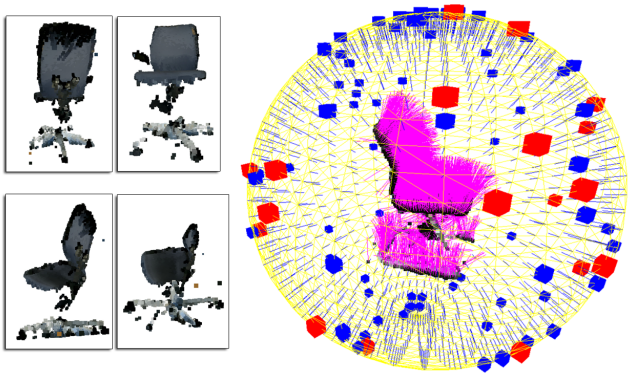


Figure 2: Example of selection and generation of multiviews for a chair object point cluster. Bounding sphere vertices (blue) are selected as multiview camera positions and directions (red), based on the parallelism entropy measure between the inverted bounding vertex normal (blue) and the point cluster normal vectors (purple).

Two different clustering methods for spatially segmenting the furniture objects are used: DBSCAN [Ester et al. 1996] and *k*-means clustering [Lloyd 1982]. We prefer the DBSCAN algorithm for 3D point clouds that contain more defined spatial divisions between the furniture objects. Otherwise, the use of *k*-means clustering is suitable for clustering cluttered scenes or scenes where no clear spatial divisions are present (e.g., if chairs are tucked in partially under the tables). The DBSCAN algorithm is sensitive to the density of the 3D point cloud and the number of generated clusters depends on the user parameter inputs that need to be configured for each given scene [Aljumaily et al. 2017]. Alternatively, *k*-means clustering can be used by setting a default number of clusters and increasing the number of cluster segments to the 3D point cloud with each iteration until a stopping condition is reached [Kim and Sukhatme 2014], or with the user determining how many clusters are required by visually inspecting the given 3D point cloud. We use the spatial distance between each of the points as the main attribute for both clustering approaches. Our implementation currently supports up to 20 clusters for each of the clustering methods. Due to recursion limits in JavaScript, the given 3D point cloud has to be down-sampled to below 10 000 points for DBSCAN clustering. For the DBSCAN, we sample a minimum number of 1 point and compute the Euclidean distance as the main distance measurement value between points. Because of this, the DBSCAN algorithm performs faster in our prototype application. The DBSCAN clusters generated from the sub-sampled 3D point cloud are then used to segment the clusters of the full resolution 3D point cloud.

3.1 Viewpoint Selection

Selection of viewpoints is performed for each furniture object cluster, obtained from the full resolution point cloud of the given indoor scene. We define the cluster C as $C := \{P \in \mathbb{R}^3, \vec{N} \in [-1,1]^3, RGB \in [0,1]^3\}$. For each cluster C representing a finite set of points, we then generate a bounding sphere, which we define as a standard 3D Sphere $\mathbb{B} := \{M \in \mathbb{R}^3, r, S_w, S_h, \phi, \theta\}$, with M being the center point, r the sphere radius from the center point, S_w and S_h horizontal and vertical sphere divisions, and ϕ and θ the horizontal and vertical sweep angles.

The bounding sphere B generated for each cluster, based on $\mathbb{B}$, can be defined as $B := \mathbb{B}(C)$. We then randomly sample the vertices from the bounding sphere, defined as subset B_v, where $B_v = \sigma(B) := \{P \in \mathbb{R}^3, \vec{N} \in [-1,1]^3\}$. We test each of the randomly-sampled inverted bounding sphere vertex normal vectors for parallelism with the normal vectors from each of the points in C.

We define this as the *Entropy Function* $f : \mathbb{R}^3 \rightarrow \mathbb{R}$, with $\mathbb{R} \rightarrow \{0,1\}$, where $f(C_p, B_p) = \begin{cases} 1, T \geq C_{\vec{N}} \cdot B_{\vec{N}} \leq 1.0 \\ 0 \end{cases}$, and where the bounding sphere vertices and cluster points that have normal vectors considered close to parallel within threshold level T are marked as *TRUE* positions and directions for the virtual camera. Finally, we define our virtual camera as $Cm := \{\vec{L_d} \in \mathbb{R}^3, \vec{L_u} \in \mathbb{R}^3, P \in \mathbb{R}^3\}$, where $\vec{L_d}$ represent the camera direction, $\vec{L_u}$ the camera up vector, and P as the camera position. The virtual camera object is used to synthesize the multiview 2D images. This approach is further summarized in Algo. 1.

Algorithm 1 Viewpoint selection

Require: C, B_v, Cm
  $B \leftarrow C$ {Generate bounding sphere around each point cluster}
  $B_v = \sigma(B)$ {Randomly sample vertices from the bounding sphere}
  for $i = 0$ **to** $length(B_v)$ **do**
    for $j = 0$ **to** $length(C_P)$ **do**
      if $T \geq C_{\vec{N}} \cdot B_{\vec{N}} \leq 1.0$ **then**
        $Cm \leftarrow B_p, B_{\vec{N}}$ {Set camera position and direction to bounding sphere vertex position and inverted normal}
      end if
    end for
  end for

3.2 Multiview Classification

3.2.1 Retraining. We use the Inception V3 CNN model to retrain two CNNs for classifying common office furniture items (e.g., chairs, tables and sofas). We only retrained the last bottleneck layer of the complete CNN model; Inception V3 was originally trained using ImageNet dataset [Russakovsky et al. 2015]. For the training data, we used 9759 different RGB images of chairs, tables and sofas as we consider these common office furniture. These images were obtained by using a batch image download script for Google Image search written in Python. We downloaded in bulk sets with up to 500 images, downloading batch images with search terms such as "office chair", "office sofa", "conference table", etc. For the three different classification categories of chairs, tables, and sofas we use 3605,

2900, and 3254 images per image training class. The images were reviewed and those that featured pictures of furniture items without too much visual clutter were selected and resized to 300×300 pixels (with the aspect ratio preserved). The training data input vector size is $300 \times 300 \times 3$ elements. Random distortion of training data (brightness, scale, and cropping) was not utilized for the retraining. The predicted classification accuracy using the retrained Inception V3 CNN is 92.9%, using 4000 training steps with a learning rate of 0.01. The second version CNN is used for classifying scenes with only chairs and tables, being retrained using the same training parameters, except using only using photos of chairs and tables. The second CNN model has a predicted classification accuracy of 94.5 %. In initial research tests we found that using a CNN with only two object furniture criteria for scenes that feature only two types of furniture point clusters provided us with more accurate classification results.

3.2.2 Classification. Each of the generated 2D images with optimal viewpoint is given a file name containing the cluster ID they belong to and the image sequence they are part of. These images are stored on the client and sent to the server for classification via a Node.js express server application. The Node.js server application calls the image classification script that classifies each of the images and generates a text file with a probability value for each of the three classification categories (chairs, sofas and tables). The classification script is based on Tensorflow 1.11.0 (compiled for AVX2 CPU instructions for cross-system compatibility), to create and run a classification tensor using the retrained version of the Incpetion V3 CNN. The classification probability scores for the input image data are calculated based on a linear softmax function.

Once all of the images have been classified and their text results generated, we call the Node.js express server to parse each of the text files and extract the cluster id from their file name (generated result text files have same names as the image files that are classified), and the classification probability values. The Node.js server then sends these results back to the client. The client associates each batch of results with each cluster, and generates and average classification value for that cluster. Since each cluster usually represents a single furniture object (or groups of the same type of object), we can associate the classification value with each point cluster, thus semantically enriching it.

3.3 Service-Oriented Design

Methods for service-oriented visualization and semantic enrichment of geospatial and point-cloud data have previously been described by [Discher et al. 2018b; Hagedorn and Döllner 2007; Stojanovic et al. 2018a]. We used this approach to implement a prototypical 3D web application based on the service-oriented architecture paradigm (Fig. 3).

We make use of HTML5 and Three.js [Cabello et al. 2010] for the client-side 3D visualization and implement our express server using Node.js based on an echo/server architecture. We can currently load and classify scenes with up to 4.5 million RGB points for approximately 20 m^2 of indoor space using direct and non-optimized 3D rendering of point clouds with Three.js. The majority of client and server framework is implemented in Javascript. We make use of Websockets to enable communication between the client and server. For image classification tasks, Tensorflow with Python 3.6.1 was used, and the specific Python scripts are called via Node.js.

The server for our web application is implemented using an echo client/server architecture. It listens to any communication by the client from a given port (e.g., SEND and GET responses for classification data). The server loads in the generated classification results as simple text arrays, before parsing the numerical classification values as floating point variables and sending these back to the client. Once the current classification operations are completed, the server removes all of the classification images and text file results. Using the client interface, a user is able to load in a pre-segmented RGB 3D point cloud for classification and visualization (we use RGB point-cloud data in the Stanford PLY file format). The optimal viewpoint images are generated by the client by accessing the *domElement* data URL of the current scene 3D rendering context. The client saves the images as JPEG files (using the resolution of the current 3D rendering context), via an automatic download link generation. Using a image resolution of 3000×1772 pixels, the average generated multiview image file size is 221 kB. A default download directory that is on the client computer is chosen prior to running the generation of optimal viewpoint images. Images from this directory are sent to the server for classification. While our approach of generation of multiview images and clustering is performed client-side with potential limitation of data throughput, it enables potentially sensitive data to be kept on the client side and provides better security management of local client data.

3.4 Testing Procedure

We conducted a total of 64 different tests, whereby we had 32 tests using the Stanford dataset [Armeni et al. 2017] with 16 different scenes, and 32 tests based on 16 synthetic scenes that we constructed. We created the synthetic scenes of varying complexity from furniture point clusters captured using a mobile phone scanning device and also objects from the RGB-D Object Dataset from University of Washington [Lai et al. 2011]. The non-synthetic Stanford dataset was chosen as it features a high number of cluttered items in the desired classification categories (e.g., chairs, tables and sofas). We did not spatially edit the Stanford dataset, as we just extracted the 3D point clusters featuring the furniture objects using manual segmentation with the CloudCompare software tool. For each of the two datasets, we tested each of the 16 scenes using either *k*-means or DBSCAN clustering. The field-of-view based on the circumference of a 3D point cluster bounding sphere was kept a constant value of 1.8 for all of the tests; this enabled each image to completely capture each cluster from an optimal view position and direction. The optimal classification parameters were chosen after initial tests were carried out to fine-tune the parameters for best results. The point size of the 3D point cluster was increased to 4 pixels to prevent any background color to be shown.

In terms of calculating the probability of correct classification of each scene, we used a score of 1, 0.5, 0.25, or 0 for furniture clusters that are either completely or partially classified. We decided to include the classification score of partially classified objects as this is a common classification outcome in cluttered indoor environments. The scores for the correct or incorrect classification of each furniture object were then assigned empirically using visual observation. In

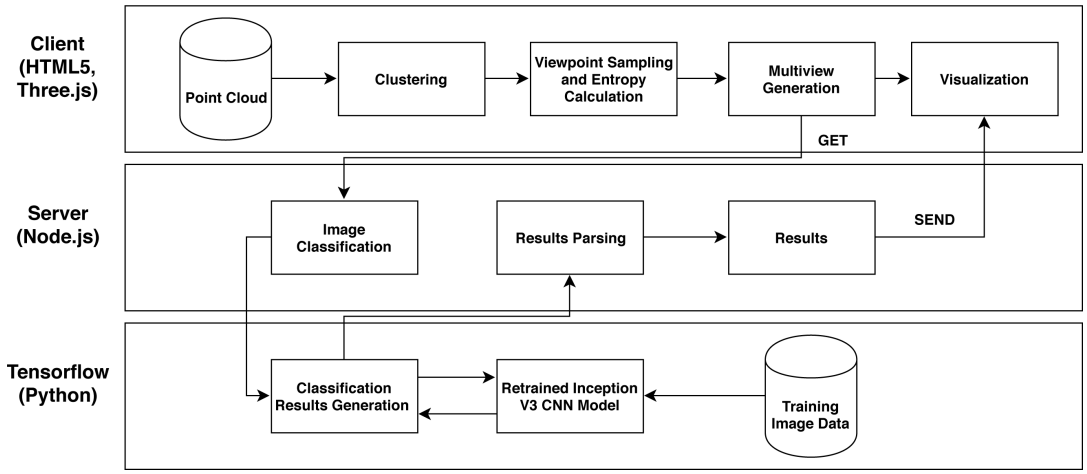


Figure 3: A high-level overview of the implemented client-server model for our web-based indoor point cloud multiview classification application prototype. The initial point cloud is acquired by the user, stored on the server and the user adjusts the clustering and viewpoint entropy sampling parameters prior to classification.

terms of object semantics, we assume all sofa chair objects as sofas, and connected tables are treated as a single table object unless they are adjacent or if there is a visible space between the point clusters. Additionally, coffee tables, office desks and conference desks are all treated as "table" objects. The selected number of clusters for each scene was determined visually by counting how many objects of each type are present in the scene. For example, if a scene has two chairs and a table, a minimum number of three clusters would be chosen to be approximated using either k-means or DBSCAN clustering. Since our current implementation supports up to 20 clusters, some complex scenes from the Stanford dataset had to be subdivided into two smaller scenes to enable finer classification.

To investigate if we could reduce the number of generated multiview 2D images while keeping the probability of correct classification at an acceptable percentage, we decided to sample up to 36 different views per point cluster (3.6 %of the bounding sphere vertices) as this is the approach used in previous research (Section 2.3). Using this sampling size, we usually ended up with an average of 11 images per cluster. The selected parallel threshold level was chosen to be between 0.75 and 0.9 in most cases as we wanted to use mostly parallel normal vectors for sampling when calculating the entropy for the views to be selected. For some scenes with highly planar surfaces, we took a parallel threshold above 0.5. The computer used for testing was a commodity laptop with an Intel i5 1.8 GHz CPU, 8 GB RAM, and NVidia GeForce MX150 GPU with 2 GB video memory, using the FireFox 64.0 web browser.

3.5 Experimental Results

The results in Fig. 4 show the classification results for specific furniture type point clusters, using our multiview sampling approach with either DBSCAN or k-means clustering. The obtained experimental results from the tests show that the classification score for the Stanford data set is 63.57 % average using DBSCAN clustering, and 59.43 % average using k-means clustering. The classification score for our synthetic dataset is 71.51 % average using DBSCAN clustering, and 60.75 % average using k-means clustering. We also compared the classification of our approach against the cubemap image classification method described by Stojanovic et al. [2018a]. We used the same testing methodology (using the same version of the CNN retrained on photographs, and classification tests conducted using the same synthetic and Stanford datasets). The results show that the approach using the method by Stojanovic et al. [2018a] provides overall lower correct classification of 47.96 % average for the synthetic dataset, and 47.18 % average for the Stanford dataset.

For the clustering performance evaluation, both the k-means and DBSCAN algorithms were implemented in JavaScript, and we only sampled the X and Z coordinates of scenes from the synthetic 3D point cloud (which were sub-sampled to below 10 000 points). The average elapsed time for computing an average of 3.25 clusters for k-means is 1040.06 ms, while for DBSCAN it is 586.936 ms. The results show that DBSCAN is approximately twice as fast as the k-means clustering result, and its computation performance correlates with the density of the 3D point cloud rather than the number of clusters that need to be computed.

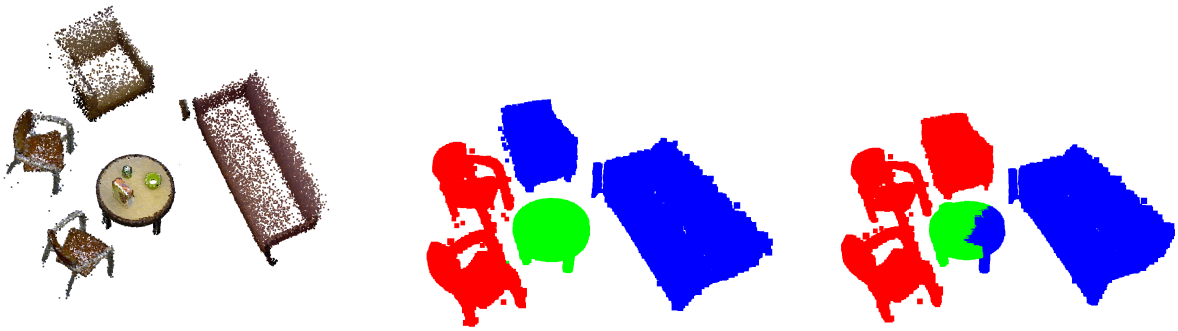
We also provide empirical results as four different classified 3D point cloud scenes from both datasets, using the selected clustering methods (Figs. 4 to 7). We also show a comparative empirical result (Fig. 8), using our approach versus the classification approach described in Stojanovic et al. [2018a]. Red clusters indicate chair object, blue clusters indicate sofa object, and green clusters indicate table object classifications.

4 DISCUSSION

The entropy-based viewpoint selection method allows our approach to generate multiview images that feature perspective projection of 3D point clusters based on the amount of the point normal vectors that are within a parallel threshold to the viewpoint camera. The one drawback of using only highly parallel normal vectors as an entropy measure is that for flat surfaces only directly top-down views will mostly be included in the entropy measure. Therefore,

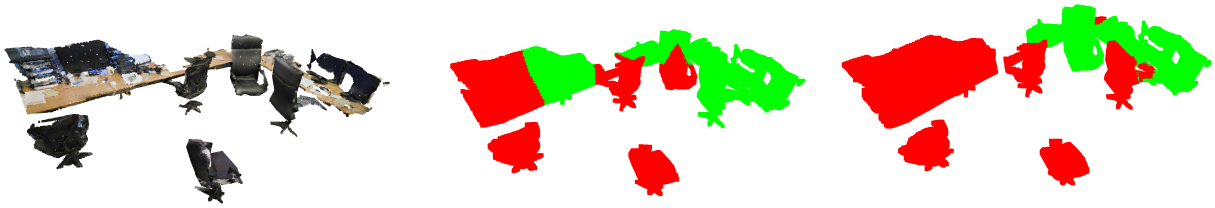
(a) (b) (c)

Figure 4: Classification results for a Stanford set scene. (a) Point cloud with 59 566 points, featuring 1 chair and 2 tables. (b) Classification based on *k*-means clustering, and (c) DBSCAN clustering.



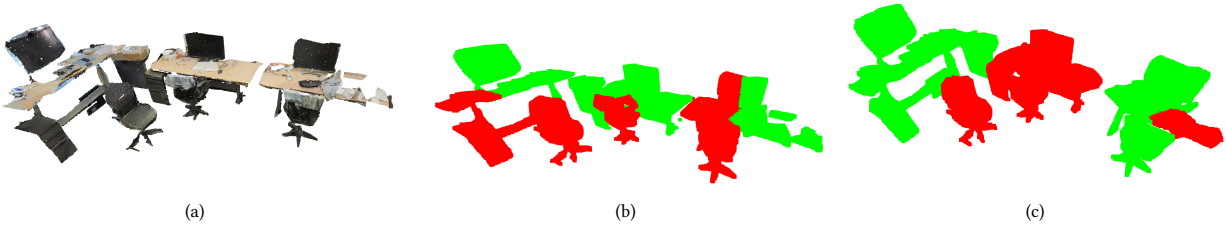
(a) (b) (c)

Figure 5: Classification results for a synthetic set scene. (a) Point cloud with 81 500 points, featuring 3 chairs, 1 sofa and 1 table. (b) Classification based on *k*-means clustering, and (c) DBSCAN clustering.



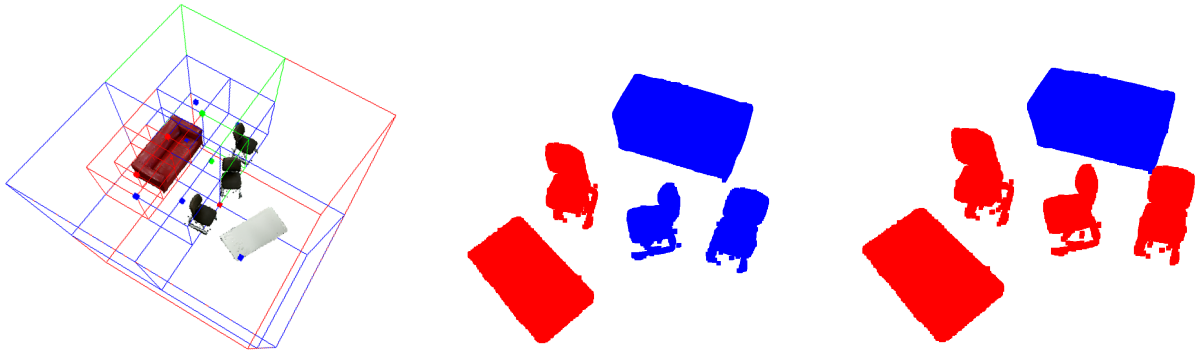
(a) (b) (c)

Figure 6: Classification results for a Stanford set scene. (a) Point cloud with 52 010 points, featuring 5 chairs and 2 tables. (b) Classification based on *k*-means clustering, and (c) DBSCAN clustering.



(a) (b) (c)

Figure 7: Classification results for a Stanford set scene. (a) Point cloud with 40 897 points, featuring 3 chairs and 3 tables. (b) Classification based on *k*-means clustering, and (c) DBSCAN clustering.



(a) (b) (c)

Figure 8: Comparative classification results for a synthetic set scene, with 50 955 points, featuring 3 chairs, 1 sofa and 1 table. (a) Classified using approach by Stojanovic et al. [2018a] with 16 object type nodes. (b) Our approach with classification based on *k*-means clustering, and (c) Our approach with classification based on DBSCAN clustering.

the parallel threshold needs usually needs to be lowered for scenes that feature objects with flat surfaces such as tables. The chosen sampling size for generating random camera positions and directions on the bounding sphere mesh of a given point cluster is low enough to maintain a small output of multiview images featuring a high amount of entropy. Using the described sampling and entropy calculation method, we generate an average of 11 images for a given point cluster (from a total of 36 randomly sampled bounding sphere points). This allows us to classify a scene in a few minutes on average (approximately three minutes on average using the test datasets on the test computer configuration). The number of generated images increases with the lowered parallel vector threshold for the normal vectors comparison. The empirical results (Section 3.6.2) demonstrate visually how our multiview approach handles scenes with various spatial configurations and varying levels of scene clutter. Additionally, the comparison of our approach to a similar multiview approach described by Stojanovic et al. [2018a] shows that our approach is able to classify both synthetic and real-world 3D point clouds with overall better probability of correct classification.

In terms of clustering 3D point clouds, we have evaluated both the DBSCAN and *k*-means algorithms. With our approach the highest probability of correct classification is achieved if the DBSCAN clustering method is used for clustering. We observed that the DBSCAN algorithm generally provides better clustering results, if the point set has uniform density (and in our implementation we only sample 2D points along the X and Z axes). The DBSCAN algorithm is also faster at computing clusters than the *k*-means algorithm. The *k*-means algorithm is better suited for clustering scenes with lots of clutter and lack of spatial divisions, but is not constrained by the non-uniform density of points. We have also found that over-segmentation using *k*-means can provide better segmentation results—in this case we generate a minimum of $n + 1$ clusters, where n is the number of objects in the given scene. For improved classification results, each of the main system parameters (clustering size, zoom out factor, entropy sampling size, and parallel threshold) should be adjusted according to the density of the 3D point cloud, the spatial profile of the scene (e.g., if the scene is cluttered or not), and the amount of objects needed to be classified. Currently, the number of clusters for *k*-means and the density sampling parameters for DBSCAN need to be set manually. The generated multiview images can then be classified using a retrained version of the Inception V3 CNN. Since the retrained CNN was trained using actual photographs of furniture items (chairs, tables, and sofas), thus it is more general in terms classification application as the furniture objects are use-case specific. While the probability of correct classification largely varies on the types of scenes (e.g., if the scene contains incomplete 3D point clusters), and the visual style of the 3D point clusters (e.g., chairs that are the same color as sofa objects can be classified as sofa objects, as shown in Fig. 5), our approach allows for fast and rapid approximation of 3D point cluster semantics.

We have also described how our approach can be used as a classification component within service-oriented systems. The presented software components, both client and server applications, were developed using modern web technologies with a focus on flexibility and robustness. Three.js for client-side visualization allows for interactive visualization of low to medium resolution 3D point clouds, while the use of Node.js and Python for server-side processing allows for interfacing with various machine-learning technologies.

5 CONCLUSIONS AND FUTURE RESEARCH

We have presented an approach for classification and semantics parsing of indoor 3D point clouds based on multiview 2D images taken from selected viewpoints. Based on a view entropy measure, we calculate adjustable and flexible viewpoints without the need of any previous scene semantics. The experimental results show that our approach is mostly accurate and able to quickly classify more than half of the objects in the tests scenes correctly. We have implemented our approach as a service-based software component, which can simplify its integration in complex and larger service-oriented Building Information Modelling, Facility Management, Digital Twin and Geoinformation systems. For future work, we plan to evaluate spatial inference methods to better evaluate the classification results (e.g., bounding volume comparisons), and to evaluate the OPTICS [Ankerst et al. 1999] clustering algorithm.

ACKNOWLEDGMENTS

This work was funded by the Research School on *Service-Oriented Systems Engineering* of the Hasso Plattner Institute, Faculty of Digital Engineering, University of Potsdam, Germany.

REFERENCES

Harith Aljumaily, Debra F Laefer, and Dolores Cuadra. 2017. Urban point cloud mining based on density clustering and MapReduce. *Journal of Computing in Civil Engineering* 31, 5 (2017).

Mihael Ankerst, Markus M Breunig, Hans-Peter Kriegel, and Jörg Sander. 1999. OPTICS: ordering points to identify the clustering structure. In *ACM Sigmod record*, Vol. 28. ACM, 49–60.

Iro Armeni, Sasha Sax, Amir R Zamir, and Silvio Savarese. 2017. Joint 2d-3d-semantic data for indoor scene understanding. *arXiv preprint arXiv:1702.01105* (2017).

Iro Armeni, Ozan Sener, Amir R Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 2016. 3d semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1534–1543.

Volker Blanz, Michael J Tarr, and Heinrich H Bülthoff. 1999. What object attributes determine canonical views? *Perception* 28, 5 (1999), 575–599.

Xavier Bonaventura, Miquel Feixas, Mateu Sbert, Lewis Chuang, and Christian Wallraven. 2018. A survey of viewpoint selection methods for polygonal models. *Entropy* 20, 5 (2018), 370.

Ricardo Cabello et al. 2010. Three. js. *URL: https://github. com/mrdoob/three. js* (2010).

Pascual Castelló, Mateu Sbert, Miguel Chover, and Miquel Feixas. 2006. Techniques for computing viewpoint entropy of a 3d scene. In *International Conference on Computational Science*. Springer, 263–270.

A Dietze, Marcel Klomann, Y Jung, Michael Englert, S Rieger, A Rehberger, S Hau, and Paul Grimm. 2017. SMULGRAS: a platform for smart multicodal graphics search. In *Proceedings of the 22nd International Conference on 3D Web Technology*. ACM, 17.

Sören Discher, Rico Richter, and Jürgen Döllner. 2018a. A Scalable webGL-based Approach for Visualizing Massive 3D Point Clouds Using Semantics-dependent Rendering Techniques. In *Proceedings of the 23rd International ACM Conference on 3D Web Technology (Web3D '18)*. ACM, 19:1–19:9. https://doi.org/10.1145/3208806.3208816

Sören Discher, Rico Richter, Matthias Trapp, and Jürgen Döllner. 2018b. Service-Oriented Processing and Analysis of Massive Point Clouds in Geoinformation Management. In *Service Oriented Mapping: Changing Paradigm in Map Production and Geoinformation Management*, Jürgen Döllner, Markus Jobst, and Peter Schmitz (Eds.). Springer.

Helin Dutagaci, Chun Pan Cheung, and Afzal Godil. 2010. A Benchmark for Best View Selection of 3D Objects. In *Proceedings of the ACM Workshop on 3D Object Retrieval*. ACM, 45–50.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, Vol. 96. 226–231.

Ronald Aylmer Fisher, Frank Yates, et al. 1949. Statistical tables for biological, agricultural and medical research. *Statistical tables for biological, agricultural and medical research*. Ed. 3. (1949).

Daniel Girardeau-Montaut. 2011. Cloudcompare-open source project. https://www.danielgm.net/cc/. (2011).

Timo Hackel, Nikolay Savinov, Lubor Ladicky, Jan D Wegner, Konrad Schindler, and Marc Pollefeys. 2017. Semantic3d. net: A new large-scale point cloud classification benchmark. *arXiv preprint arXiv:1704.03847* (2017).

Benjamin Hagedorn and Jürgen Döllner. 2007. High-level web service for 3D building information visualization and analysis. In *Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems*. ACM, Article 8, 8:1–8:8 pages.

Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. 1992. *Surface reconstruction from unorganized points*. Vol. 26. ACM. 71–78 pages.

Jing Huang and Suya You. 2016. Point cloud labeling using 3d convolutional neural network. In *2016 23rd International Conference on Pattern Recognition (ICPR)*. IEEE, 2670–2675.

Xiang Huang, Mantao Wang, Dejun Zhang, Yu Zhu, Lu Zou, Jun Sun, Fei Han, and Linchao He. 2018. Multi-view Fusion with Deep Learning for 3D Shape Classification. In *2018 International Conference on Audio, Language and Image Processing (ICALIP)*. IEEE, 189–194.

Anastasia Ioannidou, Elisavet Chatzilari, Spiros Nikolopoulos, and Ioannis Kompatsiaris. 2017. Deep learning advances in computer vision with 3d data: A survey. *ACM Computing Surveys (CSUR)* 50, 2 (2017), 20.

Asako Kanezaki, Yasuyuki Matsushita, and Yoshifumi Nishida. 2018. Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 5010–5019.

David Inkyu Kim and Gaurav S Sukhatme. 2014. Semantic labeling of 3d point clouds with object affordance for robot manipulation. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 5578–5584.

Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. 2011. A large-scale hierarchical multi-view rgb-d object dataset. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 1817–1824.

Mikko Lauri, Nikolay Atanasov, George Pappas, and Risto Ritala. 2015. Active object recognition via Monte Carlo tree search. In *Workshop on beyond geometric constraints at the international conference on robotics and automation (ICRA)*.

Chang Ha Lee, Amitabh Varshney, and David W Jacobs. 2005. Mesh saliency. In *ACM transactions on graphics (TOG)*, Vol. 24. ACM, 659–666.

Stuart Lloyd. 1982. Least squares quantization in PCM. *IEEE transactions on information theory* 28, 2 (1982), 129–137.

Niloy J Mitra and An Nguyen. 2003. Estimating surface normals in noisy point cloud data. In *Proceedings of the nineteenth annual symposium on Computational geometry*. ACM, 322–328.

Claudio Mura, Gregory Wyss, and Renato Pajarola. 2018. Robust normal estimation in unstructured 3D point clouds by selective normal space exploration. *The Visual Computer* 34, 6-8 (2018), 961–971.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE* 1, 2 (2017), 77–85.

T Qu, J Coco, M Rönnäng, and W Sun. 2014. Challenges and trends of implementation of 3D point cloud technologies in building information modeling (BIM): case studies. In *Computing in Civil and Building Engineering (2014)*. 809–816.

Hayko Riemenschneider, András Bódis-Szomorú, Julien Weissenberg, and Luc Van Gool. 2014. Learning Where to Classify in Multi-view Semantic Segmentation. In *ECCV*.

LS Runceanu and N Haala. 2018. Indoor Mesh Classification for BIM. *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* (2018).

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision* 115, 3 (2015), 211–252.

Vladeta Stojanovic, Matthias Trapp, Rico Richter, and Jürgen Döllner. 2018a. A service-oriented approach for classifying 3D points clouds by example of office furniture classification. In *Proceedings of the 23rd International ACM Conference on 3D Web Technology*. ACM, 2.

Vladeta Stojanovic, Matthias Trapp, Rico Richter, Benjamin Hagedorn, and Jürgen Döllner. 2018b. Towards The Generation of Digital Twins for Facility Management Based on 3D Point Clouds. In *Proceeding of the 34th Annual ARCOM Conference*. 270–279.

Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. 2015. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*. 945–953.

Shigeo Takahashi, Issei Fujishiro, Yuriko Takeshima, and Tomoyuki Nishita. 2005. A feature-driven approach to locating optimal viewpoints for volume visualization. In *VIS 05. IEEE Visualization, 2005.* IEEE, 495–502.

Paul Teicholz et al. 2013. *BIM for facility managers*. John Wiley & Sons.

Pere-Pau Vázquez, Miquel Feixas, Mateu Sbert, and Wolfgang Heidrich. 2001. Viewpoint selection using viewpoint entropy. In *VMV*, Vol. 1. 273–280.

Chu Wang, Marcello Pelillo, and Kaleem Siddiqi. 2017b. Dominant Set Clustering and Pooling for Multi-View 3D Object Recognition. In *Proceedings of British Machine Vision Conference (BMVC)*. 12.

Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. 2017a. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 72:1–72:11.

Johannes Wolf, Rico Richter, and JÃijrgen DÃűllner. 2019. Techniques for Automated Classification and Segregation of Mobile Mapping 3D Point Clouds. In *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 1: GRAPP*. SciTePress, 201–208. https://doi.org/10.5220/0007308802010208

Hitoshi Yamauchi, Waqar Saleem, Shin Yoshizawa, Zachi Karni, Alexander Belyaev, and H-P Seidel. 2006. Towards stable and salient multi-view representation of 3D shapes. In *IEEE International Conference on Shape Modeling and Applications 2006 (SMI'06)*. IEEE, 40–40.

Haoxuan You, Yifan Feng, Rongrong Ji, and Yue Gao. 2018. Pvnet: A joint convolutional network of point cloud and multi-view for 3d shape recognition. In *2018 ACM Multimedia Conference on Multimedia Conference*. ACM, 1310–1318.

Ye Zhu, Sven Ewan Shepstone, Pablo Martínez-Nuevo, Miklas Strøm Kristoffersen, Fabien Moutarde, and Zhuang Fu. 2018. Multiview Based 3D Scene Understanding On Partial Point Sets. *CoRR* (2018).