Eurographics Workshop on 3D Object Retrieval (2023)
S. Biasotti, M. Daoudi, U. Fugacci, G. Lavoué and R. C. Veltkamp (Editors)

Short Paper

synScan - A Large-Scale Dataset for Instance Level Recognition on Partial Scan Data

M. Bookhahn and F. Neumann

HTW Berlin, Germany

Abstract

Devices supporting depth-sensing technologies become more and more available, making it easier to access geometry-data driven services like 3D model or scene reconstruction. Utilizing these depth sensors, very large datasets have been created to enable deep learning for object detection and depth upsampling.
We want to tackle the task of instance level recognition (ILR), where 3D scans of objects can be searched against a database of CAD models based on embeddings of their geometry. The distinctive property of this retrieval task is the existence of only a single corresponding database shape for each query.
To the best of our knowledge all the existing datasets either lack in providing the exact CAD model correspondences or lack in scale and a variety of object categories.
Therefore, we introduce synScan, a large-scale dataset synthetically generated via physically plausible domain randomization (PPDR) of 3D scenes and object-centric scan trajectories with the goal to mimic real-world object scan scenarios with a variety of incomplete views and occlusions. We provide approximately 39,000 randomly sampled scenes, made from 9,400 different shapes with semantic per-point annotation. We train and test different ILR algorithms (e.g. PointNetVLAD, MinkLoc3Dv2) designed for place-recognition in self-driving cars on the dataset and validate our results on a smaller real-world dataset. Utilizing a rather simple data generation pipeline, we can show that deep learning methods trained on our synthetic dataset can successfully adapt to real-world scan data. In this manner, synScan helps to overcome the lack of labeled training data.

CCS Concepts

• ***Computing methodologies*** → ***Instance-based learning;***

1. Introduction

Services for reverse image search are adopted by millions of users nowadays. Likewise we want to establish similar search abilities based on 3D scans of scenes and components with depth-sensing devices. For such search services a high adoption potential can be foreseen in various industrial scenarios:

- Identification of highly complex warehouse goods in the context of logistics processes: Such use cases aim at the precise recognition of an unknown component within a stock of tens of thousands of components.
- Search for spare parts on site by service technicians: Such scenarios occur when repairing e.g., production systems, escalators or elevators where the article number of an urgently needed spare part on site has to be determined. Here, a 3D scan may speed up the search within an online database
- Identification of visual art work such as sculptures, reliefs, etc. within a catalog of stolen art objects

The aforementioned scenarios all have in common that the scan data supplied by the depth-sensor for any component is incomplete because they are either installed and not completely visible, heavily worn or partially destroyed. Consequently, the search algorithm has to recognize components based on fragments of their geometry. Using geometric data may also be more appropriate if textures are altered or information about them is missing.

To train neural networks for such tasks, a suitable dataset is needed which holds at least two different representations for each shape inside the dataset as illustrated in Fig. 1. The first representation is designated to be used in a query database and should represent the complete outer shell of a particular shape. The second is an arbitrary incomplete scan of that shape serving as search input. Given the huge number of real-world objects and their different labeled representations needed for this kind of dataset, only a synthetically generated dataset is really feasible.

In this paper, we make the following contributions:

- We introduce *synScan*[1], a synthetically generated large-scale dataset for 3D scan data, which mimics real-world scenarios with partial (incomplete) views resulting from scans with depth-sensing devices. It is especially designed for ILR. We provide ap-

DOI: 10.2312/3dor.20231149

Figure 1: *For instance level recognition on 3D data, the original instance of an object in a database is searched using a 3D scan of an object. A metric determines the distance* d *to all the shapes held in the database.*

proximately 39,000 randomly sampled scenes, made from 9,400 different shapes with semantic per-point annotation.
- Based on *synScan* we train different ILR algorithms (e.g. *PointNetVLAD* [UL18], *SOE-Net* [XXL*21], *MinkLoc3Dv2* [Kom22]) designed for place-recognition in self-driving cars on the dataset and evaluate their performance on real-world data.
- We show, that methods trained on our synthetic dataset, can successfully be used with real-world scan-data.

[1] https://github.com/mbookhahn/synScan

Our study also identified two interesting related topics that could provide valuable insights if further investigated.

- We can show that the *PointOE* module used within *SOE-Net* is not invariant against rotation.
- We demonstrate that pointcloud-based ILR can benefit from data augmentation, and we encourage further research into augmentation techniques based on these results.

2. Related Work

2.1. 3D Scan Datasets

In the context of scene understanding, the release of several indoor and outdoor RGB-D datasets over the past few years has started a series of research activities in the area of semantic segmentation, object detection, shape retrieval, depth upsampling and more.

Indoor datasets A very early contribution is the *NYU Depth Dataset V2* [NSF12], which provides 1,449 semantically labeled frames of 464 real-world indoor scenes. The dataset *RGB-D Object-to-CAD Retrieval* [PTL*18] utilizes 2,101 scans of objects from *SceneNN* [HPN*16] and *ScanNet* [DCS*17] and creates a shape retrieval challenge. The shape matching for retrieval was performed manually with 3,308 CAD models of 20 categories from *ShapeNetSem* [SCH15]. Recently, *ARKitScenes* [BCD*21] establishes the largest real-world indoor dataset for object detection and depth upsampling with 5,047 scans from 1,661 scenes captured with Apple's LiDAR sensor. None of the aforementioned real-world datasets provides the original structural similar ground truth shapes needed for ILR. Among other synthetic datasets, *SceneNet RGB-D* [MHLD17] is the closest to ours. *SceneNet RGB-D* randomly creates 16,895 scene configurations with sequential video trajectories and photo-realistic scene renderings. However, with 263 object categories our dataset provides a lot more diversity compared to its 42 categories. Unlike ours, it does not provide a dedicated object scanning trajectory. Thus, most of the generated depth frames contain only a small amount of object information compared to ours.

	ARKitScenes	*SceneNet RGB-D*	***synScan***
No. of unique shapes	unknown	23,354	9,400
No. of obj. categor.	17	43	263
No. of scene config.	1,661	16,895	39,000
No. of depth frames	$>$ 1M	$>$ 5M	$>$ 1.9M
Sample rate	$\sim$ 20 Hz	1 Hz	5 Hz
Model files provided	no	yes	yes
Object scanning trajectory	no	no	yes
RGB Texturing	Real	Photo-realistic	no

Table 1: *A comparison table of 3D scanning datasets.* ARKitScenes *[BCD*21] is a large real-world RGB-D indoor dataset captured with the iPad LiDAR sensor. It does not provide any 3D model files. SceneNet RGB-D [MHLD17] is a very large synthetic indoor RGB-D dataset, however the provided trajectories and low sample rates are not well suited for detailed object reconstruction and thus neither for ILR.*

Outdoor datasets *The Oxford RobotCar Dataset* [MPLN17] collects more than 20 TB of image, LiDAR and GPS data from over 1,000 km of driving. In various works researchers aimed at making place recognition more robust against environmental changes (e.g. snow, fog, rain, ...), by using pointcloud data from the LiDAR sensor.

2.2. Algorithms for Instance Level Recognition

The main challenge for instance level recognition is to develop a distance function with a low value for a pair of similar shapes and high for a pair of distinct shapes.

Image based algorithms The bag-of-word approach of VLAD was successfully applied to various tasks in ILR. *NetVLAD* [AGT*16] adopted this idea and proposed a metric learning deep network to predict a global VLAD descriptor end-to-end. It was trained for place recognition on the Google Street View Time Machine database, which collects user images taken at the same places at different times.

Pointcloud based algorithms

With the introduction of *PointNet* [QSMG17] and its successor *PointNet++* [QYSG17], performing deep learning directly on pointclouds has gained traction in recent years. Consequently, *PointNetVLAD* uses a combination of *PointNet* and *NetVLAD* to predict a VLAD descriptor for pointclouds end-to-end. *PCAN* develops this approach and adapts *PointNet++* to generate an attention map. In contrast, *LPD-Net* [LZS*19] relies on handcrafted local features and graph neural networks to extract local contextual information and aggregate it via *NetVLAD*. Based on *PointNetVLAD*, *SOE-Net* uses self-attention to learn long range context dependencies and combines local descriptor extraction and aggregation to enable one-stage training to generate a global descriptor. By first quantizing the input pointcloud to a sparse voxelized representation, *MinkLoc3Dv2* [Kom22] adapts Feature Pyramid Network to extract local features and then uses a Generalized-Mean pooling layer to aggregate local features into a global descriptor. This method achieves state-of-the-art performance on the place recognition task.

3. synScan and Real-World Validation Dataset

With our *synScan* dataset we want to promote the use of partial 3D scan data for object recognition in deep learning. Specifically, we aim at using consumer-grade sensors like Apple's LiDAR for instance level recognition, when no other data than a synthetic 3D shape collection is available.

The fundamental idea behind our approach is to automatically retrieve real-world-like incomplete object scans by creating arbitrary complex occlusions with synthetically generated cluttered scenes and virtual camera trajectories, which are physically plausible.

3.1. Acquiring Shapes with Metric Scaling

As already addressed by *SceneNet RGB-D*, many publicly available 3D models are provided with normalized or arbitrary mesh sizes and have no further information about their dimension in the real-world. Consequently, we use a special subset of *ShapeNet* [CFG*15], *ShapeNetSem* [SCH15], which provides scaling factors for 12,000 shapes from 270 classes, to retrieve reasonable metric dimensions. Taking into account that very different object sizes require an adaptive scanning strategy, we limit and filter *ShapeNetSem* and set the maximum extend for all shapes along all axes to 3 m.

3.2. Scene Generation

For a diverse dataset not only many different shapes are desirable, but a broad range of different scene configurations is required. This is because a wide variety of occlusions and views are essential. Similar to [MHLD17] we randomly initialize a physics simulation, where arbitrary shapes are dropped into a scene. Considering the close proximity needed for object scanning to gather densely distributed points, we neglect any kind of room layout. Instead, we rely on random scene configurations to create the foreground and background.

Especially cluttered scenes with complex occlusions are challenging for ILR. Thus, scene-defining parameters like the number of objects and space for initial positions are set to generate these cluttered scenes more likely and still guarantee short simulation times.

Each scene is initialized with 9 random objects and their initial position is uniformly sampled from a cube of size 3 by 3 m length and width and 10 m height. We choose each initial orientation to be completely random around all 3 axes. The objects positions are initialized sequentially. For each new position, an axes aligned bounding box is calculated, to efficiently check for intersections between meshes during initialization. If an overlapping bounding box is detected, a new random position is sampled. A scene generation has a maximum of 200 initialization attempts and will terminate if this number is exceeded.

Regarding the rather simple requirements of the described system-dynamics, our decision to use *bullet* respectively *pybullet* [CB21] for the simulation environment is based on its competitive speed, its built-in ray tracing and its popularity.

3.3. Virtual Camera Trajectory Generation

For ILR it is beneficial to collect as much structural information of the searched object as possible. This means gathering densely distributed points of the target from many different views. Thus, the target or parts of the target should be kept in the center of the sensor's field of view during object scanning. Nevertheless, with manually performed object scanning, varying levels of completeness are expected even with accurate instructions. Taking this into account, we extend the approach of [MHLD17] and use two randomly accelerated free-floating bodies to generate trajectories for the camera center and the point in space the camera is focusing on. Furthermore, we generate physical bounds to limit the movement of the two bodies as follows.

By limiting the movement of the focusing point to the inner of the aligned bounding box of the target, the target is likely to be inside the camera's field of view. At the same time different parts of the target are being focused during scanning. For a more dense sampling of the target, we limit the movement of the camera body

to be outside the aforementioned bounding box but inside another box. This bounding box is defined by an offset on all sides of the first bounding box. Additionally, both bodies cannot move through the floor plane, but only the camera body can collide with the rest of the static scene. We expect the influence of camera tilt to be rather low and keep the camera up axis always aligned with the gravity vector. From each trajectory we sample discrete views with a rate of 5 Hz.

Fig. 2 illustrates a few resulting examples of our dataset. Some complex occlusions can be seen for example in the table in the top row.

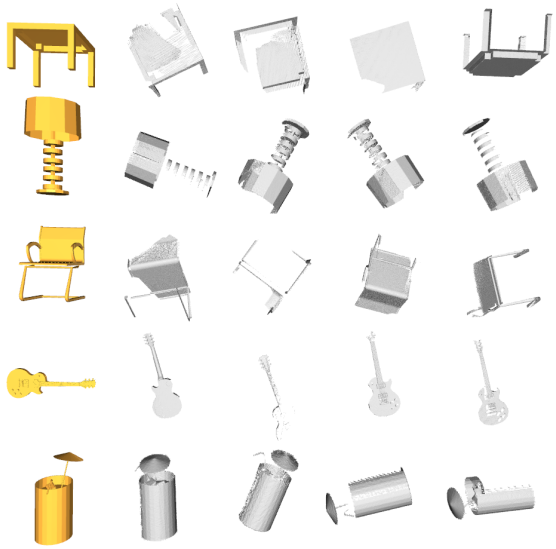

Figure 2: *Very left column: database shapes; Other columns: synthetic scans*

3.4. Preprocessing

Removal of inner structures

Scans cannot capture the inner structures of their associated CAD meshes. Therefore, they do not serve as useful information for object recognition. To avoid wasting descriptor capacity on these inner structures, we provide all database files from *ShapeNetSem* with removed inner structures. To remove inner structures, all meshes are split into their individual components (e.g. wheels of a car) by voxelizing the mesh into voxels of size 0.03 m with the help of *trimesh* [Daw] and cluster all connected voxels with *scipy*'s [VGO*20] multidimensional image label function. Now each vertex can be associated with the label of its belonging voxel. With these labels all components are separated accordingly. After having meshes split into their individual components, *ManifoldPlus* [HZG20] is used to remove inner structures.

Duplicates Filtering

Even though ShapeNetSem is composed of unique objects, many objects only differ in texture or only by a few vertices (cf. Fig. 4).

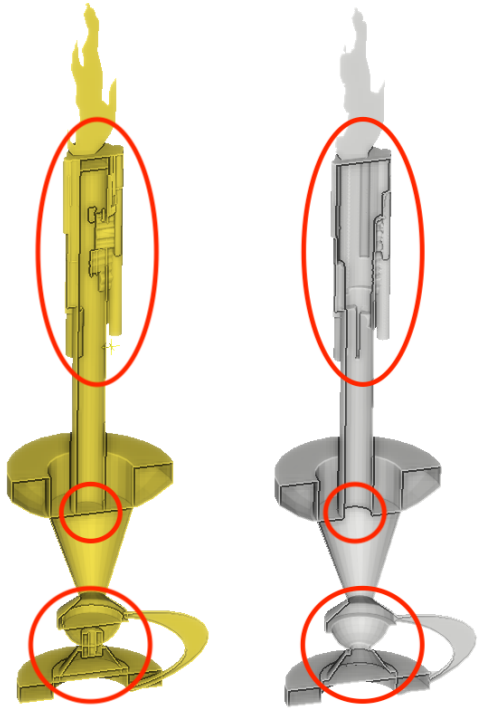

Figure 3: *Left: Original mesh with inner structures; Right: Mesh with removed inner structures*

Considering labels for positive and negative samples in training and evaluation, these duplicates have to be identified to create a clean dataset.

Figure 4: *Examples of duplicates found in* ShapeNetSem

Thanks to the model alignment of *ShapeNetSem*, we simply vox-

elize each shape and calculate its pairwise distance by applying a logical XOR operator on the occupancy grids. We then sum up all the differences. In total, for 266 shapes we found exact duplicates and for 444 we found very similar shapes.

3.5. Provided Data

It can be challenging to recognize an object from an incomplete scan representation, if only a small fraction of the structural information is gathered. This raises the question of what fraction of structural information is needed for recognition to succeed and how to define that fraction in this particular case? We want to provide two measures α and θ to help understand what to expect from our dataset.

Considering the legs of a table as an example for convex features, the legs are only a small part in terms of the size of the table. However, they provide essential structural information for it. An intuitive way to take this into account would be the ratio θ of the volume of the convex hull of the partial shape to the original complete shape.

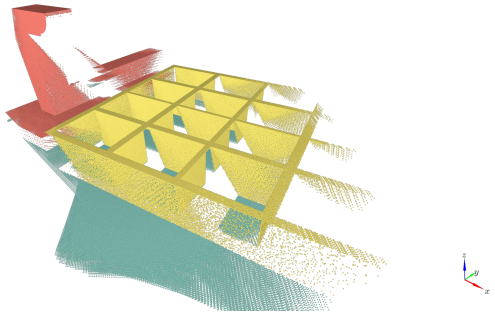

Figure 5: *Scan of a bookshelf.* $\alpha = 0.15$ | $\theta = 0.79$

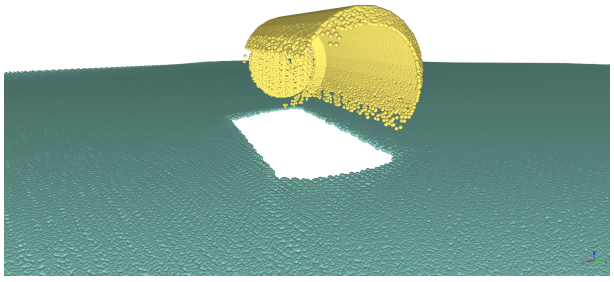

Figure 6: *Scan of a can.* $\alpha = 0.43$ | $\theta = 0.79$

$$\theta = \frac{V_{partial\_chull}}{V_{complete\_chull}} \tag{1}$$

Likewise the inner structures of a bookshelf (cf. Fig. 5) can be regarded as concave features, which hold essential structural information. If such a shape has only been seen from the back, θ could be close to 1.0 even if essential structural information from the inside is missing. For these concave features, the ratio α of the area of the partial shape to the area of the original complete shape could be an appropriate measure.

$$\alpha = \frac{A_{partial\_surface}}{A_{complete\_surface}} \tag{2}$$

In Fig. 7 the distribution of both values in our dataset is shown. While for most of the samples θ is likely to be close to 1, only very few have a α value above 0.5. Considering Fig. 5 it becomes clear that capturing the surface area completely needs many more views compared to just capturing the outer shell.

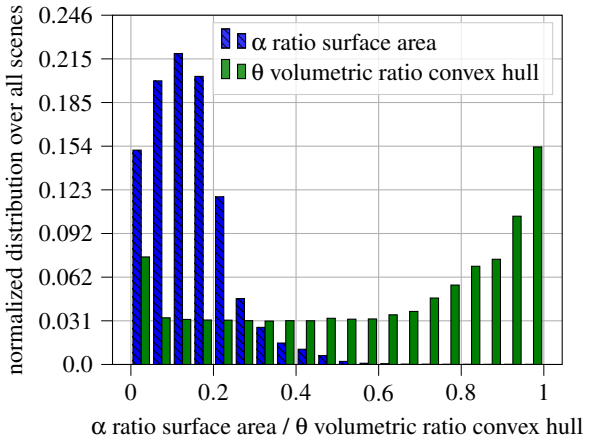

Figure 7: *Distribution of ratio between surface area (α) and volume (θ) of the partial scan with the original part*

3.6. Real-World Validation Dataset

For the purpose of validation of our *synScan* dataset, we manually gathered 81 scans from 34 pieces of furniture, using an iPad Pro 2020 LiDAR sensor. A truncated signed distance function with a cell size of $0.005m$ was used for fusing the captured depth images in order to sample pointclouds from the object surfaces.

4. Experiments and Discussion

4.1. Baseline Results

Following our intention to establish a dataset well suited for ILR, we evaluate state-of-the-art methods for 3D pointcloud place recognition. This formulates a problem very similar to our intended task of instance shape recognition.

Data preprocessing All of the methods originally consume normalized pointclouds downsampled to 4,096 points. Therefore, we uniformly sample pointclouds from the database and

the reconstructed scan meshes after centering and normalizing them accordingly. To make sure all scan data contains enough information to be recognizable, we choose to filter our dataset for $\theta \geq 0.5$, generating a subset of 23,180 scans from 6,499 objects. These samples are then split into 80% for training 10% for testing and 10% for validation.

Method implementation details Building the foundation for a lot of methods recently developed, we chose to evaluate the performance of *PointNetVlad*. Following the authors, we create the training tuples from an anchor pointcloud, two positives and 18 negative samples. To make sure a partial scan is always queried against its original shape, containing the entirety of features, we only use the database shapes as anchors and the partial scans as positive samples. We add the arbitrary rotated original shape to the positive sample list to avoid overfitting for queries where only a few positive samples exist. In the same manner, *SOE-Net* was evaluated, but the negative samples are reduced to 8 pointclouds with respect to the network size. The number of epochs is increased to 50, which is 2.5 times the original setting. All other model parameters are kept in their original settings. Considering the more efficient batch-hard negative mining used by *MinkLoc3Dv2*, we choose to use the partial scans as anchors and the database shapes as positive samples for this model. Unlike the original implementation, we do not use the positive-negative distance instead of the anchor-negative distance, if it violates the margin more. For all methods we apply random jitter with a value drawn from $\mathcal{N}(0;0.001)$.

Baseline Results Tab. 2 summarizes the evaluation results for the instance level recognition task on the *synScan* test split. Overall, *PointNetVLAD* and *MinkLoc3Dv2* are able to perform metric learning using the *synScan* dataset.

	AR@1%	AR@1
PNVLAD	63.2	34.2
PNVLAD + Attention	48.1	17.0
MinkLoc3Dv2	**91.0**	**67.9**
PNVLAD + *PointOE*	did not converge	did not converge
SOE-Net	did not converge	did not converge

Table 2: *Benchmark results (average recall at 1% and average recall at 1) for instance level recognition on incomplete scan data.*

MinkLoc3Dv2 achieves the highest results for average top 1% recall and average top N recall. Despite that, we found that [SOE-Net] did not converge. Presuming it to be not robust against the additional rotation around all 3 axes, we trained it on *The Oxford RobotCar Dataset* again with rotation around 3 axes. It did not reach convergence either. To further investigate that behaviour, we added the attention unit from *SOE-Net* to *PointNetVLAD* which decreased its performance but succeeded in training. Considering just adding the *PointOE* module from *SOE-Net* to *PointNetVLAD* did not converge as well, we conclude the *PointOE* module to be not equivariant at rotation around 3 axes.

4.2. Real-World Validation Dataset

For validation we create a query database by mixing the 34 CAD shapes of these objects with the *synScan* test split. After training both methods on our dataset, the average recall of all 81 real-world scans is calculated. Tab. 3 shows that they can successfully adapt to real-world scans after training on our dataset.

	AR@1%	AR@1
PNVLAD	60.5	34.6
MinkLoc3Dv2	**69.1**	**43.2**

Table 3: *Recall for* PointNetVlad *and* MinkLoc3Dv2 *on our real-world Validation dataset, trained on* synScan*.*

Fig. 8 illustrates top 5 query results for a given partial scan. When big parts of an object are captured, the network is able to find structurally similar shapes. However, it fails to retrieve the right shape if the scanned part is too small. An example can be seen in the last row, where only the table top was scanned, but not the legs. Nevertheless, the discrepancy in recognition performance, compared to our synthetic validation split data, indicates room for improvement.

Figure 8: *Examples of Minkloc3Dv2 top 5 retrieval results on our real-world validation dataset.*

4.3. Evaluation of effectiveness

To evaluate the effectiveness of our dataset compared with much less computation-intensive augmentation methods, we train

MinkLoc3Dv2 on the same subset of *ShapeNetSem* shapes but substitute the synthetic scans with samples with up to 95% randomly removed points and random jitter with a value drawn from $\mathcal{N}(0;0.001)$. Again, the method is tested on our real-world scan dataset (RW Scans).

	our RW Scans	
	AR@1%	AR@1
Minkloc3Dv2 w. Random Removal	9.8	1.2
Minkloc3Dv2 w. ***our*** *synScan*	**69.1**	**43.2**

Table 4: *Recall for training* MinkLoc3Dv2 *on* synScan *and* Random Removal.*(average recall at 1% and average recall at top 1)*

Compared to random removal training data, the results of Tab. 4 indicate that ILR methods trained with our *synScan* perform much better on partial scan data.

4.4. Zero-Shot Capability

To evaluate if methods trained with our *synScan* dataset have better zero-shot capabilities, we test *PointNetVlad* and *MinkLoc3Dv2* with *The Oxford RobotCar Dataset*. For better comparison, we also do the exact opposite and test *MinkLoc3Dv2* trained on *The Oxford RobotCar Dataset* with our real-world dataset (RW Scans).

	Oxford		ours RW Scans	
	AR@1%	AR@1	AR@1%	AR@1
PointNetVlad synScan weights	35.6	21.8	60.5	34.6
MinkLoc3Dv2 synScan weights	**46.2**	**27.7**	**69.1**	**43.2**
MinkLoc3Dv2 Oxford weights	96.3	98.9	4.9	2.5

Table 5: *Average recall for testing* MinkLoc3Dv2 *on* The Oxford RobotCar Dataset *trained on* synScan *and for comparison testing* MinkLoc3Dv2 *trained on* The Oxford RobotCar Dataset *on our real-world scan-data.*

As can be seen in Tab. 5, methods trained with our dataset achieve acceptable results on a completely different domain like *The Oxford RobotCar Dataset*, which illustrates the long-tail class distribution of *synScan*.

4.5. Ablation Study

Removal of inner structures

Scans cannot capture the inner structures of CAD meshes. Therefore, they are useless information for object recognition. To avoid wasting descriptor capacity on these inner structures, we provide all database files from *ShapeNetSem* with removed inner structures. When we retrain *MinkLoc3Dv2* on these data, Tab. 6 shows that we can increase the average recall at 1% by 6.18% whereas the top 1 average recall stays the same.

	AR@1%	AR@1
MinkLoc3Dv2 w/ inner structures	64.2	37.0
MinkLoc3Dv2 w/o inner structures	**69.1**	**43.2**

Table 6: *Recall for training* MinkLoc3Dv2 *on* synScan *with and without inner structures (average recall at 1% and average recall at top 1)*

5. Conclusions and Future Work

In this paper we show how to successfully create synthetic training data for 3D instance level recognition from partial object scan data. Our new *synScan* dataset offers significant advantages compared to existing datasets regarding the number of object categories included, the number of unique scene configurations as well as the provision of proper object scanning trajectories.

With our *synScan* dataset we want to promote the use of partial 3D scan data for object recognition in deep learning. Utilizing a rather simple data generation pipeline, we can show that deep learning methods trained on our synthetic dataset can successfully be used with real-world scan data. In this manner, *synScan* helps to overcome the lack of corresponding labeled training data.

The tested methods are able to produce good results for the task of ILR on incomplete scan data, apart from *SOE-Net* that is not robust against rotation in 3 axes.

We propose a simple pipeline to remove the inner structures of CAD meshes. Our results show that removing the inner structures of the database shapes can improve overall ILR performance.

By training *PointNetVlad* and *MinkLoc3Dv2* on our dataset and testing them on *The Oxford RobotCar Dataset* we show that the proposed dataset can enable few-shot or even zero-shot learning for ILR methods.

There are various perspectives for *synScan*'s application in future research: The dataset may be used for competitions in the area of algorithms for ILR based on partial scan data comparable to the current application of *ModelNet* [WSK*15] in the area of 3D object classification and retrieval. For this purpose, the dataset and associated scripts will be made available soon at https://github.com/mbookhahn/synScan.

References

[AGT*16] ARANDJELOVIĆ R., GRONAT P., TORII A., PAJDLA T., SIVIC J.: NetVLAD: CNN architecture for weakly supervised place

recognition. In *IEEE Conference on Computer Vision and Pattern Recognition* (2016). 3

[BCD*21] BARUCH G., CHEN Z., DEHGHAN A., DIMRY T., FEIGIN Y., FU P., GEBAUER T., JOFFE B., KURZ D., SCHWARTZ A., SHULMAN E.: ARKitscenes - a diverse real-world dataset for 3d indoor scene understanding using mobile RGB-d data. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)* (2021). URL: https://openreview.net/forum?id=tjZjv_qh_CE. 2

[CB21] COUMANS E., BAI Y.: Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2021. 3

[CFG*15] CHANG A. X., FUNKHOUSER T., GUIBAS L., HANRAHAN P., HUANG Q., LI Z., SAVARESE S., SAVVA M., SONG S., SU H., XIAO J., YI L., YU F.: *ShapeNet: An Information-Rich 3D Model Repository*. Tech. Rep. arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. 3

[Daw] DAWSON-HAGGERTY ET AL.: trimesh. URL: https://trimsh.org/. 4

[DCS*17] DAI A., CHANG A. X., SAVVA M., HALBER M., FUNKHOUSER T., NIESSNER M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE* (2017). 2

[HPN*16] HUA B.-S., PHAM Q.-H., NGUYEN D. T., TRAN M.-K., YU L.-F., YEUNG S.-K.: Scenenn: A scene meshes dataset with annotations. In *International Conference on 3D Vision (3DV)* (2016). 2

[HZG20] HUANG J., ZHOU Y., GUIBAS L.: Manifoldplus: A robust and scalable watertight manifold surface generation method for triangle soups, 2020. arXiv:2005.11621. 4

[Kom22] KOMOROWSKI J.: Improving point cloud based place recognition with ranking-based loss and large batch training, 2022. arXiv:2203.00972. 2, 3

[LZS*19] LIU Z., ZHOU S., SUO C., LIU Y., YIN P., WANG H., LIU Y.-H.: Lpd-net: 3d point cloud learning for large-scale place recognition and environment analysis, 2019. arXiv:1812.07050. 3

[MHLD17] MCCORMAC J., HANDA A., LEUTENEGGER S., DAVISON A. J.: Scenenet rgb-d: 5m photorealistic images of synthetic indoor trajectories with ground truth, 2017. arXiv:1612.05079. 2, 3

[MPLN17] MADDERN W., PASCOE G., LINEGAR C., NEWMAN P.: 1 Year, 1000km: The Oxford RobotCar Dataset. *The International Journal of Robotics Research (IJRR) 36*, 1 (2017), 3–15. URL: http://dx.doi.org/10.1177/0278364916679498, arXiv:http://ijr.sagepub.com/content/early/2016/11/28/0278364916679498.full.pdf+html, doi:10.1177/0278364916679498. 2

[NSF12] NATHAN SILBERMAN DEREK HOIEM P. K., FERGUS R.: Indoor segmentation and support inference from rgbd images. In *ECCV* (2012). 2

[PTL*18] PHAM Q.-H., TRAN M.-K., LI W., XIANG S., ZHOU H., NIE W., LIU A., SU Y., TRAN M.-T., BUI N.-M., DO T.-L., NINH T. V., LE T.-K., DAO A.-V., NGUYEN V.-T., DO M. N., DUONG A.-D., HUA B.-S., YU L.-F., NGUYEN D. T., YEUNG S.-K.: RGB-D Object-to-CAD Retrieval. In *Eurographics Workshop on 3D Object Retrieval* (2018), Telea A., Theoharis T., Veltkamp R., (Eds.), The Eurographics Association. doi:10.2312/3dor.20181052. 2

[QSMG17] QI C. R., SU H., MO K., GUIBAS L. J.: Pointnet: Deep learning on point sets for 3d classification and segmentation, 2017. arXiv:1612.00593. 3

[QYSG17] QI C. R., YI L., SU H., GUIBAS L. J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space, 2017. arXiv:1706.02413. 3

[SCH15] SAVVA M., CHANG A. X., HANRAHAN P.: Semantically-Enriched 3D Models for Common-sense Knowledge. *CVPR 2015 Workshop on Functionality, Physics, Intentionality and Causality* (2015). 2, 3

[UL18] UY M. A., LEE G. H.: Pointnetvlad: Deep point cloud based retrieval for large-scale place recognition, 2018. arXiv:1804.03492. 2

[VGO*20] VIRTANEN P., GOMMERS R., OLIPHANT T. E., HABERLAND M., REDDY T., COURNAPEAU D., BUROVSKI E., PETERSON P., WECKESSER W., BRIGHT J., VAN DER WALT S. J., BRETT M., WILSON J., MILLMAN K. J., MAYOROV N., NELSON A. R. J., JONES E., KERN R., LARSON E., CAREY C. J., POLAT İ., FENG Y., MOORE E. W., VANDERPLAS J., LAXALDE D., PERKTOLD J., CIMRMAN R., HENRIKSEN I., QUINTERO E. A., HARRIS C. R., ARCHIBALD A. M., RIBEIRO A. H., PEDREGOSA F., VAN MULBREGT P., SCIPY 1.0 CONTRIBUTORS: SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods 17* (2020), 261–272. doi:10.1038/s41592-019-0686-2. 4

[WSK*15] WU Z., SONG S., KHOSLA A., YU F., ZHANG L., TANG X., XIAO J.: 3d shapenets: A deep representation for volumetric shapes, 2015. arXiv:1406.5670. 7

[XXL*21] XIA Y., XU Y., LI S., WANG R., DU J., CREMERS D., STILLA U.: Soe-net: A self-attention and orientation encoding network for point cloud based place recognition, 2021. arXiv:2011.12430. 2