Semantic Risk-aware Costmaps for Robots in Industrial Applications using Deep Learning on Abstracted Safety Classes from Synthetic Data

Thomas Weber[1], Michael Danner[1], Bo Zhang[2], Matthias Rätsch[1] and Andreas Zell[3]
[1]*Reutlingen Research Institute, Reutlingen University, Alteburgstrasse 150, 72762 Reutlingen, Germany*
[2]*School of Electronics Information, Xi'an Polytechnic University, Xi'an, China*
[3]*Cognitive Systems, Eberhard-Karls-University Tübingen, 72076 Tübingen, Germany*

Keywords: Data Sets for Robot Learning, Deep Learning, Safety in Human and Robot Interaction, Detection and Recognition.

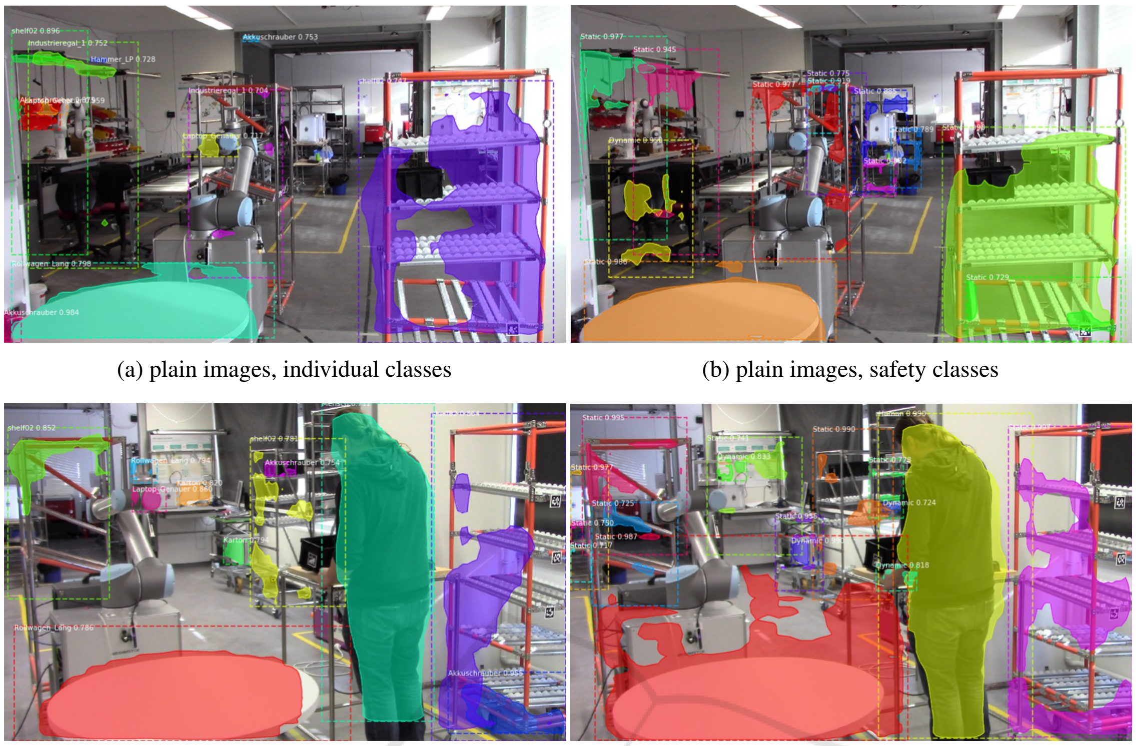
Abstract: For collision and obstacle avoidance as well as trajectory planning, robots usually generate and use a simple 2D costmap without any semantic information about the detected obstacles. Thus a robot's path planning will simply adhere to an arbitrarily large safety margin around obstacles. A more optimal approach is to adjust this safety margin according to the class of an obstacle. For class prediction, an image processing convolutional neural network can be trained. One of the problems in the development and training of any neural network is the creation of a training dataset. The first part of this work describes methods and free open source software, allowing a fast generation of annotated datasets. Our pipeline can be applied to various objects and environment settings and is extremely easy to use to anyone for synthesising training data from 3D source data. We create a fully synthetic industrial environment dataset with 10 k physically-based rendered images and annotations. Our dataset and sources are publicly available at https://github.com/LJMP/synthetic-industrial-dataset. Subsequently, we train a convolutional neural network with our dataset for costmap safety class prediction. We analyse different class combinations and show that learning the safety classes end-to-end directly with a small dataset, instead of using a class lookup table, improves the quantity and precision of the predictions.

1 MOTIVATION

Autonomous industrial robots for logistic transport often navigate in diverse and dynamic environments. For collision and obstacle avoidance as well as trajectory planning robots usually generate and use a simple two-dimensional costmap from a planar 2D laser scan. These costmaps do not contain any information about the obstacles: whether they are static or dynamic, moving in a certain direction or are living beings. Therefore, a robot's path planning will simply inflate around obstacles by an arbitrary large safety margin. This procedure generalises all obstacles in one category and is far from the optimal representation, exemplary in fig. 2a. Around a static obstacle (e.g. a cardboard box, europallet) the robot could navigate quickly and with little safety margin. The risk of a collision is limited; due to the immobility of the obstacle, it is practically impossible. Dynamic obstacles (e.g. a lift truck) are likely to change positions and are therefore a risky category. Humans crossing the path of the robot represent a high risk and should ideally be avoided by the robot at a large distance behind of their movement path direction. To accomplish this behaviour, however, the costmap would have to provide or contain this information (risk, direction, etc.). With additional information a 2D costmap with specific safety margins can be generated, see fig. 2b.

An image processing convolutional neural network (CNN) can be trained for this purpose. A challenge in the development of neural networks is not only the choice of architecture but also collecting the training dataset. Neural networks in the field of image processing for scene labelling and object segmentation require several thousand training images. These images must contain ground truth annotations and classes for all the objects of interest in the images. Conventionally, such datasets consist of real-world images, in which the objects are painstakingly classified manually. This procedure is very labour- and time-consuming. Computer graphics is a powerful tool and capable of synthesising photo-realistic images. The main

Weber, T., Danner, M., Zhang, B., Rätsch, M. and Zell, A.
Semantic Risk-aware Costmaps for Robots in Industrial Applications using Deep Learning on Abstracted Safety Classes from Synthetic Data.
DOI: 10.5220/0010904100003124
In *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2022) - Volume 4: VISAPP*, pages 984-990
ISBN: 978-989-758-555-5; ISSN: 2184-4321

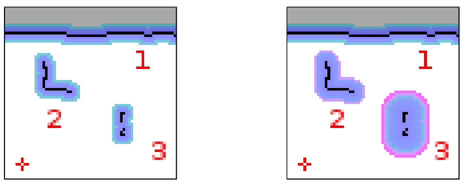


(a) plain images, individual classes (b) plain images, safety classes

(c) augmented images, individual classes (d) augmented images, safety classes

Figure 1: Inference results of different configurations on real-world images.

contributions of this work are (1) Introducing and publishing a tool-set for individual training and reducing manual procedures by methods allowing the fast generation of annotated datasets, which can be applied to various objects and environment settings and are extremely facile and rapid to use. (2) Proof of feasibility for training a CNN on a fully synthetic dataset to generate output information for enhancing the obstacle 2D costmap and thus enabling risk-aware navigation of autonomous robots in the real-world. (3) Providing a fully synthetic industrial environment dataset with 10 k physically-based rendered images and annotations, publicly available at https://github.com/LJMP/synthetic-industrial-dataset.



(a) standard (b) enhanced

Figure 2: Comparing 2D costmaps containing (1) static wall, (2) dynamic object contour, (3) human legs, (+) robot position.

2 RELATED WORK

Semantic segmentation, labelling and scene understanding are central problems in risk-aware robotic navigation and relevant work is comprehensive. For machine vision tasks like surface normal prediction, semantic segmentation, and object boundary detection in indoor scenes Li *et al* (Li et al., 2018), Song *et al* (Song et al., 2016) present datasets with millions of interior designs, furniture and object assets and Zhang *et al* (Zhang et al., 2016) improve the detection rate with a pretrained synthetic dataset.

Besides annotated indoor scenes (Song et al., 2015), datasets with an application to autonomous driving are very popular. In 2012 Geiger *et al* (Geiger et al., 2012) equipped a standard station wagon with two high-resolution colour and grey-scale video cameras to collect stereo and optical flow image pairs. Menze and Geiger (Menze and Geiger, 2015) propose a novel model and dataset for outdoor scenes decomposed into a small number of independently moving objects. Synthesising data approaches are also very common since (deep-)CNNs require lots of parameters from raw images as input, which has to be annotated manually. For the SYNTHIA Dataset (Ros et al., 2016) a virtual city with precise pixel-level semantic annotations has been generated. Wrennige *et al* (Wrenninge and Unger, 2018) introduce a synthetic dataset for street scene parsing and Richter *et al* (Richter et al., 2016) reconstruct traffic scenes from game engines.

Datasets of this kind cover domestic indoor and road traffic scenes, for which real-world annotated datasets are readily available. This allows comparing results of transfer-learning and fine-tuning networks with the respective real-world data.

To the best knowledge of the authors, neither synthetic nor real-world datasets for industrial environments are widely and freely available. While there are lots of logistic applications for robots in general and smart transportation systems, publicly accessible datasets are noticeably missing.

3 SYNTHESISING THE DATASET

Our work follows a similar approach to the related work, aiming at semantic segmentation and scene understanding, but in contrast, our goal is to improve real-world robot navigation and risk-assessment in factory logistic applications. The generation procedure of the aforementioned datasets involves fairly complex and unconventional pipeline setups. We attach importance to provide easy access to our setup and to facilitate adjustments and changes to the 3D source data.

This is why we opted for commonly used free and open source software (FOSS) in the field of 3D rendering: the *Blender (2.79b)* 3D creation suite (Foundation, 279b) on *Ubuntu (16.04)*. The fully graphical user interface of *Blender* enables easy "What You See Is What You Get" (WYSIWYG) editing of the 3D source data and the included *Python* API offers the capability to control all its functionality.

As studied in (Zhang et al., 2016), physically-based rendering (PBR) is more computationally expensive, yet provides better results in training a CNN. The PBR approch allows overcoming the domain gap with a limited set of parameters, which mimic the real-world closely, contrary to Tobin *et al* (Tobin et al., 2017) and is a computation and time trade-off. *Blender*'s physically-based path tracer for production rendering is *Cycles*. It provides physically-based results out-of-the-box and a flexible node tree feature for manipulating 3D information, shaders, etc. before outputting the data in a rendering. Thereby various additional annotations for the same 3D source data can be created with very little effort required from the user, see e.g fig. 3c.

The final dataset consists of 10 k physically based rendering RGB images with accompanying pixel-perfect object instance ground truth bit-masks. These bit-masks are convertible to binary mask and bounding box annotations in the commonly used COCO format (Lin et al., 2014).

Our dataset creation process is as follows: (1) Set up a 3D scene with readily available high quality 3D models from e.g. (TurboSquid, 2021; Sketchfab, 2021; Models, 2021; CGTrader, 2021; BlendFab, 2021). Shaders of these models[1] must be compliant to PBR or updated to a "Principled BSDF", (2) Use PBR lights indoor as well as high dynamic range 360° photos from e.g. (Zaal, 2021; Cywka, 2021; Reimer, 2021) for realistic HDR image-based scene lighting, (3) Define valid positions for the render camera (corresp. positions a robot will likely traverse) by weight-painting mesh vertices in the scene. This method avoids complicated programmatic checks for whether the camera would be placed randomly inside an object, (4) Program a script utilising the *Blender Python* API to step the camera through the 3D scene and randomly change camera angles in a defined interval for each rendering. Additional conditions for object count and coverage are also in this script; e.g. to avoid rendering an image of just a wall.

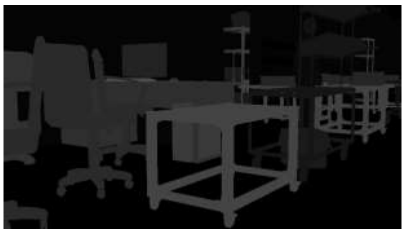
To accomplish the pixel-perfect object instance ground truth annotation our *Python* script once assigns a unique so called "PassIndex " to every object in the 3D scene. This object instance mapping is saved for later reference. For any rendered `png` RGB image the PassIndex values — representing the object instances in this image — are all rendered pixel-wise into a separate *OpenEXR* (`exr`) file. A sample RGB image is depicted in fig. 3a and fig. 3b shows the corresponding bit-masks in `exr`. The `exr` format saves the PassIndex value as is (no value mapping) and thus allows up to 2^{32} different objects within an image.

Rendering the bit-masks combined into an `exr` file reduced computation time. Further, changing the mapping in post-process conversion of bit-masks to a COCO annotation `json` file allows grouping of object instances into classes as desired. For the conversion to COCO annotation, we use *pycococreator* (waspinator, 2018). The `exr` file is read into a *Numpy* array and each PassIndex value is singled out into a binary mask. This mask is saved into the `json` as an instance annotation with a class according to its mapping.

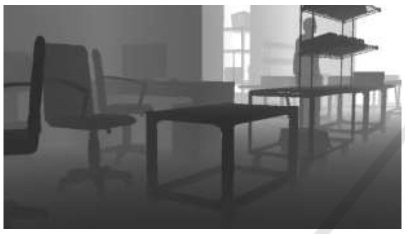
We render on a single machine with a *Nvidia Titan X (11GB)* and an *Intel i7-6800*. Final renders are 1280×720 pixels at 600 samples with *denoising* as a reasonable trade-off between RGB image quality and

[1]The authors highly advise to use "object instances with linked mesh data". Simply copying models will duplicate mesh data and increase memory consumption and render time immensely!

(a) RGB image



(b) Combined bit-masks



(c) Depth render pass

Figure 3: A rendering with annotations.

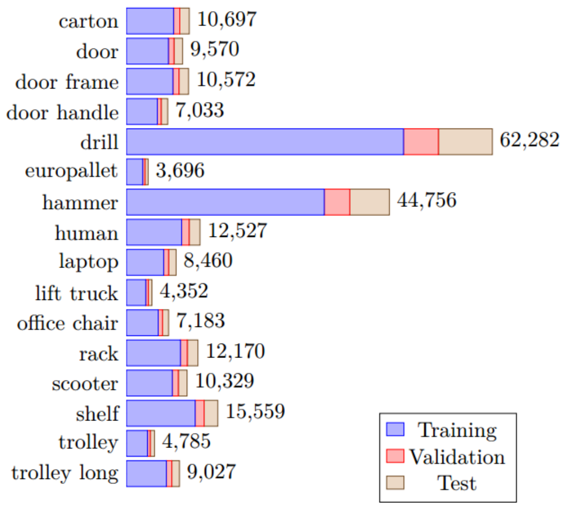


Figure 4: Total number of raw class instances and split distribution in the dataest.

computation time. Please see our public source data for further details on *Blender* setup and scripts.[2]

Our final dataset consists of 10 k images, split as follows: (a) 7500 training, (b) 1000 validation and (c) 1500 test images. The raw classes of objects in the dataset are deduced from the used 3D models (bold number denotes class ID): **0** background (ceiling, floor, wall), **1** door frame, **2** door, **3** door handle, **4** trolley, **5** laptop, **6** rack, **7** drill, **8** europallet, **9** scooter, **10** lift truck, **11** human, **12** trolley long, **13** office chair, **14** carton, **15** hammer and **16** shelf.

The total number of class instances and split distribution over all 10 k images is depicted in fig. 4. Examples of plain, unaltered RGB image renderings are shown in fig. 5.

4 EXPERIMENT SETUP

Our hypothesis is training a CNN directly end-to-end for safety classes – instead of individual classes and using a lookup table to map class to safety category – improves the overall performance in detection quantity and precision. Especially small datasets benefit by this approach to reduce class variance.

As humans are the most critical factor for risk-aware navigation of autonomous robots, "human" is always a separate class. Grouping similar type objects may improve the precision, as the dataset contains very similar objects (e.g. rack and shelf), which could be easily confused and therefore could lower the score.

Consequently, we propose three different class groupings on our dataset for experiments: (1) All dataset objects as their individual class, (2) similar objects grouped by type class, (3) objects directly grouped into safety classes. These groupings with the object class IDs are stated in table 1.

We chose the *Mask R-CNN* architecture (He et al., 2017) for our initial experiments. It is capable of object instance segmentation with the prediction of both masks and bounding boxes in parallel and has proven

Table 1: Different class grouping and IDs.

		grouped by													
		type										**safety**			
dataset class	**individual**	(background)	door	trolleys	tools	shelves	transports	scooter	lift truck	human	office chair	(background)	static	dynamic	human
(background)	0	0										0			
door frame	1	0										0			
door	2		1											2	
door handle	3	0										0			
trolley	4			2									1		
laptop	5				3							0			
rack	6					4							1		
drill	7				3							0			
europallet	8						5							2	
scooter	9							6						2	
lift truck	10								7					2	
human	11									8					3
trolley long	12			2									1		
office chair	13										9			2	
carton	14						5							2	
hammer	15				3							0			
shelf	16					4							1		

[2]The authors are aware of the new *Blender* release 2.80 and it's new physically-based real-time renderer *Eevee*. At the time of dataset creation the *Eevee* render engine produced unacceptable lighting artefacts. Also, the PassIndex method breaks and would need significant changes to work in *Eevee*.

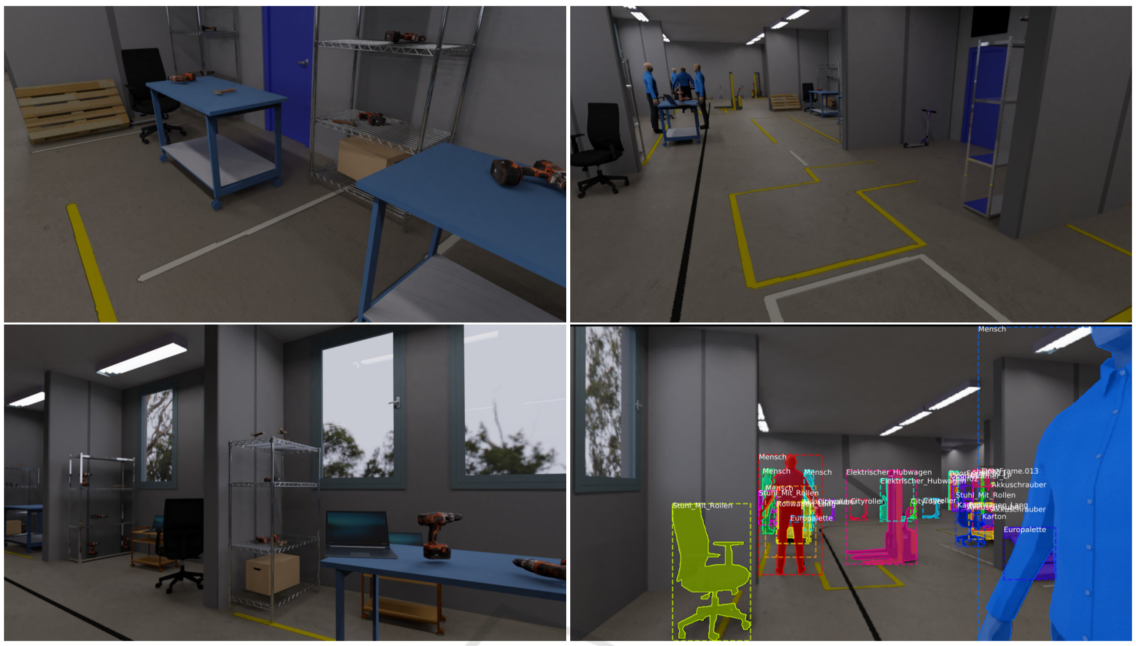


Figure 5: Examples of plain, unaltered renderings.

to generalise well in a broad range of other tasks. Readers are encouraged to test other architectures with our dataset. Masks and bounding box output information are deemed useful in post-process enhancement of a robot's costmap.

We start each training initialised with pre-trained COCO weights. The following are our hyper parameters and procedure: (a) backbone "resnet50", (b) learning rate 10^{-3}, (c) image dimensions 1280×1280 pixels, (d) batch size 2, (e) training the first ten epochs layers "heads", (f) and concluding with five epochs fine-tuning layers "all".

Overfitting on the dataset is to be expected, due to its small size, but this effect may not show in only 15 training epochs. However, we want to ensure predictions on real-world images are feasible. Therefore we repeat the three experiments with augmented images, as seen in fig. 6, to obtain a more robust preliminary score. Multiple of the following augmentations are applied to the RGB images in memory at random: (1) horizontal flip, (2) motion blur, (3) dropout and noises, (4) hue and saturation shift, (5) perspective transform. By augmenting "live" in training, the network will practically never be exposed to the same dataset image twice.

5 RESULTS

The results of our six experiments are displayed in table 2 with the configurations stated in section 4. The score per class is the mean of all scores on the testset calculated at an IoU of at least 50 % on the bounding boxes.

Comparing the experiments on plain versus augmented images, on average a 11 %-points drop in score shows. The decrease is owed to the same amount of fifteen training epochs on yet a much more diverse image input. The testset images are never augmented. Real-world image input will diverge even more from the testset, which is of our interest.

We observe an overall increase from the grouping "individual" over "type" to "safety". The same increase also arises in "safety class by lookup" scores. Scores of these looked up "dynamic", "human" and "static" classes are computed from the respective classes as defined in table 1. The grouping has virtually no effect on the "human" class. The results confirm our previously established hypothesis, training end-to-end for safety classes instead of a lookup table, on sparse data improves performance. A CNN seems able to abstract more similarities in the different objects concerning their "safety" aspect. This is also apparent in real-world images inference results, shown in fig. 1. Keep in mind, the trained network has never seen comparable real-world data tantamount to our synthetic dataset in training. Apparently, it is problematic for *Mask R-CNN* to learn concave / cutout masks, nonetheless the bounding boxes fit the object shape closely. In fig. 1b the network trained on the "safety" grouping shows more small objects detections in the background. The same can be noticed in fig. 1d. This represents an in-

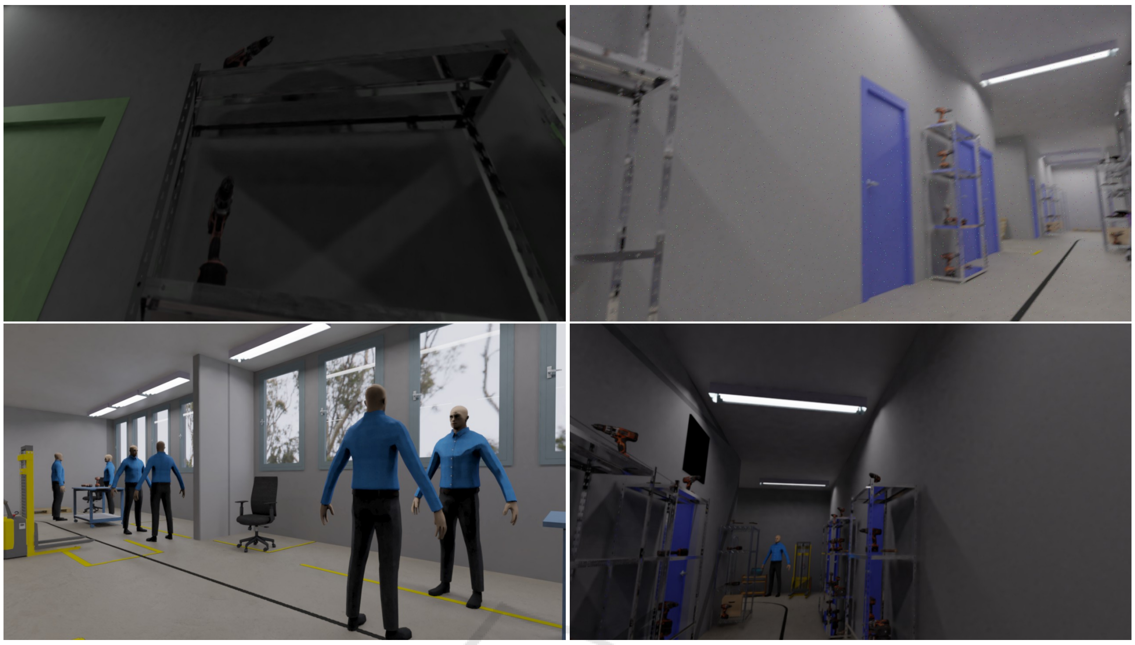


Figure 6: Examples of augmented images.

Table 2: Class mAP$_{50}$ [%] of different configurations on testset.

individual	**plain**	**augm.**	type	**plain**	**augm.**	safety	**plain**	**augm.**
carton	81.76	75.91	door	82.80	75.28	dynamic	80.17	69.46
door	73.63	52.36	human	81.68	71.13	human	83.85	77.64
door frame	51.43	21.50	lift truck	83.17	77.07	static	72.13	58.70
door handle	53.82	48.85	office chair	83.67	72.98			
drill	68.72	62.60	scooter	87.92	76.70			
europallet	79.52	73.71	shelves	59.14	48.16			
hammer	43.70	34.69	trolleys	76.20	62.57			
human	82.84	78.83	tools	61.12	54.75			
laptop	66.52	57.13	transports	37.33	25.72			
lift truck	87.78	76.81						
office chair	80.65	69.67						
rack	49.89	36.67						
scooter	40.16	25.64						
shelf	61.76	50.16						
trolley	70.95	58.63						
trolley long	81.63	69.67						
average	**66.05**	**54.90**	average	**70.85**	**60.72**	average	**77.78**	**66.88**
safety class by lookup								
dynamic	73.92	62.35	dynamic	74.98	65.55			
human	82.84	78.83	human	81.68	71.13			
static	66.06	53.78	static	67.67	55.37			

formation advantage in enhancing a robot's obstacle costmap. A greater amount of the surroundings are assets for risk in navigation.

6 CONCLUSION AND FURTHER WORK

We introduce a fully synthetic industrial environment dataset with 10 k physically-based rendered images and annotations. A pipeline is presented, which can be applied to various objects and environment settings and is extremely facile to use to anyone for synthesising training data from 3D source data. Annotated data, 3D source data, scripts and tutorials are published at https://github.com/LJMP/synthetic-industrial-dataset.

We show in multiple experiments, directly learning safety classes end-to-end on our dataset, instead of using a lookup table, substantially increases the prediction scores and quality on real-world data. This confirms our hypothesis that small datasets benefit from training abstracted similarities in different objects.

The results from these experiments prove the benefit of using our dataset for logistic-relevant industrial tasks. Expanding experiments with our dataset on different architectures can consolidate the effect of grouping on network performance. Additionally, our pipeline allows synthesising new datasets for novel scenes and environments, like consumer stores, warehouses and groceries stores.

Further work on this research focuses on creating a ROS-compatible (Garage and Laboratory, 2021) drop-in replacement for the commonly used 2D costmap generator (Marder-Eppstein et al., 2021). This will allow a robot's trajectory planning to take a more optimal approach. The safety-class-adjusted margin to an obstacle in the semantic costmap can now be considered, instead of a blunt arbitrary margin around obstacles, thus enabling a risk-aware navigation, as previously described in fig. 2.

ACKNOWLEDGEMENTS

This work is partially supported by a grant of the BMWi ZIM program, no. ZF4029424HB9

REFERENCES

BlendFab (2021). 5000+ Different free assets available for download.

CGTrader (2021). Search thousands of 3D models on sale.

Cywka, H. A. (2021). HDRi maps for CG artists.

Foundation, B. (v2.79b). Blender free and open source complete 3D creation pipeline.

Garage, W. and Laboratory, S. A. I. (2021). The Robot Operating System (ROS).

Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? The KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, pages 3354–3361. IEEE Computer Society.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. B. (2017). Mask R-CNN. *CoRR*, abs/1703.06870.

Li, W., Saeedi, S., McCormac, J., Clark, R., Tzoumanikas, D., Ye, Q., Huang, Y., Tang, R., and Leutenegger, S. (2018). InteriorNet: Mega-scale Multi-sensor Photo-realistic Indoor Scenes Dataset. *CoRR*, abs/1809.00716.

Lin, T., Maire, M., Belongie, S. J., Bourdev, L. D., Girshick, R. B., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft COCO: Common Objects in Context. *CoRR*, abs/1405.0312.

Marder-Eppstein, E., Lu!!, D. V., and Hershberger, D. (2021). costmap_2d.

Menze, M. and Geiger, A. (2015). Object scene flow for autonomous vehicles. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 3061–3070. IEEE Computer Society.

Models, B. D. (2021). The Original Blender 3D Model Repository.

Reimer, A. (2021). royalty free assets for your cgi productions.

Richter, S. R., Vineet, V., Roth, S., and Koltun, V. (2016). Playing for Data: Ground Truth from Computer Games. In Leibe, B., Matas, J., Sebe, N., and Welling, M., editors, *European Conference on Computer Vision (ECCV)*, volume 9906 of *LNCS*, pages 102–118. Springer International Publishing.

Ros, G., Sellart, L., Materzynska, J., Vázquez, D., and López, A. M. (2016). The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 3234–3243. IEEE Computer Society.

Sketchfab (2021). Publish & find 3D models online.

Song, S., Lichtenberg, S. P., and Xiao, J. (2015). SUN RGB-D: A RGB-D scene understanding benchmark suite. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 567–576.

Song, S., Yu, F., Zeng, A., Chang, A. X., Savva, M., and Funkhouser, T. A. (2016). Semantic Scene Completion from a Single Depth Image. *CoRR*, abs/1611.08974.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. (2017). Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30.

TurboSquid, I. (2021). Free 3D Models and Commercial Use 3D Models.

waspinator (2018). pycococreator is a set of tools to help create coco datasets.

Wrenninge, M. and Unger, J. (2018). Synscapes: A Photorealistic Synthetic Dataset for Street Scene Parsing. *CoRR*, abs/1810.08705.

Zaal, G. (2021). high quality HDRIs for free, no catch.

Zhang, Y., Song, S., Yumer, E., Savva, M., Lee, J., Jin, H., and Funkhouser, T. A. (2016). Physically-Based Rendering for Indoor Scene Understanding Using Convolutional Neural Networks. *CoRR*, abs/1612.07429.