SPARK: A Universal Approach to 3D Point Cloud Segmentation Using 2D Image Segmentation Models – Demonstration on Traffic Objects

Michael Disser[1], Vivien Volland[2] and Gent Demiri[1]
[1]Institut für Numerische Methoden und Informatik im Bauwesen, Technische Universität Darmstadt, Franziska-Braun-Straße 7, 64287 Darmstadt, Deutschland
[2]FG Geodätische Messsysteme und Sensorik, Technische Universität Darmstadt, Franziska-Braun-Straße 7, 64287 Darmstadt, Deutschland
E-mail(s): disser@iib.tu-darmstadt.de, vivien.volland@tu-darmstadt.de

Abstract: The generation of point clouds using LIDAR or photogrammetry is used in various fields of (civil) engineering and allows to capture the geometry of scenes or objects. The categorisation of objects and their segmentation has to be done either manually or using isolated solutions with limited functionality. This paper presents the general purpose application SPARK, which uses 2D object recognition to perform automated 3D object segmentation in point clouds. The point cloud mesh is imported into the Unity game engine and an orthographic camera is flown over the mesh to capture images. 2D object segmentation CNNs are then used to classify and mask the detected objects. This information is projected onto the mesh using raycasts to create a three-dimensional localisation of the classified points in the scene. These points are clustered into bounding boxes and used to localise elements in the original point cloud and to create classified sub-point clouds. The workflow is demonstrated in the SPARK application using the example of capturing road objects with the object segmentation models pre-trained on the ADE20K and Cityscapes datasets. A case study with three point clouds of urban street scenes is performed and the results and applicability are discussed.
Keywords: Point Cloud, Segmentation, Computer Vision, 2D-3D Transformation, Gaming Engine

Erschienen in Tagungsband 35. Forum Bauinformatik 2024, Hamburg, Deutschland, DOI: 10.15480/882.13515

1 Introduction

Point clouds are a reliable, accurate tool for capturing geometric objects and are used in civil engineering for as-built documentation of environments and buildings. In geodesy, LIDAR (Light Detection and Ranging) using TLS (Terrestrial Laser Scanning) or MLS (Mobile Laser Scanning) and photogrammetry are common methods for generating point clouds. Point clouds provide a geometric representation of objects and environments. However, when used as a project-related data source, the semantic context of the represented objects must be identified and individual relevant objects must be (manually)

labelled or separated. This separation of point cloud objects is necessary to enable integration into model-based planning processes such as BIM (Building Information Modelling) or for simulations such as digital twins using as-built geometry. While automated methods exist for the recognition of various simple geometric bodies and some objects (see Chapter 2), the latter in particular are isolated solutions and in practice the majority of objects are identified and segmented manually. Training CNNs (Convolutional Neural Networks) for object recognition directly on point clouds is time-consuming and hardware-intensive, and generating data sources and labelling is labour-intensive [1].
The aim of this work is to present a generally applicable workflow that is capable of automatically recognising, localising and geometrically describing a large number of different objects for different use cases. Since a large number of labelled datasets already exist for training 2D image segmentation models (see Chapter 2), these will be used to support 3D object segmentation. Alternatively, users can train their own 2D image segmentation models by labelling images and training CNN models and include them in the workflow. In order to use the image segmentation models for 3D segmentation of point clouds, a game engine application has been developed that automatically generates photos in a game engine, segments them using the CNN models, projects the results back onto the point cloud (mesh) and thus creates a spatial reference. This spatial reference can then be used to identify the classified objects in the original point cloud and, if desired, create sub-point clouds of the isolated objects. The presented application allows the creation of segmented, labelled three-dimensional point clouds of specific objects and a hierarchical understanding of the scene can be generated. The segmented point clouds can also be used as labelled data for training 3D object recognition algorithms. The workflow is generally valid and is implemented on the example of traffic related objects and validated on three different point clouds.

2 Related Work

Point clouds acquired by methods such as terrestrial laser scanning (TLS), mobile laser scanning (MLS) or unmanned aircraft systems (UAS) often consist of millions of 3D points. The segmentation and classification of objects within point clouds is the subject of research for various application purposes. To segment and classify objects, 2D and/or 3D parts of point clouds are subdivided. A common method is the use of deep learning, especially CNNs, and feature identification algorithms.
K-means clustering (a method for vector quantization; also used for cluster analysis) of 3D point clouds is one approach to classify 3D points [2], [3]. In addition to classifying each point, [3], [4] use algorithms such as RANSAC (Random sample consensus) to detect planes within point clouds. [5] also uses RANSAC in combination with region growing algorithms to detect planes and thus semi-automatically detect objects in interior point clouds. [1] present a deep learning framework, including k-means clustering, that predicts a semantic label for each point in a 3D point cloud. They use point neighbourhoods and loss functions to refine feature spaces. Similar methods are used by [6], [7]. [2] extracts 3D and 2D geometric features by considering the local neighbourhood of each 3D point and projecting them onto a horizontal plane to extract the 2D features. Many segmentation and classification methods focus on indoor scenes. A common dataset for analysing outdoor scenes is the KITTI dataset, which is often used for autonomous driving scenarios [1]. [8] used this dataset for a real-time 3D object detector called PIXOR (ORiented 3D object detection from PIXel-wise neural

network predictions) to operate on point clouds. [8] outputs accurately oriented bounding boxes in real dimensions in a bird's eye view. Another common method is to use 3D voxel grids or 2D projections on surfaces in combination with CNNs [9], [10]. [9] discretises the projection into a 2D image and applies 2D convolutions. Another work focusing on outdoor scenarios with traffic-related objects is by [11]. The deep learning algorithm KPConv is used in combination with the NPM3D dataset. This dataset contains an outdoor scenario including poles, benches, bins and signs. Semantic segmentation combined with partial local neighbourhood segmentation is performed on the MLS data. Another technique for point cloud segmentation is the use of meshes. [12] used aerial and autonomous LIDAR point clouds and RGBD cameras for indoor floor and wall detection. The input data are 2D point sets, 3D point clouds and meshes. After mesh smoothing, dominant plane normal estimation, planar segmentation and polygon extraction, their system, called Polylodar3D, outputs planes and corresponding polygon representations. For outdoor scenarios, they also use the KITTI dataset, which is analysed from a bird's eye view. Raycasting is another method of detecting the location of points within a point cloud for more accurate segmentation. [13] use a raycasting approach to detect changes in point clouds of large urban areas. [14] also use a raycasting method in combination with a deep neural network, but for geolocating people and determining distances for search and rescue scenarios using UAS imagery.
In addition to the KITTI dataset, other datasets, their trained classes and their features were analysed for their suitability in the road infrastructure use case. ADE20K [15] and COCO [16] contain a wide range of common objects and are suitable for general use case independent analysis. Citycapes [17], KITTI [18] (both German) and BDD100K [19] (American) labelled road specific and urban objects like roads, signs, traffic lights, poles for use cases like autonomous driving. GTSRB [20] has over 40 different types of German road signs and can be used to specify which road sign is recognised.
The approach of the paper combines several of the above methods: pre-trained CNN-DL algorithms with 2D views of meshed point clouds, raycasting, and the final mapping of the segmented and classified traffic-related objects back onto the point cloud. To the authors' knowledge, such an approach does not yet exist.

3 The SPARK Application

The main requirement of the application is to be able to transfer the results of 2D object segmentation algorithms to the 3D point cloud. To ensure transferability to other use cases, it should be possible to combine matching object segmentation algorithms in a modular way. In order to generate images from different positions and angles, an automation of the image generation is required that allows (partial) automation based on user input for parameters and camera path. The application should be able to handle large point cloud meshes and their colliders, but there is no requirement for the application to run in real time, as no user input should be made at runtime. The results of the segmentation in 3D space should be clustered and the instances should be geometrically described, classified and exported in an applicable data scheme. To cover a wide range of use cases, the application should be scalable, have modular functionality and run on conventional hardware and applications.
To meet these requirements, a workflow concept is developed and an application called SPARK (Segmentation Processing and Analysis for Recognizing Keyobjects) is implemented. The described

workflow consists of modifications of point clouds with external consumer software as well as the application, which consists of different scripts and designed GameObjects in the game engine Unity, and the management of the different object segmentation algorithms through Python scripts. The modules of the workflow are shown in the Figure 1 and described in the following sections.

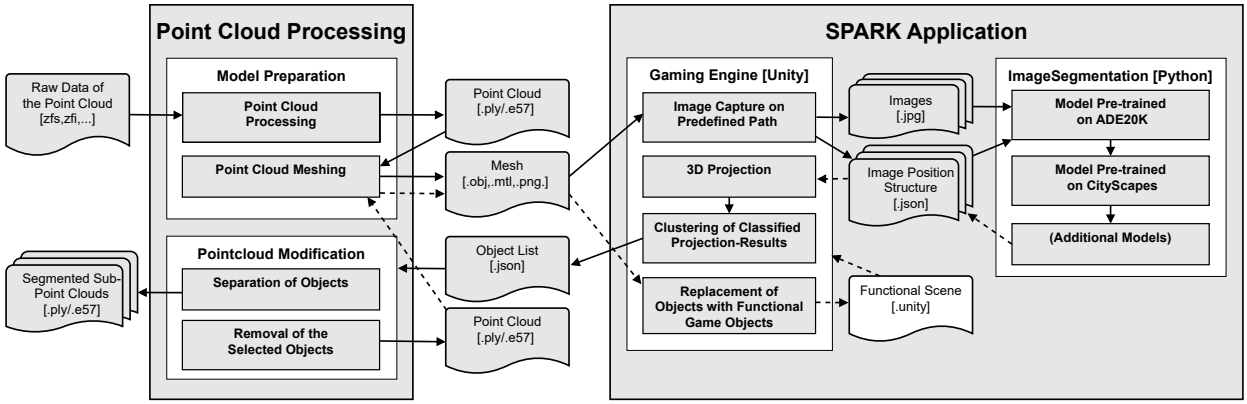

Figure 1: Overview of the SPARK application.

3.1 Pointcloud Preparation

Once the raw data of a scene has been captured, it needs to be processed so that it can be stored as a point cloud. The point cloud is meshed so that it can physically interact with the mesh collider in Unity. The results are then exported as an .obj with material and texture files.

3.2 Image Retrieval

To generate images from the point clouds for object recognition, an orthographic camera is moved along a path that takes images at fixed user-defined intervals. The path is a GameObject containing a line renderer with a polyline (see Figure 2). The goal is to capture all objects to be detected from at least three sides in order to draw a bounding box around the object (see Section 3.5). By rotating the camera at each capture point, images are taken from different directions, depending on the user's parameter settings for the GameObjects script. As a result, objects along the path are captured from different angles and directions as the camera moves past them. An orthographic camera was chosen to make it easier for the recognition algorithms in Section 3.5 to recognise objects compared to perspective cameras. It is also easier to recalculate the 3D points using the orthographic camera setting. The images are stored in a folder and the recording details of each image recording are stored using a class schema and saved as .json.

3.3 Image Segmentation

After the camera completes its path, an external Python script is triggered to analyse the captured images to identify and segment objects. The script uses a machine learning model to detect objects and and mark them with bounding boxes and masks. For the mask, not every pixel is used, but every n-th value, to produce a relatively detailed list of results, but not too many raycasts in the next step. For SPARK, different models are used that have been pre-trained with the Tensorflow Deeplab_v3 models

using the ADE20K [15] dataset and the Cityscapes [17] dataset (available at [1]). These datasets were chosen to get a mix of common objects and German urban street objects, as the point clouds are also generated on German scenes. The backbones used are MobileNetV2, MobileNetV3 and Xception. An example of the results of the Cityscapes dataset pre-trained on Deeplab_v3 can be seen in Figure 3.

Figure 2: Example of a manually created camera path; Current camera-POV in the bottom right corner.

Figure 3: Masking of the image object segmentation using Cityscapes.

3.4 3D Projection to the Mesh

The results of the image segmentation Python script are returned as output to the Unity application and embedded into the existing class structure. The instance of the captured image is updated with information about the objects detected within it, including the object's classes, locations, bounding boxes and masks. The user can filter the classes of objects to be included in the projection process to exclude irrelevant objects such as sky or void. The next step is to project the collected 2D information of the masked objects on each image into the three-dimensional mesh of the urban scene. To do this, the camera moves and orients itself to the exact orientation and position where the image was taken. Each label mask is iterated, and the two-dimensional pixel values of each object are also iterated. Each two-dimensional pixel coordinate is converted to the base plane (rectangle) of the orthographic camera. From this point, a raycast is created, oriented in the same direction as the camera view (perpendicular to the camera's base plane). This raycast represents the view direction of the renderer for that exact pixel. Therefore, the first part of the urban scene mesh hit by the raycast represents the object captured by the camera. This hitpoint is a three-dimensional representation of the two-dimensional object segmentation, as shown in Figure 4. By doing this for all pixels in the mask, and for multiple images with multiple positions and orientations, each detected object is now tagged with a multitude of classified coordinates.

3.5 Segmentation and Replacement

For each detected class, all 3D points are clustered using the DBSCAN algorithm. Each cluster result is then identified as an instance of the object and a bounding box is created to distinguish the points as part of the instance. The instance number, class label, bounding box geometry and identified

[1] Tensorflow Model-Zoo: https://github.com/tensorflow/models/blob/master/research/deeplab/g3doc/cityscapes.md

Figure 4: Visualisation of the segmentation results with lines and spheres as 3D representation of the raycast hits.

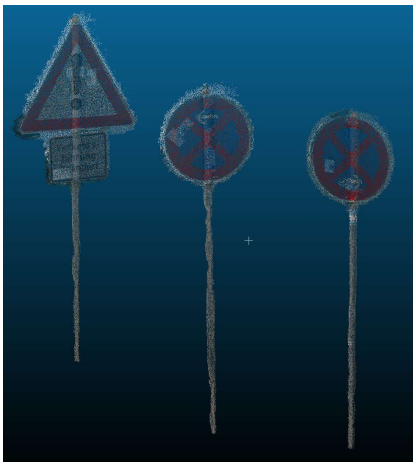

Figure 5: Point clouds of separated street signs and poles.

points are then exported as .json. This information is then used by another Python script which has two options: First, it can separate the detected objects as sub-point clouds, generating smaller point clouds with all points between the respective bounding boxes. This can be used to create classified point clouds consisting only of these objects. These can be used to create instances of realistic GameObjects or as training data for object detection in 3D point clouds. On the other hand, some classes can be selected by the user to be cut out of the whole point cloud. This can be used to replace these objects in the mesh for functional or interactive GameObjects. For example, the traffic lights in the original mesh could be replaced with functional traffic lights in the game engine by placing a functional GameObject at the position of the bounding box.

4 Results and Outlook

To validate the concept and application, three point clouds were generated in different types of urban scenes. While two of the point clouds contain large road crossings (one in the city centre and one in a vegetation-rich area), the third was in a relatively object-free environment. Two of the point clouds were generated using the NavVis VLX (v1) mobile laser scanner and one was generated using the Z+F 5016 imager. The raw data was processed using Z+F Lasercontrol and Scantra as well as the NaVis Ivion cloud application. Meshing was done using RealityCapture, Meshlab and CloudCompare. The results for the point cloud of an urban crossing with the detection can be seen in Figure 4 and the separation of traffic signs the associated poles in Figure 5.

The SPARK application and the case study proved that the workflow of the concept is a suitable method to detect and segment objects in point clouds. The modularity of the application allows it to be used for different use cases and can be adapted by the users through the parameters they can set for the GameObjects. Compared to relying only on images taken from the point cloud produced by scanners or photogrammetry, the paper approach has the advantage of generating images from different perspectives than the original images, resulting in better detection of objects due to better visibility and user-defined angles. The approach works even if the images have not been provided to the user or if only a mesh is available. The objects detected by the 2D object segmentation

algorithms can be seamlessly projected into the 3D world. Some pixel coordinates at the edge of the objects missed their target, which can be solved by omitting the edge pixels. As the results are highly dependent on the quality of the mesh and the object segmentation models, the functionality of the method is limited by the quality of these algorithms. This was particularly the case for small or thin objects such as road signs and traffic light poles. Depending on the quality of the mesh, these objects became bulky and sometimes unidentifiable. At this stage of the application, multiple objects of the same class next to each other are grouped as one object by the clustering algorithm. A solution to this would be to use instance segmentation algorithms such as those used in the BDD100K dataset [19]. The next step for the application is to introduce new segmentation algorithms to detect different types of objects. Lane detection, as introduced in BDD100K [19], can be used to identify the different lanes. The detected objects can be compared with publicly available information for traffic related objects such as those from OpenStreetMap or Mapillary. Scalability needs to be tested with larger datasets and also with point clouds produced by photogrammetry to enable larger scale capture of road infrastructure. Other use cases can be tested to ensure universal applicability. For larger or very detailed meshes, the use of the original mesh as a mesh collider was identified as a performance limiting bottleneck. To overcome this, an invisible lower poly mesh could be used for the collider. In addition, image quality could be improved by using Gaussian splats or a visualisation of the point cloud itself combined with a low-poly mesh collider to interact with the raycasts.

References

[1] F. Engelmann, T. Kontogianni, J. Schult, and B. Leibe, “Know What Your Neighbors Do: 3D Semantic Segmentation of Point Clouds”, en, in vol. 11131, Dec. 2018, pp. 395–409. DOI: 10.1007/978-3-030-11015-4_29.

[2] M. Weinmann, B. Jutzi, and C. Mallet, “Feature relevance assessment for the semantic interpretation of 3D point cloud data”, en, *ISPRS*, vol. II-5/W2, pp. 313–318, Nov. 2013. DOI: 10.5194/isprsannals-II-5-W2-313-2013.

[3] D. Iwaszczuk, T. Koch, and U. Stilla, “Innenraumrekonstruktion aus semantisch angereicherten 3D Punkten und Linien”, de, *DGPF, Band 26*, 2017. DOI: https://doi.org/10.1007/s41064-017-0029-9.

[4] J. Schmidt, V. Volland, D. Iwaszczuk, and A. Eichhorn, “Detection of Hidden Edges and Corners in SLAM-Based Indoor Point Clouds”, en, *ISPRS*, vol. XLVIII-1/W1-2023, pp. 443–449, May 2023. DOI: 10.5194/isprs-archives-XLVIII-1-W1-2023-443-2023.

[5] J. López Iglesias, J. A. Diaz Severiano, P. E. L. Amorocho, *et al.*, “Revision of Automation Methods for Scan to BIM”, in *XXIX INGEGRAF*, Logroño, Spain: Springer Nature Switzerland AG, 2019, pp. 482–490.

[6] S. A. Bello, S. Yu, C. Wang, J. M. Adam, and J. Li, “Review: Deep Learning on 3D Point Clouds”, en, *Remote Sensing*, vol. 12, no. 11, p. 1729, May 2020. DOI: 10.3390/rs12111729.

[7] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, *PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space*, en, Jun. 2017. [Online]. Available: http://arxiv.org/abs/1706.02413.

[8] B. Yang, W. Luo, and R. Urtasun, *PIXOR: Real-time 3D Object Detection from Point Clouds*, en, Mar. 2019. [Online]. Available: http://arxiv.org/abs/1902.06326.

[9] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, *Multi-View 3D Object Detection Network for Autonomous Driving*, en, Jun. 2017. [Online]. Available: http://arxiv.org/abs/1611.07759.

[10] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner, *Vote3Deep: Fast Object Detection in 3D Point Clouds Using Efficient Convolutional Neural Networks*, en, Mar. 2017. [Online]. Available: http://arxiv.org/abs/1609.06666.

[11] S. Yousefimashhoor, "Deep learning on MLS Point Clouds: Feasibility of Improving Semantic Segmentation Result Using Part Segmentation", en, Tech. Rep., Jul. 2022. [Online]. Available: https://essay.utwente.nl/91777/1/Yousefimashhoor_MA_ITC.pdf.

[12] J. Castagno and E. Atkins, *Polylidar3D - Fast Polygon Extraction from 3D Data*, en, Jul. 2020. [Online]. Available: http://arxiv.org/abs/2007.12065.

[13] J. Gehrung, M. Hebel, M. Arens, and U. Stilla, "A Voxel-Based Metadata Structure for Change Detection in Point Clouds of Large-Scale Urban Areas", en, *ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci.*, vol. IV-2, pp. 97–104, May 2018. DOI: 10.5194/isprs-annals-IV-2-97-2018.

[14] G. Paulin, S. Sambolek, and M. Ivasic-Kos, "Application of raycast method for person geolocalization and distance determination using UAV images in Real-World land search and rescue scenarios", en, *Expert Systems with Applications*, vol. 237, p. 121 495, Mar. 2024. DOI: 10.1016/j.eswa.2023.121495.

[15] B. Zhou, H. Zhao, X. Puig, *et al.*, *Semantic Understanding of Scenes through the ADE20K Dataset*, 2016. DOI: 10.48550/ARXIV.1608.05442. [Online]. Available: https://arxiv.org/abs/1608.05442.

[16] T.-Y. Lin, M. Maire, S. Belongie, *et al.*, *Microsoft COCO: Common Objects in Context*, Feb. 2015. [Online]. Available: http://arxiv.org/abs/1405.0312 (visited on 01/23/2024).

[17] M. Cordts, M. Omran, S. Ramos, *et al.*, *The Cityscapes Dataset for Semantic Urban Scene Understanding*, Apr. 2016. DOI: 10.48550/arXiv.1604.01685. [Online]. Available: http://arxiv.org/abs/1604.01685.

[18] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset", en, *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, Sep. 2013. DOI: 10.1177/0278364913491297.

[19] F. Yu, H. Chen, X. Wang, *et al.*, *BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning*, 2018. DOI: 10.48550/ARXIV.1805.04687.

[20] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "The German Traffic Sign Recognition Benchmark: A multi-class classification competition", in *IJCNN*, San Jose, CA, USA: IEEE, Jul. 2011, pp. 1453–1460. DOI: 10.1109/IJCNN.2011.6033395.