A Deep Learning Framework for Semantic Segmentation of Underwater Environments

Preprint · August 2022

CITATIONS
0

READS
317

3 authors:

Amos Smith
Deutsches Forschungszentrum für Künstliche Intelligenz
4 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Jeremy Coffelt
Universität Bremen
5 PUBLICATIONS **0** CITATIONS
SEE PROFILE

Kai Lingemann
Deutsches Forschungszentrum für Künstliche Intelligenz
101 PUBLICATIONS **3,303** CITATIONS
SEE PROFILE

A Deep Learning Framework for Semantic Segmentation of Underwater Environments

Amos Smith
Plan-Based Robot Control
German Research Center
for Artificial Intelligence (DFKI)
Osnabrück, Germany
amos.smith@dfki.de

Jeremy Coffelt
Institute for Artificial Intelligence
University of Bremen
Bremen, Germany
jcoffelt@uni-bremen.de

Kai Lingemann
Plan-Based Robot Control
German Research Center
for Artificial Intelligence (DFKI)
Osnabrück, Germany
kai.lingemann@dfki.de

***Abstract*—Perception tasks such as object classification and segmentation are crucial to the operation of underwater robotics missions like bathymetric surveys and infrastructure inspections. Marine robots in these applications typically use a combination of laser scanner, camera, and sonar sensors to generate images and point clouds of the environment. Traditional perception approaches often struggle to overcome water turbidity, light attenuation, marine snow, and other harsh conditions of the underwater world. Deep learning-based perception techniques have proven capable of overcoming such difficulties, but are often limited by the availability of relevant training data. In this paper, we propose a framework that consists of procedural creation of randomized underwater pipeline environment scenes, the generation of corresponding point clouds with semantic labels, and the training of a 3D segmentation network using the synthetic data. The resulting segmentation network is analyzed on real underwater point cloud data and compared with a traditional baseline approach.**

***Index Terms*—underwater robotics, 3D point cloud segmentation, deep learning perception, data simulation**

I. Introduction

Underwater tasks such as bathymetric surveys, infrastructure inspections, and maintenance of pipeline and cable networks are crucial from a scientific, environmental, and economic perspective. Common manual approaches to these tasks, such as scuba diving or the use of a remotely operated vehicle (ROV), often require specialized personnel, advanced scheduling, and are labor, time, and cost intensive. By employing autonomous underwater vehicles (AUVs), underwater tasks may be completed in a timely manner with less reliance on human interaction.

Marine robots in these types of tasks typically rely on sensors to obtain information about the complex underwater surroundings. Sensors such as laser scanners, depth cameras, and sonar sensors are able to produce three-dimensional point clouds of the environment. Rich data from these sensors enable the robot to accomplish perception tasks such as object classification and segmentation that are crucial to the operation of underwater missions. Traditional perception approaches, some of which are detailed in Section 2, often struggle to overcome water turbidity, light attenuation, marine snow, and other harsh conditions of the underwater world.

Deep learning-based perception techniques have proven capable of overcoming such environmental difficulties and uncertainties, but are often limited by the availability of relevant training data. Not only must a large amount of training data be collected, but it must also be labeled for the appropriate perception task. For example, in order for a 3D point cloud semantic segmentation network to be trained successfully, the training point clouds must be labeled to distinguish the object classes of interest. A common approach to label such datasets is to manually select the points and apply the associated labels by using a graphical user interface (GUI). Such a manual task is quite labor and time intensive, especially when repeated on thousands of point clouds each containing millions of points.

In this paper, we propose a framework that consists of the procedural creation of randomized underwater pipeline environment scenes, the generation of corresponding point clouds with semantic labels, and the training of a 3D segmentation network using the synthetic data. A deep-learning based solution for an underwater perception task may therefore be achievable without requiring a large amount of real-world training data.

Our primary contributions include:

1) A procedure for creating randomized underwater environments with seafloors, pipelines, and boulders
2) A method for converting these randomized environments into realistically-noisy, semantically-annotated 3D point clouds
3) Training and evaluation of a deep learning framework for segmenting both real and synthetic 3D point clouds of underwater environments
4) Comparison of the deep network to a standard, RANSAC-based approach

The remainder of the paper is organized as follows: Section 2 contains information on related works. An overview of the

The work presented in this paper was funded by BMWK as part of the project CIAM (grant number: 03SX540D). The DFKI Niedersachsen (DFKI NI) is sponsored by the Ministry of Science and Culture of Lower Saxony and the VolkswagenStiftung. This project has also received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 956200. For more info, please visit: https://remaro.eu/.

real data and a detailed generation process for the synthetic data is provided in Section 3. Section 4 discusses RANSAC-based and deep learning-based pipeline segmentation processes. Section 5 provides the results of applying the pipeline segmentation processes on the real and synthetic data, while Section 6 offers conclusions.

II. Related Works

A. Point Cloud Segmentation

Several surveys exist in the literature for 3D point cloud segmentation (PCS), including [1], [2], and, most recently, [3]. As these surveys reveal, newer approaches for PCS tend to rely primarily on deep learning. Still, classical approaches are being developed and proving successful.

1) Classical Approaches: Most classical PCS is based on edges, regions, models, or graphs. Edge-based techniques use local surface features, such as normals and curvature, to detect points where changes exceed some threshold. These edge points then form boundaries that segment the cloud into different regions [4]. Region-based techniques also use local surface features, but instead rely on merging neighboring points with similar features [5]. These techniques usually use seed points for which neighborhoods are grown around with similar features [6]. Model-based techniques attempt to fit geometric primitives to groups of points in the clouds. The two most familiar approaches – random sample consensus (RANSAC) and Hough transforms (HT) – both rely on voting procedures [7]. Graph-based techniques construct a graph with vertices for cloud points and edges determined by some proximity metric. For instance, in [8], a k-nearest neighbors is used to construct the graph and min-cut [9] is used to partition the graph. More details on the methods above are given in [10].

2) Deep Learning Approaches: Several recent surveys have been published on DL for 3D point clouds ([11], [12], [13], [14]). In particular, [15] is devoted solely to DL-based 3D PCS, which can be broadly classified as direct and indirect methods.

Indirect methods attempt to add structure to unstructured point clouds so that 2D-like solutions can be applied. The most common approaches involve multiview projection ([16], [17], [18]) and voxelation ([19], [20], [21]).

Direct methods work directly on the unstructured 3D point cloud by overcoming the difficulties of cloud sparsity, while remaining invariant to geometric transformations and point orderings [22]. Such methods can be classified as point ordering, multiscale, feature fusion, and graph CNNs [15].

Point ordering networks attempt to discover order and regularity in unstructured clouds without resorting to the projection and voxelation procedures discussed above ([23], [24], [25]). Multiscale methods combine multiple levels of detail to balance the inclusion of local features present at small scales and the exclusion of irrelevant information found at large scales ([26], [27], [28]. Feature fusion attempts to fuse global and local features. ([29], [30], [31]). Finally, graph CNNs (GCNNs) combine the capabilities of CNNs and the efficient structure of graphs ([32], [33], [34], [35], [36]).

For more details about these deep learning solutions, including their relative strengths and weaknesses, please see [15].

B. Previous Approaches for Pipeline Segmentation

Several previous attempts have been made specifically to segment pipelines in point clouds. In [37], the standard 5D Hough transform (HT) for cylinder fitting is reduced to a 2D transform to determine the axis of a cylinder, followed by a 3D transform to determine the position and radius of the cylinder. This greatly increases the efficiency of HT for cylinder fitting, as demonstrated on datasets from (above ground) industrial sites. In [38], RANSAC and principal component analysis (PCA) were used instead for cylinder matching. That work continues by using line and spline segments to connect cylinders into a continuous pipeline. Another approach to cylinder extraction is given in [39]. In that paper, potential cylinder points are determined from normal and curvature information. Then, neighborhood points are considered as inlier points to iteratively fit a cylinder. Such approaches would likely perform poorly in the underwater scenarios considered is this paper, where the ground is not assumed flat, the pipelines are only partially visible, and the resulting point clouds are much sparser and noisier.

A few attempts have even been made to segment underwater pipelines. These include [40], which combines point features and Bayesian estimation to segment pipe parts (valves, elbows, sockets, etc.) from uncolored point clouds. However, their procedure relies on RANSAC for removal of the planar surfaces of their testing pool, leaving only the pipe in the remaining point cloud. This pipe was also constructed from straight segments with known diameter and other parts stored in a database of partial CAD-based point clouds. In [41], a stereo camera was used to construct 3D RGB point clouds of underwater pipes, which were then segmented using the PointNet architecture. In this paper, we perform a similar task with much less information (no color channels or a priori assumptions about pipe components or diameters) and with a more modern deep learning framework.

C. Point Cloud Simulation

Because of the cost of acquiring real point clouds and the need for such data for machine learning training purposes, several solutions have been proposed for simulating and generating synthetic clouds. For instance, the Gazebo simulator has been used to simulate point clouds generated by both LIDAR [42] and sonar [43]. In [44], Gazebo was also used to automatically segment point clouds for natural, above-ground environments. To accomplish the same task free from the Robot Operating System (ROS) ecosystem, the BLAINDER add-on [45] can be used with the free and open-source 3D computer graphics program Blender. This add-on is capable of simulating point clouds from time-of-flight or depth sensors, including both LIDAR and sonar. Several simulators built upon game engines are also capable of generating synthetic point clouds [46], including Sim4CV [47] and AirSim [48], which are both built upon the Unreal engine. To the best of our

knowledge, our approach is the first to automatically generate segmented point clouds for randomized underwater scenes.

III. Data

Two datasets, one real and one synthetic, are used in the training and evaluation of the semantic segmentation network. An overview of the real-world data and its acquisition process are is provided in Section 3.1. Section 3.2 describes the generation procedure of the synthetic point cloud data with semantic labels.

A. Real Data

A freshwater reservoir was used as a test bed to acquire underwater point cloud data. The reservoir, with an approximate depth of 12 m and a flat, tarred bottom, contains a sunken mock-up pipeline structure. The pipeline structure is organized as an octagonal ring constructed from 8-in (219-mm) diameter pipe, with the overall diameter of the ring being 40 m.

A remotely-operated platform is outfitted with an ixBlue Phins C7 inertial navigation system (INS) and a Nortek DVL500 Doppler velocity log (DVL) to provide positional information. The platform additionally contains a downward-pointing AlliedVision Manta G-235 camera with a Sony IMX174 monochrome sensor. A laser line generator is situated on the platform which projects straight lines downwards and laterally onto planar surfaces.

A procedure as described in [49] is applied to the camera image to extract the laser line as a two-dimensional profile. The platform position estimate from the INS along with the known, relative pose of the camera and laser, allow for the compilation of consecutive 2D laser profiles into a 3D point cloud. Fig. 1 further provides a visual example of how the 2D laser profiles can be used to compose a 3D point cloud.

The dataset in this paper was obtained by navigating the testing platform through a single loop around the pipe ring structure at a height of 2.3 m above the pipeline center. The resultant point cloud, containing approximately six million points, is displayed in Fig. 2.

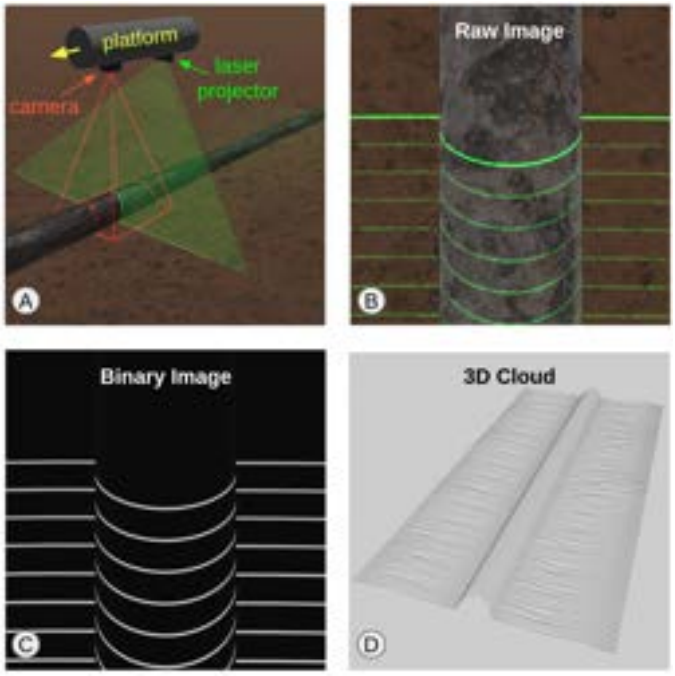

Fig. 1. Point cloud construction: (A) a laser is projected downward onto the seafloor and pipeline, (B) a raw camera image is collected, (C) a binary mask is used to segment pixels corresponding to the laser profile, and (D) the 2D laser profile, along with camera calibration info and the 3D position of the camera, produce a 3D point cloud.

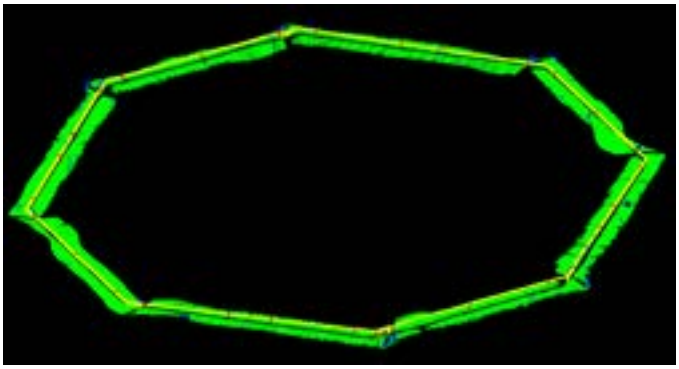

Fig. 2. CloudCompare screenshot of the manually-annotated test bed data with seafloor in green, pipe in yellow, flanges in red, and noise in blue.

B. Synthetic Data

An overview of the framework used to generate randomized synthetic data is shown in Fig. 3. At its core is the BLAINDER add-on for Blender, consisting of plugins that automate the generation of terrain meshes and semantically-annotated point clouds. We developed additional plugins to randomly place boulders of various sizes and another to create randomized pipelines of various diameters, bend radii, flange sizes, and burial depths. All of these parameters were saved within a YAML configuration file, which also specified the spatial size of each world, the number of worlds to be generated, and the formatting of the exported data.

Since our primary interest is in segmenting pipelines in real sonar point clouds, a key step in this process was the randomization of realistic terrains, pipelines, and boulder placements. Fig. 4 shows a 2D view of a typical iteration, where a pipeline is randomly started at an edge of the window, aimed toward the center. Then, based on relative probabilities (specified in the configuration file), the next pipe segment (straight, bend, valve, cathode-protected, etc.) is randomly selected and appended. This is repeated until the pipeline exits the bounds of the window. Next, flanges are added at each junction point of the pipeline. Finally, boulders are randomly placed in the window outside of some minimum distance from the pipeline.

With the 2D layout established, a Bezier curve is created for the midline of the pipeline, a circular cross section is extruded along this curve, and the result is converted into a pipeline mesh. Then, flange meshes are duplicated along the length of the pipeline, centered at each segment junction and normal to the pipe midline. Another Blender add-on, called Another

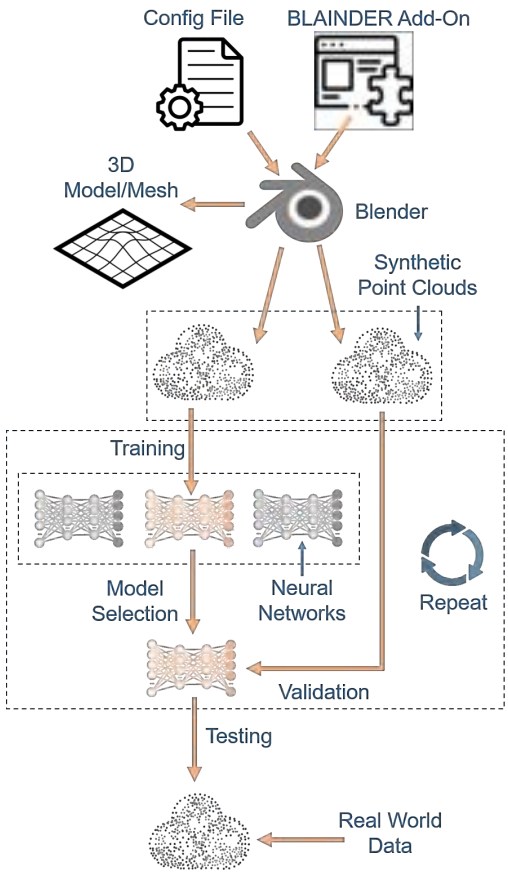

Fig. 3. Framework for end-to-end procedure of generating randomized worlds based on specifications given in a configuration file, exporting the corresponding annotated point clouds, and then training and testing of the neural network.

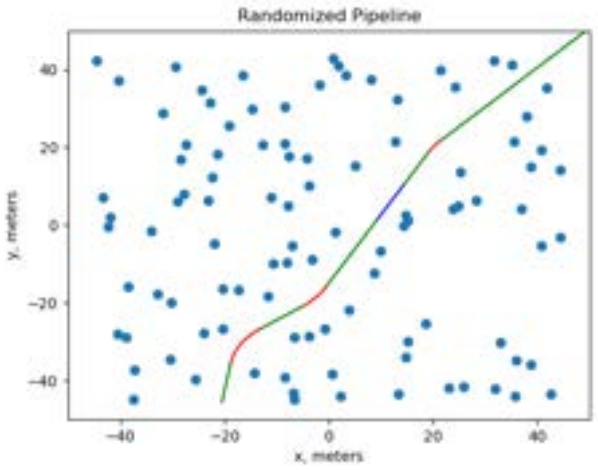

Fig. 4. Example of 2D prototyping with dots indicating the placement of boulders and various colors representing different types of pipe segments.

Fig. 5. Example of a randomized Blender scene with seafloor, pipeline, boulders, and sensor trajectory.

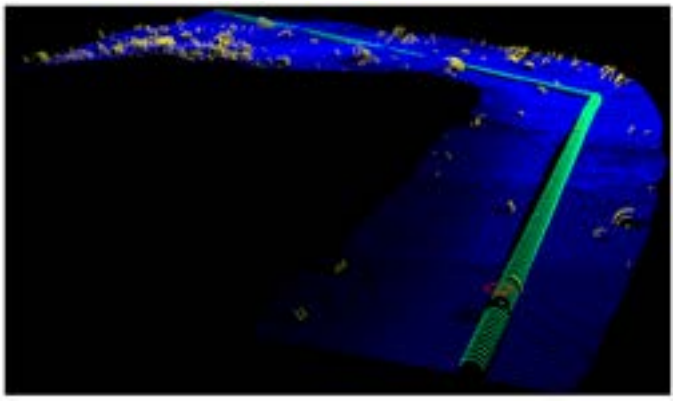

Fig. 6. CloudCompare screenshot of synthetic data with semantic labels.

Noise Texture (A.N.T.), is used to generate randomized terrain and boulder meshes based on the specifications given in the configuration file. Finally, the mesh is centered along the *xy*-plane and the pipeline is shifted up or down depending on the desired burial depth. A typical result is shown in Fig. 5.

Once the randomized terrain and pipeline are complete, another curve is generated to represent the path of the sonar sensor above the pipeline. With this done, the BLAINDER add-on generates the point cloud that results from following this curve with a sonar sensor configured with the desired field of view, maximum range, angular resolution, and noise type. Once this point cloud is generated, it is exported for further analysis. Fig. 6 shows the resultant point cloud of Fig. 5 viewed in CloudCompare.

This process for generating labeled point cloud data was repeated to produce 1500 scenes and their respective labeled point clouds to be used in the training of a deep learning segmentation network.

IV. Methods

A RANSAC-based cylinder detection is to serve as a classical baseline approach for pipeline segmentation. The automatic shape detection algorithm presented in [50] is implemented in CloudCompare, an open-source GUI tool for point cloud

viewing and manipulation. First, a plane is fit to the data with CloudCompare's plane fit tool, based on a standard least square fitting. Points from the point cloud are then filtered based on their distance from the plane, with points being within a 5 cm threshold being considered as ground points. The remaining non-ground points are then used in the RANSAC shape detection tool.

Parameters used in the cylinder detection include the following: 500 minimum support points per cylinder, a maximum distance to cylinder of 0.02 m, sampling resolution of 0.02 m, minimum cylinder radius of 0.08 m and maximum cylinder radius of 0.13 m.

After applying this process, the fit cylinders are obtained and matching points are considered to be associated with the pipeline. The leftover points are assumed to belong to some other class. An overview of this RANSAC-based approach is detailed in Fig. 7.

Due to its strong performance on 3D point cloud segmentation benchmarks such as ScanNet [51], the Minkowski Engine [52] library is used for the deep learning approach. The model applied in this paper is the Mink16UNet18, which is a Residual U-Net convolutional neural network. Few changes were made to the model for this specific application. The original number of input channels for this model is six, due to its use on RGB 3D point clouds. The input channels were therefore reduced to three, to account of the lack of color information in our data. In addition, the final layer was updated to account for the four classes considered in this paper: seafloor, pipe, flange, and other.

The Minkowski Engine library includes several functionalities to automatically load, augment, and quantize the data for training. For each training iteration, the data is augmented with a rotational and translational offset. The bounds of the rotational offsets were chosen to be $\pm 5^\circ$ about the x- and y-axes, and $\pm 180^\circ$ about the z-axis. The translational bounds were chosen to be ± 20 cm along each axis. For the quantization procedure, a voxel size of 5cm was used. The Mink16Unet18 network was trained with a batch size of 4 for 100 epochs, after which the training was terminated due to a notable convergence of the mean cross-entropy loss as shown in Fig. 8. The remaining training parameters for this network are unchanged from what is available in the Minkowski Engine repositories.[1]

The resultant segmentation network from the training process obtained an overall mean intersection over union (mIoU) score of 87.4, with class intersection over union (IoU) scores of 89.4, 99.6, 95.2, and 65.5 for none, seafloor, pipeline, and flange classes, respectively.

V. Experimental Results

The RANSAC cylinder process was executed a total of 10 different times on the test bed data in order to determine the variation in the resulting segmentation accuracy metric. The pipeline IoU scores for the RANSAC results were calculated.

[1]https://github.com/NVIDIA/MinkowskiEngine

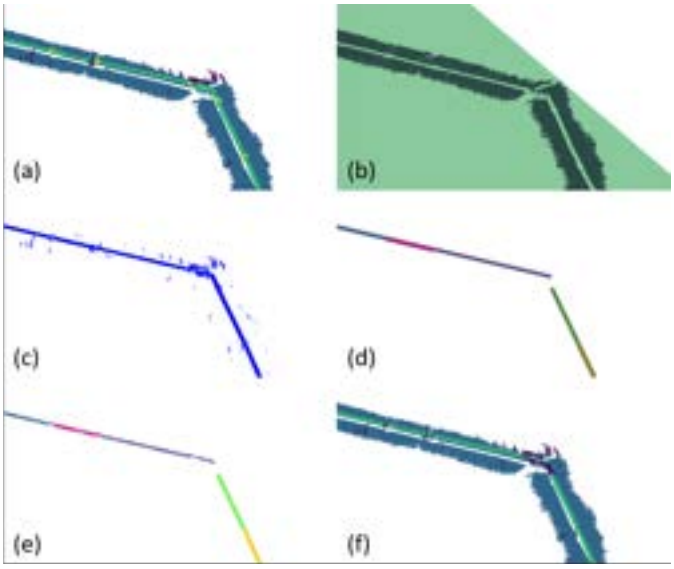

Fig. 7. RANSAC cylinder detection process. (a) original test bed dataset with manual labels for visualization, (b) plane fitting and ground point extraction, (c) remaining points after ground point extraction, (d) fitted cylinders, (e) corresponding points to cylinders, (f) final result with labels for ground (blue), pipeline (green), and none (purple).

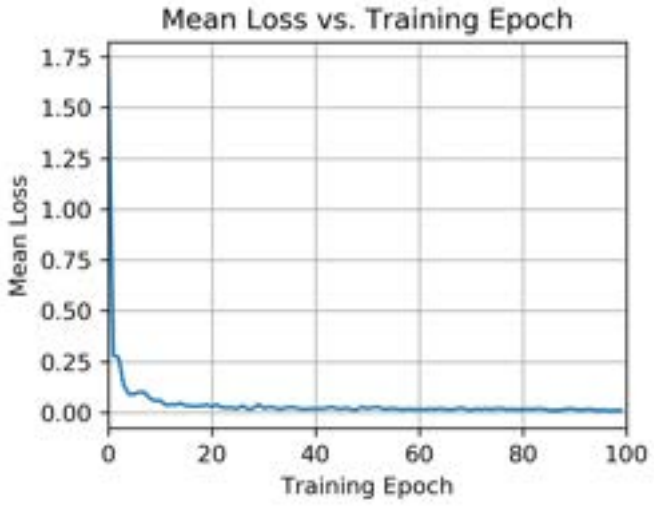

Fig. 8. Training loss versus training epoch.

The highest IoU score was found to be 0.907, the lowest score to be 0.84, and the average score to be 0.87. Since the RANSAC approach cannot segment the pipe flanges, the flange IoU score is assumed to be 0 in determining the mIoU score of 0.62. However, if the flange IoU score is instead to be ignored, then the resultant mIoU is 0.93.

The trained Mink16Unet18 network was applied to the test bed data and the IoU scores for the ground, pipeline, and flange classes were calculated. These results, presented in Table I, lead to an overall mIoU score of 0.68 for the trained network. An exemplary visual comparison of the two approaches is presented in Fig. 9.

TABLE I
SEGMENTATION COMPARISON

IoU	ground	pipeline	flange	mIoU
RANSAC	0.99	0.87	0 (N/A)	0.62 (0.93)
Mink16Unet18	0.95	0.86	0.23	0.68

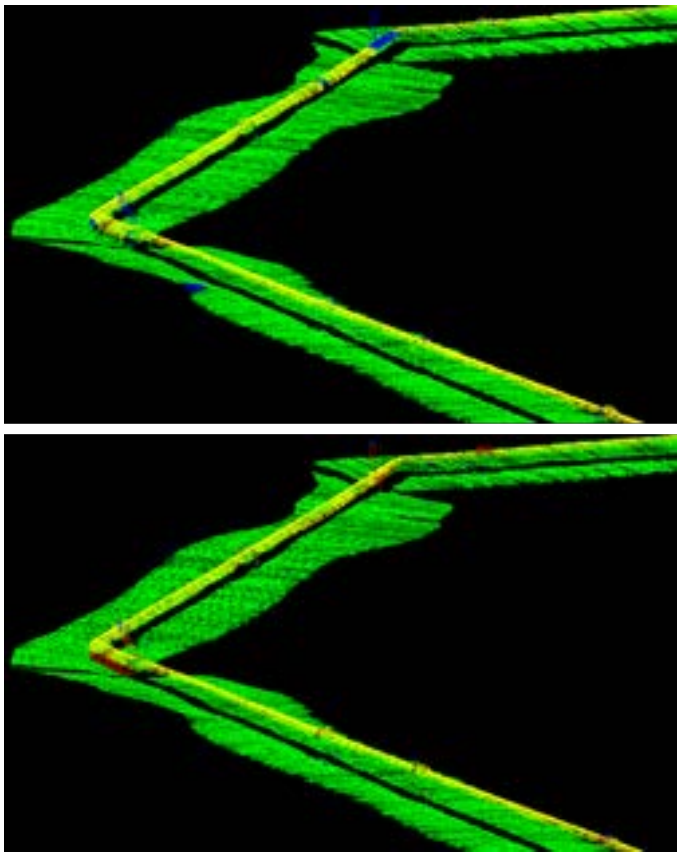

Fig. 9. Results comparison of the RANSAC approach (top) and the trained Mink16Unet18 (bottom) on real underwater point cloud data. The segmentation of the pipeline (yellow), ground (green), flange (red), and none classes (blue) are shown.

VI. CONCLUSIONS

We have shown how synthetic underwater environments can be procedurally generated and how semantically labeled point clouds of these environments can be obtained. A modern 3D point cloud segmentation network was trained with the synthetic data and analyzed on data from a real-world underwater test bed. The network was compared to a classical RANSAC cylinder detection for pipeline detection. The presented results show that the deep network was comparable to the RANSAC approach and was able to additionally provide segmentation of pipeline features such as flanges.

One of the most important benefits of the deep segmentation network is its ability to be automated. Once trained, the network can operate on data from environments with complex landscapes, varied classes of interests, and a range of pipeline diameters.

The RANSAC cylinder approach, however, requires a priori knowledge of the environment such as pipeline diameters in order for it succeed. Even with this a priori knowledge, the tuning process for the RANSAC parameters presented here required a time-intensive calibration. The real data set used in this paper was a best-case scenario for the RANSAC, due to the artificial reservoir with a flat bottom and fully exposed pipeline. The test bed environment is quite different from real situations with non-planar seabeds and partially buried pipelines. In such a more realistic environment, even the preliminary step of ground removal would be far from trivial as it was in our RANSAC-based approach.

REFERENCES

[1] A. Nguyen and B. Le, "3D point cloud segmentation: a survey," in *Sixth IEEE Conference on Robotics, Automation and Mechatronics (RAM)*. IEEE, 2013, pp. 225–230.

[2] E. Grilli, F. Menna, and F. Remondino, "A review of point clouds segmentation and classification algorithms," *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42, p. 339, 2017.

[3] Y. Xie, J. Tian, and X. X. Zhu, "Linking points with labels in 3D: a review of point cloud semantic segmentation," *IEEE Geoscience and Remote Sensing Magazine*, vol. 8, no. 4, pp. 38–59, 2020.

[4] B. Bhanu, S. Lee, C.-C. Ho, and T. Henderson, "Range data processing: representation of surfaces by edges," in *Proceedings of the Eighth International Conference on Pattern Recognition*. IEEE Computer Society Press, 1986, pp. 236–238.

[5] T. Rabbani, F. Van Den Heuvel, and G. Vosselmann, "Segmentation of point clouds using smoothness constraint," *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 36, no. 5, pp. 248–253, 2006.

[6] P. J. Besl and R. C. Jain, "Segmentation through variable-order surface fitting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 2, pp. 167–192, 1988.

[7] F. Tarsha-Kurdi, T. Landes, and P. Grussenmeyer, "Hough-transform and extended RANSAC algorithms for automatic detection of 3D building roof planes from LIDAR data," in *ISPRS Workshop on Laser Scanning 2007 and SilviLaser 2007*, vol. 36, 2007, pp. 407–412.

[8] A. Golovinskiy and T. Funkhouser, "Min-cut based segmentation of point clouds," in *Twelfth IEEE International Conference on Computer Vision Workshops, ICCV Workshops*. IEEE, 2009, pp. 39–46.

[9] Y. Boykov and G. Funka-Lea, "Graph cuts and efficient ND image segmentation," *International Journal of Computer Vision*, vol. 70, no. 2, pp. 109–131, 2006.

[10] X. Ruan and B. Liu, "Review of 3D point cloud data segmentation methods," *International Journal of Advanced Network, Monitoring and Controls*, vol. 5, no. 1, pp. 66–71, 2020.

[11] S. A. Bello, S. Yu, C. Wang, J. M. Adam, and J. Li, "Deep learning on 3D point clouds," *Remote Sensing*, vol. 12, no. 11, p. 1729, 2020.

[12] W. Liu, J. Sun, W. Li, T. Hu, and P. Wang, "Deep learning on point clouds and its application: a survey," *Sensors*, vol. 19, no. 19, p. 4188, 2019.

[13] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3D point clouds: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 12, pp. 4338–4364, 2020.

[14] H. Lu and H. Shi, "Deep learning for 3D point cloud understanding: a survey," *arXiv preprint arXiv:2009.08920*, 2020.

[15] J. Zhang, X. Zhao, Z. Chen, and Z. Lu, "A review of deep learning-based semantic segmentation for point cloud," *IEEE Access*, vol. 7, pp. 179 118–179 133, 2019.

[16] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3D shape recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 945–953.

[17] A. Boulch, B. L. Saux, and N. Audebert, "Unstructured point cloud semantic labeling using deep segmentation networks," in *Proceedings of the Workshop on 3D Object Retrieval*, ser. 3Dor '17. Goslar, DEU: Eurographics Association, 2017, p. 17–24. [Online]. Available: https://doi.org/10.2312/3dor.20171047

[18] J. Guerry, A. Boulch, B. Le Saux, J. Moras, A. Plyer, and D. Filliat, "Snapnet-r: Consistent 3D multi-view semantic labeling for robotics," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, pp. 669–678.

[19] D. Maturana and S. Scherer, "Voxnet: A 3D convolutional neural network for real-time object recognition," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 922–928.

[20] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese, "SegCloud: Semantic segmentation of 3D point clouds," in *International Conference on 3D Vision (3DV)*. IEEE, 2017, pp. 537–547.

[21] T. Le and Y. Duan, "Pointgrid: a deep network for 3D shape understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9204–9214.

[22] C. R. Qi, H. Su, K. Mo, and 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.

[23] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "Pointcnn: convolution on χ-transformed points," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[24] Q. Huang, W. Wang, and 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.

[25] J. Li, B. M. Chen, and G. H. Lee, "So-net: Self-organizing network for point cloud analysis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9397–9406.

[26] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: deep hierarchical feature learning on point sets in a metric space," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[27] Y. Ma, Y. Guo, Y. Lei, M. Lu, and J. Zhang, "3DMAX-Net: A multi-scale spatial contextual network for 3D point cloud semantic segmentation," in *24th International Conference on Pattern Recognition (ICPR)*. IEEE, 2018, pp. 1560–1566.

[28] X. Ye, J. Li, H. Huang, L. Du, and X. Zhang, "3D recurrent neural networks with context fusion for point cloud semantic segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 403–417.

[29] M. Jiang, Y. Wu, T. Zhao, Z. Zhao, and C. Lu, "Pointsift: a sift-like network module for 3D point cloud semantic segmentation," *arXiv preprint arXiv:1807.00652*, 2018.

[30] A. Komarichev, Z. Zhong, and J. Hua, "A-cnn: annularly convolutional neural networks on point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7421–7430.

[31] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "Spidercnn: deep learning on point sets with parameterized convolutional filters," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 87–102.

[32] Y. Zhang and M. Rabbat, "A graph-cnn for 3D point cloud classification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 6279–6283.

[33] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions On Graphics (TOG)*, vol. 38, no. 5, pp. 1–12, 2019.

[34] K. Zhang, M. Hao, J. Wang, C. W. de Silva, and C. Fu, "Linked dynamic graph cnn: learning on point cloud via linking hierarchical features," *arXiv preprint arXiv:1904.10014*, 2019.

[35] G. Te, W. Hu, A. Zheng, and Z. Guo, "Rgcnn: Regularized graph cnn for point cloud segmentation," in *Proceedings of the 26th ACM International Conference on Multimedia*, 2018, pp. 746–754.

[36] C. Chen, L. Z. Fragonara, and A. Tsourdos, "Gapnet: graph attention based point neural network for exploiting local feature of point cloud," 2019. [Online]. Available: https://arxiv.org/abs/1905.08705

[37] T. Rabbani and F. Heuvel, "Efficient hough transform for automatic detection of cylinders in point clouds," *Proceedings of the International Society for Photogrammetry and Remote Sensing Laser Scan Workshop*, vol. 36, 01 2005.

[38] Y.-H. Jin and W.-H. Lee, "Fast cylinder shape matching using random sample consensus in large scale point cloud," *Applied Sciences*, vol. 9, no. 5, p. 974, 2019.

[39] T.-T. Tran, V.-T. Cao, and D. Laurendeau, "Extraction of cylinders and estimation of their parameters from point clouds," *Computers & Graphics*, vol. 46, pp. 345–357, 2015.

[40] K. Himri, P. Ridao, and N. Gracias, "Underwater object recognition using point-features, bayesian estimation and semantic information," *Sensors*, vol. 21, no. 5, p. 1807, 2021.

[41] M. Martin-Abadal, M. Piñar-Molina, A. Martorell-Torres, G. Oliver-Codina, and Y. Gonzalez-Cid, "Underwater pipe and valve 3D recognition using deep learning segmentation," *Journal of Marine Science and Engineering*, vol. 9, no. 1, p. 5, 2020.

[42] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3. IEEE, 2004, pp. 2149–2154.

[43] M. M. M. Manhães, S. A. Scherer, M. Voss, L. R. Douat, and T. Rauschenbach, "Uuv simulator: a gazebo-based package for underwater intervention and multi-robot simulation," in *OCEANS MTS/IEEE Monterey*. IEEE, 2016, pp. 1–8.

[44] M. Sánchez, J. L. Martínez, J. Morales, A. Robles, and M. Morán, "Automatic generation of labeled 3D point clouds of natural environments with gazebo," in *IEEE International Conference on Mechatronics (ICM)*, vol. 1. IEEE, 2019, pp. 161–166.

[45] S. Reitmann, L. Neumann, and B. Jung, "Blainder—a blender ai add-on for generation of semantically labeled depth-sensing data," *Sensors*, vol. 21, no. 6, p. 2144, 2021.

[46] S. R. Richter, V. Vineet, S. Roth, and V. Koltun, "Playing for data: ground truth from computer games," in *European Conference on Computer Vision*. Springer, 2016, pp. 102–118.

[47] M. Müller, V. Casser, J. Lahoud, N. Smith, and B. Ghanem, "Sim4cv: a photo-realistic simulator for computer vision applications," *International Journal of Computer Vision*, vol. 126, no. 9, pp. 902–919, 2018.

[48] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: high-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics*. Springer, 2018, pp. 621–635.

[49] A. Inzartsev, G. Eliseenko, M. Panin, A. Pavin, V. Bobkov, and M. Morozov, "Underwater pipeline inspection method for auv based on laser line recognition: simulation results," in *IEEE Underwater Technology (UT)*, 2019, pp. 1–8.

[50] R. Schnabel, R. Wahl, and R. Klein, "Efficient ransac for point-cloud shape detection," *Computer Graphics Forum*, vol. 26, pp. 214–226, 06 2007.

[51] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: richly-annotated 3D reconstructions of indoor scenes," 2017. [Online]. Available: https://arxiv.org/abs/1702.04405

[52] C. Choy, J. Gwak, and S. Savarese, "4d spatio-temporal convnets: Minkowski convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3075–3084.