3D and 2D Object Detection in LiDAR Point Cloud

Keyhan Azarjoo
Teesside University
Middlesbrough, United Kingdom

K.azarjoo@tees.ac.uk

B1674080

AI has become an integral part of everyday life, significantly impacting various aspects of human existence. Among the rapidly evolving fields within AI, machine learning and computer vision hold substantial importance. In this regard, this study focuses on the development of a 3D digital version of an existing asset using LiDAR scanner technology and implementation of 3D and 2D object detection. The purpose of this project is to enable effective monitoring and management of the asset. To achieve this, deep learning techniques were employed for object detection in a point cloud dataset obtained from an indoor space at Teesside University. The research involved the use of RANSAC and DBSCAN clustering methods for object segmentation, followed by the implementation of the Point-Net neural network model for 3D object detection and YOLO V3 algorithm for 2D object detection. The model for 3D detection was trained using samples from the ModelNet10 datasets and for YOLO we just used pretrained model. The result is a 3D digital twin of the asset with detected objects.

Keywords: Object Detection, Deep Learning, LiDAR, Image Processing, Point Cloud, Neural Network

I. Introduction

The field of Artificial Intelligence (AI) has witnessed significant advancements in recent years, permeating various facets of contemporary life and becoming an indispensable component of modern technology. The continuous refinement of AI methodologies has led to the seamless integration of intelligent algorithms into everyday devices, endowing them with enhanced functionality and heightened productivity. Notably, the emergence of Deep Learning as a prominent subset of machine learning has opened up new frontiers, enabling humans to achieve remarkable feats, such as the development of autonomous vehicles.

Within the realm of AI, computer vision plays a pivotal role, leveraging sophisticated object detection algorithms to empower machines with the ability to discern and differentiate objects from their surrounding environments. By harnessing the potential of LiDAR scanners and point cloud technology, three-dimensional (3D) digital replicas of assets, objects, and environments can be generated, facilitating a comprehensive understanding of spatial structures. In the context of object detection, these 3D environments serve as a foundation for locating and identifying specific objects. While existing research predominantly focuses on the conversion of two-dimensional (2D) images into 3D models, this paper takes a distinct approach by employing a reverse methodology, transforming 3D data into 2D images to enhance the accuracy of object detection. This approach involves refining the quality of point cloud data to optimize object detection outcomes.

The primary objective of this paper is to investigate the efficacy of employing a Matterport scanner to capture detailed scans of indoor assets, which are subsequently subjected to clustering methodologies for effective segmentation. We employ a combination of 3D and 2D object detection techniques to identify and categorize objects within the scanned environment, thereby providing valuable insights into the utilization of advanced algorithms in the realm of object detection.

II. Literature Review

The field of machine learning and computer vision is an ever-evolving area of research, with the goal of imparting computers with human-like abilities to sense and understand data and make decisions based on past and present outcomes. In this context, object detection has emerged as a crucial aspect of various practical domains, such as autonomous driving, robot vision, and video surveillance [1].

The proliferation of publications centred on "object detection" during the last two decades, as shown in Fig.1, underscores its growing relevance and applicability in modern-day scenarios [2]. The research presented in [3] introduces an automated system that leverages orientations to detect human faces from images and video, utilizing the software libraries TensorFlow and Open Pose for object detection and computer vision. Furthermore, traffic detection models utilize a range of machine learning algorithms, including convolutional neural networks, recurrent neural networks, long short-term memory, gated recurrent units, and Bayesian networks. In intelligent environments, sensor data is captured and analysed for forecasting and subsequent decision-making [4, 5].

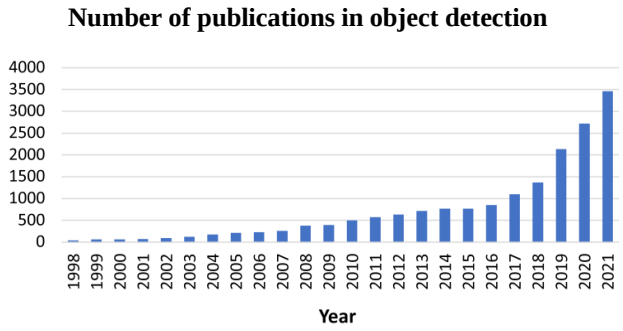

Fig. 1. Number of publications in object detection from 1998 to 2021.

Convolutional neural networks are particularly well-suited for feature extraction, a crucial task for successful object detection, without any loss of information [6]. Recent advances in the combination of convolutional neural networks (CNN) with GPU-accelerated deep-learning frameworks have introduced a new perspective to the field of object detection [7], [8], leading to significant improvements in performance standards, as demonstrated by prominent works such as Overfeat [9], R-CNN [10], Fast R-CNN [11], Faster R-CNN [12], R-FCN [13], SSD [14], and YOLO [15]–[16]. These methods have been recognized as top performers in various world-renowned competitions,

including the VOC PAS-CAL Challenge [17], COCO [18], ImageNet Object Detection Challenge [19], and Google Open Images Challenge [20].

Another critical aspect of computer vision is 3D object detection, which leverages LiDAR and cameras to perceive the surrounding environment. However, the misalignment between images and point clouds presents a significant challenge in 3D object detection [21] – [26]. Most of the 3D perception research has focused on real-time, onboard scenarios, and only takes into account sensor input from the current or a few previous frames [22].

In data mining, pattern recognition, and machine learning, clustering is a critical technique. Traditional clustering algorithms are no longer sufficient to meet the demands of big data analysis due to the exponential growth of data. Density-based clustering algorithms, such as DBSCAN, offer a solution by enabling the clustering of arbitrarily shaped datasets with unknown distributions. The simplicity and effectiveness of DBSCAN have made it a widely adopted and efficient method for data clustering analysis [23]. Personalized clustering using DBSCAN can achieve higher homogeneity and diversity on datasets with non-uniform density, broad value ranges, and gradually sparser data [24].

The identification of building roof planes or specific shapes from LiDAR data has gained widespread attention, with numerous studies utilizing the random sample consensus (RANSAC) algorithm. However, RANSAC has been found to produce full planes that encompass non-roof features, resulting in outliers beyond the roof boundary. In [25], the authors enhanced the RANSAC (I-RANSAC) algorithm to effectively remove these outliers, highlighting its potential as a more effective approach to building roof plane extraction from LiDAR data.

This paper presents an exploration of various methods, including RANSAC, DBSCAN, YOLO, and CNN, for achieving improved results in object detection. To enhance the quality of objects and further improve detection outcomes, several algorithms were employed.

III. METHODOLOGY

This paper employs quantitative methods to investigate the efficacy of object detection techniques in the context of indoor asset scanning. To this end, we utilized a 3D Matterport scanner to capture a detailed scan of the indoor asset at Teesside university(fig.2), followed by the implementation of cleaning methods to eliminate outliers. An unsupervised learning approach was then employed to segment the objects in the scan, and an algorithm was developed to increase the point cloud's quality by augmenting the number of points.

Two distinct approaches were then pursued to detect objects within the scan. In the first approach, a deep learning model was implemented for 3D object detection, while in the second approach, each 3D object was converted into 2D images, and a deep learning model was employed for 2D object detection. The results for both models were analyzed and discussed.

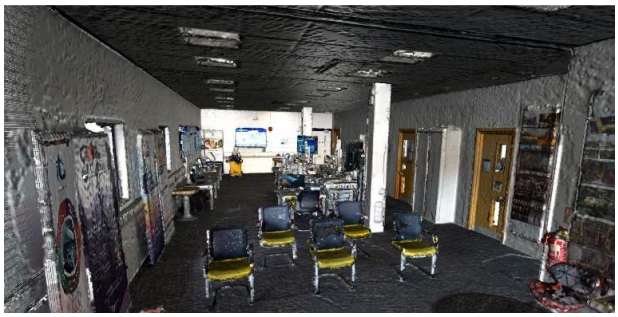

Fig.2 Point cloud captured from IDTC building at Teesside University.

A. Pre-Processing

Since point cloud data acquired by LiDAR scanners are often affected by noise and outliers, we propose a method to remove these unwanted elements from the dataset.

To begin with, we employed the Open 3D library to pre-process the point cloud data. We first applied a statistical outlier removal method to detect and remove the points that deviate significantly from the average or typical behaviour of the surrounding points. Specifically, we set the nb_neighbors parameter to 20, which determines the number of neighbors to consider when computing the statistical properties of each data point. We also set the std_ratio parameter to 2, which determines the threshold for outlier detection based on the standard deviation of the data. This procedure effectively eliminates the noise and outliers in the point cloud data, thereby enhancing the accuracy of the object detection model (fig.3).

In addition to statistical outlier removal, we also implemented a radius-based outlier removal method to further clean the point cloud data. We utilized the remove_radius_outlier function, which removes outliers from a point cloud within a specified radius of each point. The nb_points and radius parameters were used to determine which points are considered neighbors and included in the outlier detection process. In our implementation, we set the nb_points parameter to 20 and the radius parameter to 0.05. These parameters ensure that the method only considers points within 0.05 units of distance from each point when computing the statistical properties and identifying outliers.

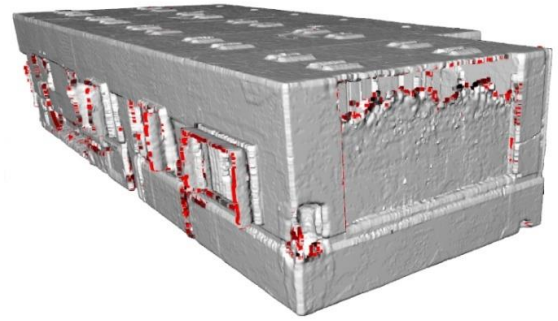

Fig.3 Point cloud with detected outliers.

B. Classification and Segmentation

1) RANSAC

After the removal of outliers from the point cloud, it is imperative to separate each object from others as comprehensively as possible. Given that objects in the real world are connected to the floor, roof, or walls, the detection of flat surfaces becomes necessary, as previously mentioned. To achieve this goal, the authors employed the technique of semantic segmentation in 3D. Semantic segmentation in 3D involves dividing a 3D scene or object into various parts or segments, where each segment corresponds to a specific object or region of interest. In semantic segmentation, the objective is to label every point in a 3D space with a class label that represents the semantic meaning of that point. For example, in a 3D scan of a room, semantic segmentation could be utilized to label each point as floor, wall, ceiling, furniture, or other objects.

Several approaches to semantic segmentation exist, including edge-based clustering, region-growing clustering, model-fitting clustering, and unsupervised clustering. While the first two models are typically employed in image segmentation, model-fitting clustering is the preferred model for searching a specific shape in a point cloud. In this study, unsupervised clustering was deemed unsuitable for the purpose of finding flat surfaces, and model fitting clustering was preferred. The crux of model fitting is matching the point clouds to different primitive geometric shapes, making it a popular shape detection or extraction method.

Different model fitting techniques exist, most of which are founded on two classical algorithms: RANSAC and Hough Transform. RANSAC (Random Sample Consensus) is a robust method used in 3D segmentation to identify outliers from a point cloud. It is often applied in applications such as object recognition and registration, where it is necessary to identify a set of points that belong to a common geometric structure or object. The fundamental concept behind RANSAC is to randomly choose a subset of points from the point cloud and then fit a model to those points. The model is then used to predict the location of the remaining points in the point cloud. Points that are consistent with the model are considered inliers and are added to the subset, while points that are inconsistent with the model are considered outliers.

This process is repeated for a fixed number of iterations, and the subset with the most inliers is considered the best fit for the model. Once the model is determined, it can be utilized to segment the point cloud into regions based on the properties of the model, such as planar or cylindrical surfaces. In this project, RANSAC has been used exclusively to detect flat surfaces and separate them from other objects in the point cloud (fig 3 and 4).

The initial stage of our approach involved identifying five distinct flat surfaces, including the walls, roof, and floor in (fig.2) .To achieve this outcome, we fine-tuned several parameters, including max_plane_idx=5, ransac_n=3, distance_threshold=0.08, and num_iterations=15000, which were selected based on our extensive experimentation with the dataset. Fig.4 shows the result of point cloud after implementing RANSAC. Following the segmentation process, we performed further cleaning on the point cloud data, utilizing the methods.

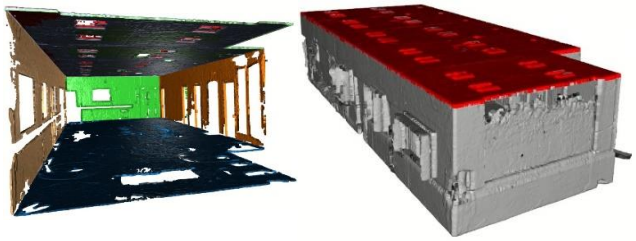

Fig.3 Finding flat surfaces.

Fig.4 Point cloud after implementing RANSAC

2) DBSCAN

Following the implementation of RANSAC, the subsequent step necessitates the clustering of the remaining points. Given that the point cloud lacks labels for objects, unsupervised clustering is a suitable approach.

Unsupervised clustering is a machine learning technique that groups similar data points in a dataset without any prior knowledge of their respective labels or groups. In this type of clustering, the algorithm identifies patterns or similarities among the data points, grouping them together based on these similarities.

In 3D unsupervised clustering, the dataset is comprised of data points with three dimensions or features, such as x, y, and z coordinates. The clustering algorithm analyzes the data and identifies patterns or similarities among the data points in three-dimensional space, grouping them together based on these similarities. Two prominent models for 3D unsupervised clustering are DBSCAN and K-means. DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a clustering algorithm used for unsupervised clustering in 3D space. DBSCAN forms clusters based on the density of data points, rendering it useful for identifying clusters of arbitrary shapes and sizes.

To apply DBSCAN in 3D space, the algorithm first selects a random data point and identifies all other data points within a specified radius or distance from that point. These data points are considered part of the same cluster. The algorithm then expands the cluster by recursively adding neighboring points that are also within the specified radius or distance.

Two key parameters in DBSCAN are the minimum number of points required to form a cluster (minPts) and the maximum distance or radius (eps) within which neighboring points are considered part of the same cluster. These parameters are determined based on the specific characteristics of the data being analyzed. DBSCAN is particularly adept at identifying clusters of varying densities in 3D space and can also pinpoint noise points that do not belong to any cluster. Its applications range from pattern recognition in medical imaging and geological data analysis to object detection in 3D point cloud data.

Unlike k-means, DBSCAN is a density-based clustering algorithm that does not necessitate the specification of the number of clusters in the data. It can identify arbitrarily shaped clusters, making it well-suited to LiDAR point cloud data. Upon applying DBSCAN to the dataset, the result comprised 203 classes. We used eps=0.029 and min_points=25 as the DBSCAN parameters, which were chosen based on experimenting with different parameter values and observing the results. Fig.5 displays some of the identified objects.

Fig.5 Some classes after implementing DBSCAN.

C. quality improvement

After the point cloud data was classified, a function was implemented to separate all classes into an array and filter them to find those with more than 2048 points. Since the aim of this project was to detect larger objects such as monitors and chairs, groups with less than 2048 points were deemed less useful and were separated from the dataset. The remaining groups were sorted based on the number of points they contained, with larger objects appearing first in the list. After separating the groups with more than 2048 points, there were 47 groups of points that were to be used for model prediction. However, for the classes that had less than 2048 points, an algorithm was developed to increase the number of points in the point cloud, effectively creating larger objects with better quality.

The algorithm for increasing the number of points involved finding points that were almost in the same X and Y coordinate and then dividing the point cloud into 250*250 parts in the X and Y directions. The algorithm then measured the distance in the Z dimension for each point in the group and inserted a new point between points that were close to each other and had similar colors. This algorithm can be repeated for X and Y dimensions instead of Z, and it could be iterated as many times as necessary to create larger and more detailed point clouds. This function was useful in converting the 3D point cloud data into 2D pictures, especially when dealing with point clouds that had few points.

D. 3D Object Detection

1) DataSet

The ModelNet 10 dataset is a well-established subset of the larger ModelNet dataset, a prominent collection of 3D models commonly used in machine learning research. Notably, the ModelNet 10 dataset is constructed from CAD models, which are digital representations of 3D objects generated using computer-aided design software. This specific subset comprises ten object categories, namely chairs, desks, dressers, monitors, nightstands, sofas, tables, toilets, wardrobes, and bookshelves, and serves as a benchmark for evaluating algorithms for 3D object classification and retrieval. Typically, these models are expressed as meshes, defined by vertices, edges, and faces that define the shape of the object. In the context of processing Lidar point cloud data, it is necessary to convert these models from meshes to points, rather than the other way around. Thus, CAD models are transformed to points to train models like PointNet for the processing of point cloud data.

The process of converting meshes to points entails sampling a fixed number of points from the surface of the mesh to create a point cloud representation of the object. The resulting point cloud is then utilized to train the model to recognize and classify the object or environment based on features extracted from the point cloud data. Upon completion of the mesh-to-point transformation, the data can be used to train models like PointNet, which are specifically engineered to process and classify point cloud data. The trained model can then be utilized for various applications, including object detection, segmentation, and 3D shape recognition.

2) POINT NET Neural Network

In the context of Matterport scanning, where the point cloud data is non-sequential, many deep learning models for object detection and classification are not applicable. However, PointNet is a suitable choice, as it can handle non-sequential point cloud data and has achieved high accuracy on the ModelNet40 dataset. In fact, the accuracy of PointNet on the ModelNet40 dataset was 89.2%.

PointNet is a deep learning architecture that was specifically designed to process point clouds for tasks such as object classification, segmentation, and detection. It has also been used for point cloud clustering, which involves grouping together points that belong to the same object or surface in the scene. The PointNet architecture consists of two main components: a shared Multi-Layer Perceptron (MLP) and a symmetric function that aggregates information across points (fig.6). The shared MLP is applied to each point in the input point cloud individually to extract local features, and the resulting features are then aggregated into a global feature vector that summarizes the entire point cloud. The symmetric function is applied to the global feature vector and produces a single vector that captures the

important characteristics of the point cloud, such as object shape, size, and orientation.

PointNet can be trained end-to-end, meaning that the entire architecture is learned directly from the input data and the corresponding labels. This is accomplished using standard backpropagation and gradient descent algorithms. The architecture can also be easily extended to handle more complex point cloud data, such as those with varying densities or sizes, by modifying the MLP or the symmetric function.

To train a model for object detection using PointNet, we used the ModelNet10 dataset. The input shape of the model was (None, 2048, 3), meaning that the model can handle inputs, where each input can have 2048 points. The PointNet model that we trained consists of a 1D convolution layer that learns 32 filters, followed by batch normalization and ReLU activation. This is followed by another 1D convolution layer with 64 filters, again followed by batch normalization and ReLU activation. The third convolution layer has 512 filters, followed by batch normalization and ReLU activation. Then, a global max pooling layer is applied to reduce the dimensionality of the output to 512. The output is then fed into two fully connected layers, each with batch normalization and ReLU activation. The final output layer has 10 units, corresponding to the 10 possible classes in the dataset.

In addition, the model includes a transformation network, which takes the output of the last fully connected layer and applies a series of convolutional and fully connected layers to generate a 3x3 transformation matrix. This matrix is then applied to the input point cloud to generate a transformed point cloud, which is fed back into the model for further processing. The PointNet architecture is designed to learn a hierarchical representation of the input point cloud data, allowing it to accurately classify the point cloud into one of the 10 possible classes. Its end-to-end training approach and ability to handle non-sequential point cloud data make it a powerful tool for a variety of computer vision tasks.

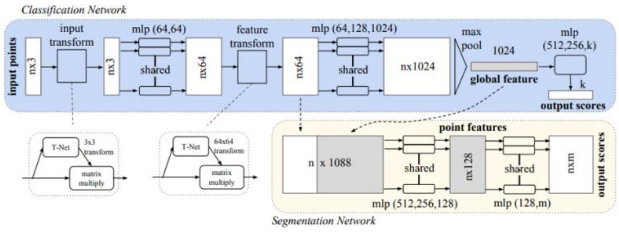

Fig.6 PointNet architecture [26]

E. 2D Object Detection

1) Converting 3D to 2D

The second approach we explored for object detection involved converting the 3D objects into 2D images and utilizing 2D object detection models. We experimented with two different approaches in this regard.

First, we attempted to squeeze the point cloud data and convert the 3D points into images from various angles. We developed a function to compress the point cloud and generate corresponding images. However, the resulting images proved to be of low quality and lacked usefulness for our purposes (fig.7). Nonetheless, we recognized the potential for enhancing image quality by increasing the number of points and applying AI algorithms. Given our time constraints, we sought alternative solutions.

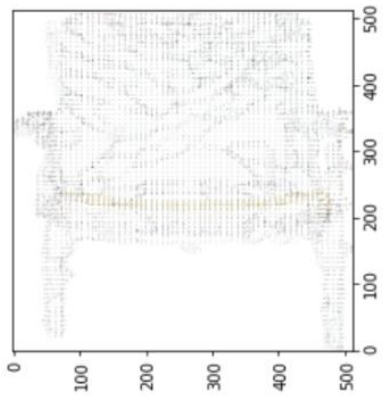

Fig.7 Result for squeezing points.

Our second approach involved capturing screenshots of the point clouds from different angles. To achieve this, we designed a function to display the 3D model from various perspectives, capture a screenshot, and then close the display window. As a result, we obtained 64 images for each class of point cloud data (fig.8).

Fig.8 Result for taking screenshots of one object.

2) YOLO V3

For the 2D object detection task, we employed the YOLO (You Only Look Once) algorithm for detecting screenshots of objects. YOLO is a real-time object detection algorithm which is capable of detecting multiple objects in an image or video frame and drawing bounding boxes around them.

The YOLO algorithm operates by dividing the input image into a grid of cells and predicting bounding boxes and class probabilities for each cell. It performs simultaneous predictions of class probabilities and bounding boxes for multiple objects in a single forward pass of the neural network. This approach makes YOLO faster compared to traditional object detection algorithms that employ a sliding window technique.

YOLO utilizes a convolutional neural network (CNN) architecture to extract relevant features from the input image and predict both object classes and bounding boxes. It is trained on extensive datasets such as COCO (Common Objects in Context) and PASCAL VOC (Visual Object Classes) to learn a wide range of object categories. In our study, we opted to utilize the pre-trained YOLOv3 weights and configuration instead of developing a custom model. We downloaded the necessary data from the YOLO website and constructed a model based on that information. This pre-trained model had been trained on a diverse range of 80 object classes from the COCO dataset.

Post-prediction, we filtered out objects that we knew would not be present in our point cloud dataset. Examples of such objects included boats, sheep, and other irrelevant classes.

IV. Results

A. 3D Object Detection

Upon training the neural network model, we proceeded to provide the model with the necessary classes for prediction. As a result, we achieved successful detection of 6 out of 8 chairs present in the point cloud dataset (fig.9). Given our focus on chair prediction, the results obtained for this particular object category were satisfactory. However, when it came to other objects, for which the model had not been specifically trained, accurate prediction was hindered. In such cases, the model tended to detect objects as the closest shape approximation to them. For instance, the model might classify a closet as a bed. These limitations posed challenges that ultimately impacted the overall accuracy of the 3D object detection process.

Fig.9 3D Detected object in point cloud

B. 2D Object Detection

In the case of 2D object detection, we provided the model with a set of 64 images (fig.8) captured from different angles for each class of point cloud data. Through this approach, the model successfully identified 2 monitors and 5 chairs accurately. However, it also produced incorrect detections for 3 objects. Fig.10 illustrates a successful example of chair detection, achieving an impressive accuracy rate of 97%.

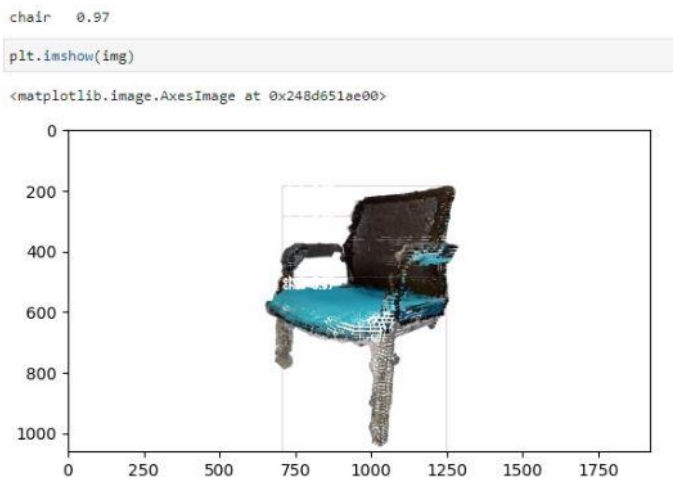

Fig.10 A sample of 2D detected object in point cloud.

V. Limitation

There are several limitations that need to be acknowledged in our study:

Limited availability of 3D object datasets: One major limitation is the scarcity of comprehensive datasets for 3D objects. If an object was not included in the training data, our model incorrectly classify it as the nearest similar object. This issue lead to misclassifications and inaccurate predictions, as depicted in (fig.11).

Scanner quality and sample limitations: The quality of the scanner used and the limited number of samples can also affect the accuracy of our predictions. In some cases, the scanner may produce low-quality scans, resulting in erroneous predictions, as shown in (fig.12).

Unsupervised segmentation challenges: The unsupervised segmentation technique employed may not always accurately separate objects. This can lead to instances where objects are merged or wrongly segmented, as illustrated in (fig.13).

Meaningless objects due to larger object context: Some objects may be part of a larger object or scene context, making their individual significance unclear. This can lead to objects being rendered meaningless or indistinguishable, as shown in (fig.14 and 15).

Furthermore, it is worth noting that our focus was primarily on predicting chairs. Although the results for chairs were promising, we encountered challenges due to the limited number of chair instances in the point cloud dataset. Additionally, there were 40 other object classes present, such as tables, closets, monitors, and more. As a result of semantic segmentation, some objects were connected together, forming complex groups. When these groups were fed into the model, it struggled to identify the individual objects within them.

In the context of 2D object detection using YOLO V3, our capabilities were restricted to the objects the model was

trained with. Moreover, connecting the predicted objects in the images to their corresponding points in the point cloud proved to be a challenging task, given that the images were obtained through screenshots and lacked direct correspondences to the point cloud data.

Fig.11 Detected as a bed.

Fig.12 Low Quality (chair).

Fig.13 Segmenting 3 objects as one.

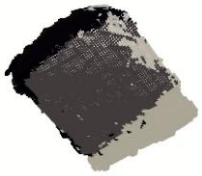

Fig.14 Meaning less points.

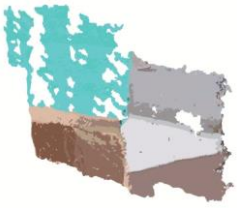

Fig.15 Meaning Less points.

VI. Ethical Issues

1. Privacy concerns: The creation of 3D digital twins raises privacy concerns as it involves capturing detailed information about individuals and environments without their knowledge or consent.
2. Data ownership and control: Clear guidelines and agreements should be established to determine who owns and controls the 3D data used for digital twins, as well as how it is shared with third parties.
3. Bias and fairness: Object detection algorithms can exhibit biases if the training data contains biases, leading to unfair outcomes. Addressing bias and ensuring fairness in the algorithms is essential.
4. Informed consent: Obtaining informed consent from individuals or property owners is important when collecting data for creating 3D digital twins, providing them with clear information and understanding of the purpose and implications.
5. Security risks: Storing and processing large volumes of 3D data can pose security risks, such as unauthorized access and data breaches. Strong security measures should be implemented to protect the data.

Addressing these ethical issues is crucial to ensure responsible and ethical use of 3D digital twins and object detection algorithms, mitigating potential harm and promoting fairness and privacy.

VII. Recommendations and Future Work

Based on our findings, converting the point cloud data into images appears to be a more promising approach for object detection within 3D digital twins. There are several avenues to explore in future research and development.

Firstly, in the 2D object detection approach, we can consider utilizing different pre-trained models or creating our own dataset to train a neural network model. Options such as R-CNN or Transformer Neural Networks could be investigated for improved object detection accuracy. Furthermore, incorporating NLP techniques and connecting detected objects to text could provide additional context and insights within the digital twin.

Alternatively, for 3D object detection, the application of Transformer Neural Networks and attention mechanisms shows promise. Training a new model specifically designed for 3D objects using the Transformer architecture could potentially yield better results. However, it should be noted that as this is a relatively new project, the accuracy of the model may not meet initial expectations. Nonetheless, Transformer Neural Networks have demonstrated impressive performance in various domains such as image and voice detection.

One crucial aspect that needs to be addressed in future work is the scarcity of available datasets for 3D object detection. Efforts should be directed towards the creation of comprehensive datasets that encompass a wide range of object classes and variations. This would contribute to more accurate and robust training of object detection models.

In summary, further exploration of the 2D approach with different models and the development of a specialized Transformer Neural Network model for 3D objects hold promise for improving object detection within 3D digital twins. Additionally, expanding the available datasets and considering the integration of NLP techniques can enhance the overall accuracy and utility of the models.

VIII. Conclusions

In this study, a 3D digital twin of an asset has been successfully generated by employing various techniques. Our approach involved initial data cleaning followed by unsupervised learning methods to cluster and separate objects for detection. Then an algorithm has been developed to improve the quality of point clouds and increase the number of points for small objects. Finally, two distinct approaches, namely 2D and 3D object detection, are adopted to accurately identify objects within the digital twin.

For 3D object detection, we utilized the PointNet model, which was trained using the ModelNet10 dataset. Conversely, in the 2D object detection approach, we transformed each class of points into multiple images and

employed a pre-trained YOLO V3 model for prediction. Comparing the results obtained from both approaches, we observed that the accuracy of YOLO V3 surpassed that of PointNet.

In the case of PointNet, the accuracy of correct and incorrect predictions was found to be relatively close. This can be attributed to the limitations we encountered during the 3D object detection process, resulting in less accurate results. Conversely, YOLO V3, benefiting from the conversion of objects into images, addressed some of the limitations associated with 3D object detection. Notably, YOLO V3 successfully detected two televisions, which were part of a larger object class that could not be identified by PointNet.

It is important to acknowledge the existing limitations within the 3D object detection model, which affected the accuracy of our results. However, the findings from the YOLO V3 approach demonstrated higher accuracy, showcasing its potential in overcoming certain limitations of 3D object detection. Further research and development in this area are essential to enhance the accuracy and effectiveness of both approaches, ensuring more reliable object detection within 3D digital twins.

ACKNOWLEDGMENT

We gratefully acknowledge the support and guidance of my first supervisor, Dr. Annalisa Occhipinti, and my second supervisor, Dr. Jawad Fayaz, throughout the course of this project. Their expertise and insights were invaluable in shaping our research direction and methodology.

We also express our gratitude to Teesside University for granting us access to their state-of-the-art equipment, which was essential in carrying out our experiments and analyses.

Furthermore, we would like to extend our appreciation to Nicander Ltd for their collaboration and support in this project. Their valuable contributions and resources enabled us to achieve the objectives of this research.

Finally, we would like to thank all those who have provided us with their time, knowledge and support, which helped us to complete this project and produce this conference paper.

REFERENCES

[1] Kerle, N., Gerke, M. and Lefèvre, S., 2019. GEOBIA 2016: Advances in Object-Based Image Analysis—Linking with Computer Vision and Machine Learning. Remote sensing, 11(10), p.1181.

[2] Zou, Z., Chen, K., Shi, Z., Guo, Y. and Ye, J., 2023. Object detection in 20 years: A survey. Proceedings of the IEEE.

[3] Sebe, N., Lew, M.S., Sun, Y., Cohen, I., Gevers, T. and Huang, T.S., 2007. Authentic facial expression analysis. Image and Vision Computing, 25(12), pp.1856-1863.

[4] Kaloskampis, I., 2013. Recognition of complex human activities in multimedia streams using machine learning and computer vision (Doctoral dissertation, Cardiff University).

[5] Al-Badi, A., Tarhini, A. and Khan, A.I., 2018. Exploring big data governance frameworks. Procedia computer science, 141, pp.271-277.

[6] Zhang, F., Li, W., Zhang, Y. and Feng, Z., 2018. Data driven feature selection for machine learning algorithms in computer vision. IEEE Internet of Things Journal, 5(6), pp.4262-4272.

[7] Hinton, G.E., Osindero, S. and Teh, Y.W., 2006. A fast learning algorithm for deep belief nets. Neural computation, 18(7), pp.1527-1554.

[8] Hinton, G.E. and Salakhutdinov, R.R., 2006. Reducing the dimensionality of data with neural networks. science, 313(5786), pp.504-507.

[9] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus and Y. LeCun, "Overfeat: Integrated recognition localization and detection using convolutional networks", CoRR, 2013.

[10] Girshick, R., Donahue, J., Darrell, T. and Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580-587).

[11] Girshick, R., 2015. Fast r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 1440-1448).

[12] . Ren, K. He, R. Girshick and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks", Advances in Neural Information Processing Systems 28, pp. 91-99, 2015.

[13] Dai, J., Li, Y., He, K. and Sun, J., 2016. R-fcn: Object detection via region-based fully convolutional networks. Advances in neural information processing systems, 29.

[14] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y. and Berg, A.C., 2016. Ssd: Single shot multibox detector. In Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14 (pp. 21-37). Springer International Publishing.

[15] Redmon, J., Divvala, S., Girshick, R. and Farhadi, A., 2016. You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).

[16] Redmon, J. and Farhadi, A., 2018. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.

[17] Everingham, M., Eslami, S.A., Van Gool, L., Williams, C.K., Winn, J. and Zisserman, A., 2015. The pascal visual object classes challenge: A retrospective. International journal of computer vision, 111, pp.98-136.

[18] Coco detection challenge (bounding box), [online] Available: https://competitions.codalab.org/competitions/20794.

[19] ImageNet. Imagenet object localization challenge, [online] Available: https://www.kaggle.com/c/imagenet-object-localization-challenge/.

[20] G. Research, Open images 2019-object detection challenge., [online] Available: https://www.kaggle.com/c/open-images-2019-object-detection/.

[21] Wang, C., Ma, C., Zhu, M. and Yang, X., 2021. Pointaugmenting: Cross-modal augmentation for 3d object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 11794-11803).

[22] Qi, C.R., Zhou, Y., Najibi, M., Sun, P., Vo, K., Deng, B. and Anguelov, D., 2021. Offboard 3d object detection from point cloud sequences. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 6134-6144).

[23] Deng, D., 2020, September. DBSCAN clustering algorithm based on density. In 2020 7th international forum on electrical engineering and automation (IFEEA) (pp. 949-953). IEEE.

[24] Canaz Sevgen, S. and Karsli, F., 2020. An improved RANSAC algorithm for extracting roof planes from airborne lidar data. The Photogrammetric Record, 35(169), pp.40-57.

[25] Li, S.S., 2020. An improved DBSCAN algorithm based on the neighbor similarity and fast nearest neighbor query. Ieee Access, 8, pp.47468-47476.

[26] Qi, C.R., Su, H., Mo, K. and Guibas, L.J., 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 652-660).