25^{th} Computer Vision Winter Workshop
Domen Tabernik, Alan Lukežič, Klemen Grm (eds.)
Rogaška Slatina, Slovenia, February 3–5, 2020

Segmentation and Recovery of Superquadric Models using Convolutional Neural Networks

Jaka Šircelj[1,2], Tim Oblak[2], Klemen Grm[1], Uroš Petković[1],
Aleš Jaklič[2], Peter Peer[2], Vitomir Štruc[1] and Franc Solina[2]
[1] Faculty of Electrical Engineering, UL, Tržaška 25, Ljubljana, Slovenia
[2] Faculty of Computer and Information Science, UL, Večna pot 113, Ljubljana, Slovenia
jaka.sircelj@fe.uni-lj.si

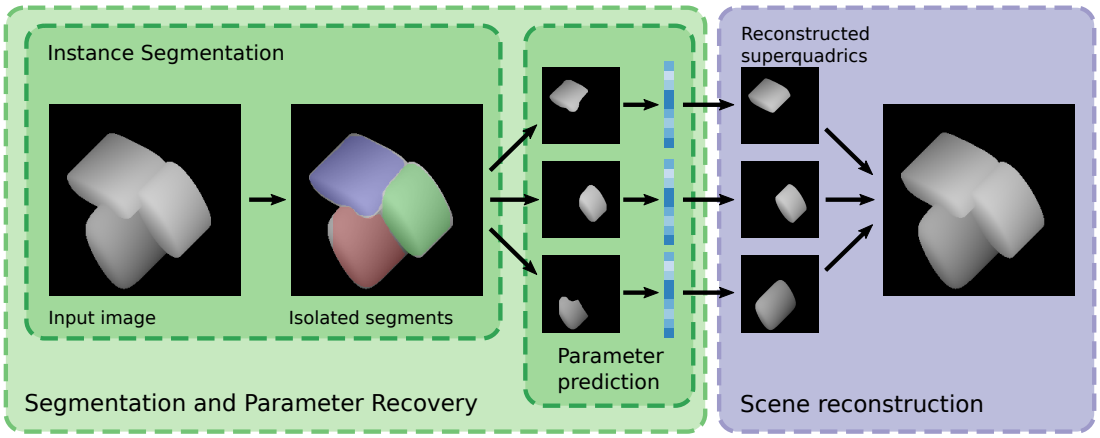


Figure 1: We study the problem of segmenting and recovering superquadric models from depth scenes. Our approach uses instance segmentation with Mask-RCNNs followed by superquadric-parameter estimation from incomplete data with a standard CNN (left part of the figure). Using the recovered superquadric models we are able to efficiently reconstruct the original depth scene (right part of the figure).

Abstract. *In this paper we address the problem of representing 3D visual data with parameterized volumetric shape primitives. Specifically, we present a (two-stage) approach built around convolutional neural networks (CNNs) capable of segmenting complex depth scenes into the simpler geometric structures that can be represented with superquadric models. In the first stage, our approach uses a Mask-RCNN model to identify superquadric-like structures in depth scenes and then fits superquadric models to the segmented structures using a specially designed CNN regressor. Using our approach we are able to describe complex structures with a small number of interpretable parameters. We evaluated the proposed approach on synthetic as well as real-world depth data and show that our solution does not only result in competitive performance in comparison to the state-of-the-art, but is able to decompose scenes into a number of superquadric models at a fraction of the time required by competing approaches. We make all data and models used in the paper available from* https://lmi.fe.uni-lj.si/en/research/resources/sq-seg.

1. Introduction

Representing three-dimensional visual data in terms of parameterized shape primitives represents a longstanding goal in computer vision. The interest in this problem is fueled by the vast number of applications that rely on concise descriptions of the physical 3D space in various sectors ranging from autonomous driving and robotics to space exploration, medical imaging and beyond [13, 21, 14].

Past research in this area has looked at different models that could act as volumetric shape primitives, such as generalized cylinders [28] or cuboids [27, 17, 11], but superquadrics established themselves as one of the most suitable choices for this task [1, 26, 10,

25, 18, 20] due to their ability to represent a wide variety of 3D shapes, such as ellipsoids, cylinders, parallelopipeds and various shapes in between. Formally, superquadrics are defined by an implicit 3D closed surface equation, i.e.:

$$\left(\left(\frac{x \quad x_0}{a_1}\right)^{\frac{2}{\epsilon_2}} + \left(\frac{y \quad y_0}{a_2}\right)^{\frac{2}{\epsilon_2}}\right)^{\frac{\epsilon_2}{\epsilon_1}} + \left(\frac{z \quad z_0}{a_3}\right)^{\frac{2}{\epsilon_1}} = 1 \quad (1)$$

where a_1, a_2, a_3 define the bounding box size of the superquadric, ε_1 and ε_2 define it's shape and $(x_0, y_0, z_0)^\intercal$ represent the center of the superquadric in a reference coordinate system [10]. Existing techniques for recovering superquadric models typically involve costly iterative parameter-estimation procedures that further increase in complexity if more than a single superquadric needs to be fitted to a scene [12, 10]. With complex scene geometries, superquadric recovery must necessarily be combined with segmentation techniques capable of partitioning the scene into simpler superquadric-like structures. This, however, puts a considerable computational burden on the fitting procedure as state-of-the-art techniques for recovery-and-segmentation of multiple superquadric models are typically extremely resource demanding.

With recent advances in computer vision and more importantly deep learning, it is possible to design solutions for simultaneous segmentation and recovery of superquadrics that are much more efficient than existing solutions. In this paper, we, therefore, revisit the problem of representing complex depth scenes with multiple superquadrics and develop an efficient solution for this task around convolutional neural networks (CNNs). Specifically, we assume that small superquadric-like structures in range images can be modeled as instances of a specific class of objects, and, therefore, train a Mask-RCNN [7] model to segment the scene, as illustrated in Fig. 1. The results of this instance segmentation are then used as input to a second CNN that recovers superquadric parameters for each of the identified superquadric-like objects. Because the identified superquadric-like objects may be partially occluded, we account for this fact during training and learn the parameters of the second CNN in a robust manner. We evaluate the performance of our approach on simulated, but also real-world range images. We achieve segmentation and recovery results comparable to the state-of-the-art, but achieve a considerable speed-up, which makes the developed solution suitable for a much wider range of applications. We note that in this paper we approach a constrained superquadric recovery problem, where we assume that the depth scene can be approximated by a number of unrotated superquadric models.

Our main contributions in this paper are:

- We present a novel solution for segmentation and recovery of multiple (unrotated) superquadric models from range images built around CNNs and evaluate it in experiments with simulated and real-world depth data.
- We show that existing Mask-RCNNs may be used for identifying superquadric-like structures in range images in an efficient manner.
- We demonstrate that superquadrics can be recovered from partial depth data using a simple CNN-based regressor and the parameter estimation errors are comparable to the error produced by state-of-the-art techniques used for this task.

2. Related work

Existing techniques to scene segmentation with superquadrics can in general be divided in one of two groups: *i)* techniques that approach the problem by segmenting the scene and recovering superquadrics at the same time (*segment-and-fit*), and *ii)* techniques that first segment the scene and then fit superquadric models to the segmented parts (*segment-then-fit*). In this section we briefly review both groups of techniques with the goal of providing the necessary context for our work. For a more comprehensive coverage of the subject, the reader is referred to [10].

Segment-and-fit. Techniques from this group typically combine the segmentation and superquadric recovery stages and often rely on superquadric models to guide the segmentation [5, 12, 10, 9]. Due to the fact that segmentation is performed with the final scene representations (i.e., the superquadric) methods from this group are considered highly robust. However, on the down side, they often also induce a considerable computational burden on the segmentation procedure. Recently, a CNN-solution [20] that falls into this group was proposed, but unlike the approach presented in this paper, was limited to segmentation of predefined classes of objects.

Segment-then-fit. Techniques from this group follow a two-stage procedure, where the data is first segmented up front and independently of superquadric recovery [10]. Thus, the entire procedure is broken down into two independent parts. Examples of techniques from this group include [6, 22, 2,

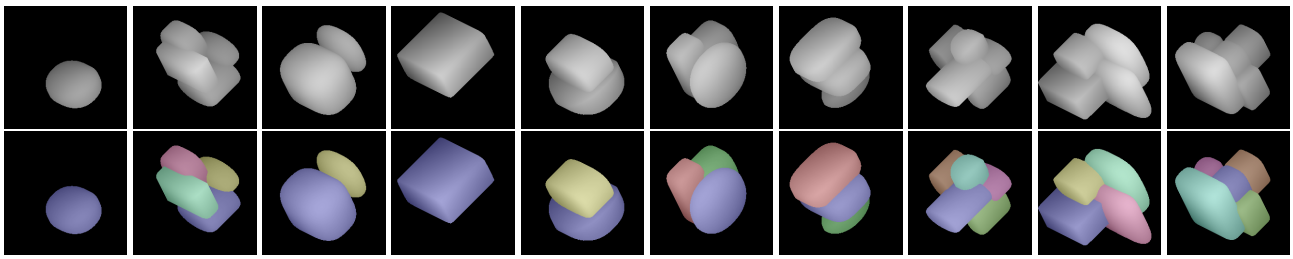


Figure 2: Example images from the generated dataset. The top row shows examples of the rendered images with different numbers of superquadric in the scene. The lower row shows examples of the corresponding segmentation masks. The figure is best viewed in color.

23]. The solution described in this work also follow the segment-then-fit paradigm, but as we show in the experimental section result in competitive performance compared to a state-of-the-art approach from the segment-and-fit group that is in general considered to be more robust.

3. Dataset

In order to train our instance segmentation and parameter estimation models, we require a large dataset of depth scenes with appropriate ground truth labels. Since no such datasets are publicly available, we generate our own and make it publicly available for the research community. In this section we present the dataset creation procedure and discuss the characteristics of the generated data.

3.1. Prerequisites

In this work we follow the methodology of Oblak *et al.* [18] and focus on unrotated superquadric models. Thus, we only try to recover the 8 open parameters from Eq. (1) for each superquadric model and omit rotations, which introduce ambiguities in the superquadric-recovery process [18]. The main goal of this work is to extend the superquadric recovery method from [18] to depth scenes with complex geometry that need to be represented with multiple superquadrics. Consequently, we fix the rotation of the objects in our dataset and render them in an axonometric projection that ensures that three sides of the objects are always visible in the rendered images.

3.2. Dataset creation

We synthesize our dataset by rendering range images with multiple superquadrics in the scene. To construct the range images we create a custom rendering tool that accepts multiple superquadric parameter sequences. The renderer then constructs the range image of a scene by finding the surface points of the superquadrics and choosing the closest point to the viewport, if there are overlapping superquadrics in the line of sight. The scene is constrained inside a 256 256 256 grid, where the first two dimensions represent the width and height of the resulting image, while the last dimension represents the depth. The scene is then mapped to the zero depth plane, resulting in a 256 256 range image, where its pixel indexes i, j correspond to the x, y coordinates in the 3D scene, while the pixel intensity relates to the z depth in the scene.

To generate a dataset with representative superquadric objects, we uniformly sample the superquadric parameters similarly to [18]. However, uniformly sampling the position and size of superquadrics independently from their neighbors causes dramatic overlaps and intersections in the scene, which hides a large number of objects. We solve this by constraining the allowed intersection-over-union volume between pairs of superquadrics in each scene, where the volume is approximated using the superquadrics bounding-box. Following this requirement we first sample the number of superquadrics in the scene from the discrete uniform distribution $U(1, 5)$. Then, for each scene, we iteratively sample superquadric parameters. If the new superquadric intersects with the superquadrics already in the scene, we discard it and sample again. This procedure continues until there are as many superquadrics on the scene as determined in the initial sampling step. Each superquadric has its size parameters sampled from a continuous uniform distribution $U(25, 76)$ and the shape parameters from $U(0.01, 1)$ limiting the appearance of the rendered models to convex shapes, which are also more representative of the real world. We sample the x_0 and y_0 center coordinates from $U(88, 169)$ while the z_0 coordinate

Table 1: Dataset summary.

#Superquadrics	1	2	3	4	5	Any
#Train Images	15882	16108	15930	15983	16097	80000
#Validation Images	3989	3944	4020	3948	4099	20000
#Test Images	3949	4023	3996	4059	3973	20000

is sampled from a tighter region $U(100, 150)$. This is done to constrain the vertical overlap between the superquadrics in the scene.

Along with the range image we also render a ground truth segmentation mask image of the scene, by coloring the different visible parts of the superquadrics with a different shade of gray. This ground truth information is used for training and evaluating the segmentation model.

3.3. Dataset totals

The complete dataset contains 120000 rendered scenes and corresponding segmentation masks. We also store range images of individual superquadrics in each scene in the dataset along with their parameters. For the experiments we split the dataset into three disjoint parts: for training, validation and testing. We use the training set to learn the parameters of our models, the validation set to observe over-fitting issues during training and the test for the final performance evaluation. A few illustrative examples from the generated dataset together with the corresponding segmentation masks are shown in Fig. 2 and a high-level summary of the dataset and experimental setting is given in Table 1.

4. Superquadric recovery methodology

In this section we now present our approach to segmentation and recovery of multiple superquadrics using CNN models.

4.1. Segmentation

As our range images contain multiple objects of the same class (i.e., superquadric-like objects), we resort to instance segmentation to identify parts of the range images belonging to structures that can be represented with superquadrics. One of the most popular models for instance segmentation is Mask R-CNN [7], which operates in a two-stage fashion. In the first stage, it uses a region proposal network (RPN) that finds candidate regions in the image. In the second stage, the final predictions are made. Here, three model heads are used: one for detection (two-class classification: object present or not), one for regression of the bounding boxes, and one for prediction of the binary segmentation mask.

Table 2: Architecture of the CNN regressor used for superquadric parameter estimation.

#	Output size	Layer operation	#kernels, size, stride
1	128 128	Conv2D+BN+ReLU	32, 7 7, $s2$
2	128 128	Conv2D+BN+ReLU	32, 3 3, $s1$
3	128 128	Conv2D+BN+ReLU	32, 3 3, $s1$
4	64 64	Conv2D+BN+ReLU	32, 3 3, $s2$
5	64 64	Conv2D+BN+ReLU	64, 3 3, $s1$
6	64 64	Conv2D+BN+ReLU	64, 3 3, $s1$
7	32 32	Conv2D+BN+ReLU	64, 3 3, $s2$
8	32 32	Conv2D+BN+ReLU	128, 3 3, $s1$
9	32 32	Conv2D+BN+ReLU	128, 3 3, $s1$
10	16 16	Conv2D+BN+ReLU	128, 3 3, $s2$
11	16 16	Conv2D+BN+ReLU	256, 3 3, $s1$
12	16 16	Conv2D+BN+ReLU	256, 3 3, $s1$
13	8 8	Conv2D+BN+ReLU	256, 3 3, $s2$
14	16384	Flatten	N/A
15	8	Dense	N/A

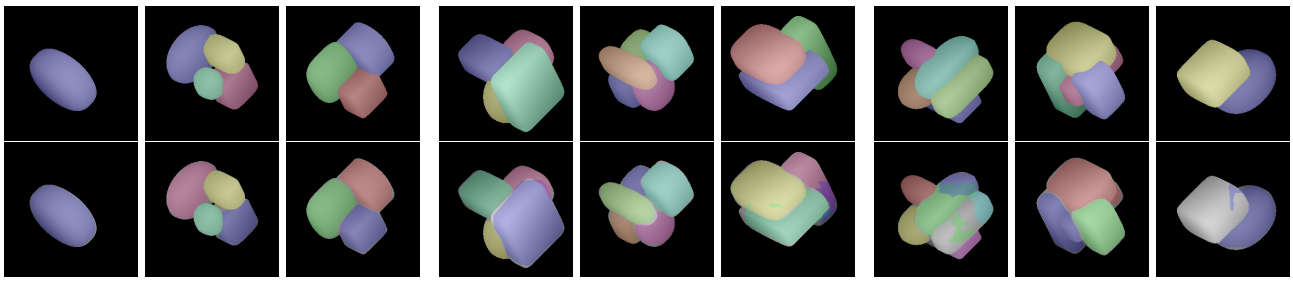
In our implementation, we use a ResNet-101 [8] backbone as the feature extractor along with a feature pyramid network (FPN) that makes it possible to exploit multiple scales of the feature maps. These features get fed trough a region proposal network which predicts object scores and their bounding boxes at each feature position. The predictions are then filtered by a non-maximum suppression algorithm, which removes overlapping bounding boxes.

The RPN bounding boxes and the FPN features get combined using the RoIAlign operator and fed into the three network heads to obtain the final class (object present or not), bounding box, and binary mask for each region proposal. Here the classification scores are used for the elimination of any background instances. For more information on Mask R-CNNs, the reader is referred to [4, 3, 24, 15, 7].

4.2. Parameter estimation

Once the scene is segmented and superquadric-like objects are identified in the input images, we feed the predictions into a CNN regressor for parameter estimation. We follow the work of [18] and use a regression model derived from the popular VGG architecture [19]. The model is designed as a 13 layer CNN with a fully-connected layer of size 8 on top. Each conv layer is followed by batch normalization and a ReLU activation, which reduces overfitting and allows the model to better generalize. The model is summarized in Table 2.

The input to the CNN regressor is a range im-



(a) 100% (b) 93.8% (c) 93.3% (d) 86.9% (e) 80% (f) 76.7% (g) 55.7% (h) 52.5% (i) 40%

Figure 3: Predicted segmentation masks from the Mask R-CNN model. The images are ordered in columns of three. Three good predictions (left), three average predictions (middle) and three bad predictions (right). In the first row we show range images with overlaid ground truth masks. The second row shows masks obtained with our segmentation model. Under the images we also report the mAP value for the segmentation. Most of the predictions are sufficient, even in the average subsection of the predictions. We observe that fine details are elusive to the model, such as disconnected masks (h) or narrow subparts of masks (e,f). Best viewed in color.

age containing a single superquadric-like instance and the output is a prediction of 8 parameters describing the size, shape and position, of the superquadric representing the input data, i.e., $\mathbf{y} = [a_1, a_2, a_3, \varepsilon_1, \varepsilon_2, x_0, y_0, z_0]$. Different from [18], the inputs to our model are not necessarily complete superquadrics, but automatically segmented range data, where parts of the object may be occluded due to overlap with other objects in the scene. Thus, we account for this in our training procedure and learn the parameters of our regressor by utilizing occluded data. As we show in the experimental section this allows us to quite efficiently estimate superquadric parameters even if part of the data is missing either due to occlusions or errors in the segmentation steps.

5. Experiments and results

5.1. Instance segmentation

The Mask R-CNN backbone is initialized with a ResNet-101 structure [8], pre-trained on the MS COCO dataset [16]. The training is split into two stages. In the first stage, we lock the training of the backbone and set the learning rate to 10^{-3}, with momentum of 0.9. In the second stage we unlock the backbone and fine-tune the network with a smaller learning rate of 10^{-4}. We present the standard mean average precision (mAP) scores of the instance segmentation in Table 3, as used in the COCO challenge. The model is trained on $80k$ training range-images of superquadric scenes, with a batch size of 2. We use an additional $20k$ images for validation and $20k$ images for testing. The model is trained on an NVIDIA GTX TITAN X GPU.

Table 3: Instance segmentation results. mAP_{50} and mAP_{75} denote scores computed at 50% and 75% IoU respectively, while mAP denotes the mean average precision averaged over IoU values from 50% up to 95%, taken at 5% steps.

mAP	mAP_{50}	mAP_{75}
85.57	97.33	95.95

In Table 3 we report the segmentation results using our Mask R-CNN model. We can see that average precision at Intersection-over-Union (IoU) thresholds 50% and 75% are higher than the averaged mAP over multiple IoU thresholds. This indicates that the model fail only at the highest intersections, segmenting the objects with good detail and precision.

In Figure 3 we present some examples of predicted masks for the training set. Most of the objects have been segmented with sufficient precision. On average, the model only misses smaller and highly occluded objects (Figures 3e and 3f). It also struggles with objects visually cut in half because of overlaps (Figure 3h). In these cases we either get multiple separate instance segments or the model fails to detect one of the parts completely. We suspect this might be caused by significant bounding box overlap between the foreground and background objects. The latter causing the former to get suppressed by the Mask R-CNN non maximum suppression algorithm.

5.2. Parameter prediction

We initialize the parameter prediction model with the weights from [18], as the same neural network architecture was used in that work. To train the pa-

Table 4: Parameter-prediction performance. The table shows MAE scores for each of the 8 superquadric parameters. The rows show results on different subsets of segmented range images test set, defined by the number of superquadrics the parent scene. The "All" row shows scores averaged over the entire set.

#sq	Dimensions [0-256]			Position [0-256]			Shape [0-1]	
	a_1	a_2	a_3	x_0	y_0	z_0	ϵ_1	ϵ_2
All	1.134	1.187	1.248	1.953	1.864	2.639	0.017	0.017
1	0.515	0.555	0.537	0.957	0.925	2.154	0.009	0.008
2	0.681	0.736	0.728	1.165	1.093	2.181	0.011	0.010
3	0.930	0.984	1.036	1.528	1.448	2.386	0.013	0.013
4	1.580	1.646	1.708	3.066	2.966	3.110	0.026	0.025
5	1.201	1.241	1.357	1.776	1.669	2.685	0.017	0.017

rameters of the model we use the ADAM minibatch stochastic gradient descent optimisation algorithm, which minimizes the MSE loss. We set the learning rate of the algorithm to 10^{-3} and keep the rate constant during training. As already indicated above, we use the segmentations produced by our Mask R-CNN model as the basis for the training to make the model robust to missing data. We only train on segmentations with an IoU higher than 50% compared to the ground truth masks. The model is trained for 63 epochs, with varying batch sizes constructed always from batches of 4 scene range images, giving us a maximum batch size of 20 segmented range images. We report performance for the CNN regressor in terms of the Mean Absolute Error (MAE) between the predicted and ground truth parameters. This measure was sufficient for our problem, since we predict superquadric parameters for superquadric visualizations, where the matching of parameters correlates with the 3D matching of the objects.

In Table 4 we present the MAE scores for each parameter on a test set of 20000 images. In addition to the MAE score for the entire test set, we also show separate MAE scores for scenes with different numbers of superquadrics. On average the model performs very well, predicting position and size in the order of one pixel accuracy compared to the $[0, 256]$ range of possible values. The shape parameters ε_0 and ε_1 also achieve about 0.017 mean absolute error which is also small compared to the $[0, 1]$ range of possible values. The model performs better in scenes with a smaller number of superquadrics since more superquadrics in the scene typically result in greater intersections and occlusions. Table 4 shows an almost monotonous increase in MAE as the number of superquadrics is increased, the only disparity is a larger error in scenes with 4 objects than in scenes with 5.

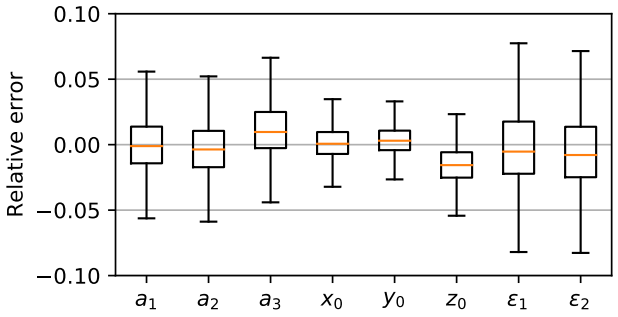


Figure 4: Box-and-whiskers plots of the relative error for each parameter.

In Figure 4 we show box-and-whiskers plots of the relative errors between ground truth and the predicted parameter values over the entire test set of segmented range images. We see that most of the error mass is close to the mean. The positional parameters are predicted with especially small variance in their errors. We also observe that the z axis size parameters are on average slightly overestimated. This seems to get compensated by an underestimation of the z axis position, thus aligning the top surface of the ground truth and the predicted superquadrics.

Scenes with larger numbers of superquadrics are harder to segment, occasionally giving our parameter prediction model highly corrupted segmentation masks, that can either blend range information from multiple objects into one segmented range image or return smaller subsets of the actual masks. On such corrupt segmented range images our prediction model naturally performs much worse than on cleaner segmentations, resulting in a somewhat heavy-tailed error distribution. We show this in Figure 5 where we plot the error distribution for all parameter predictions and subsets over the number of superquadrics in the scene. We also show how our segmentation model performs on each subset by showing the distribution of IoU values for its pre-

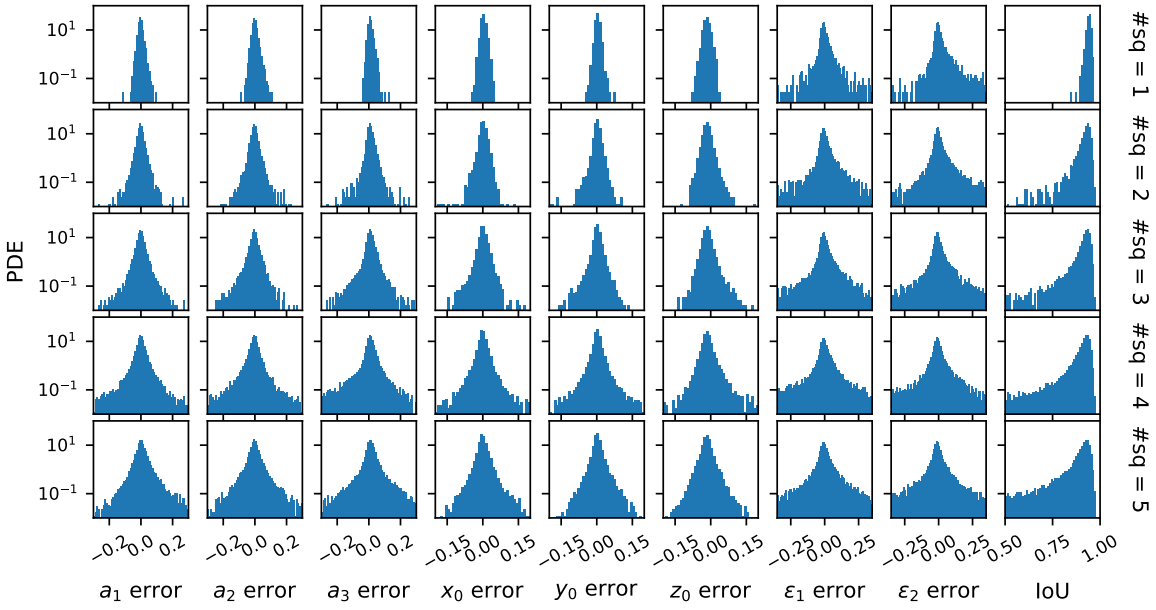


Figure 5: Our methods error distribution for each parameter. Each row shows results obtained from the 5 subsets scene images, each with a different number of superquadrics in its scenes. We also add the last column showing the IoU distribution of the predicted masks with Mask R-CNN.

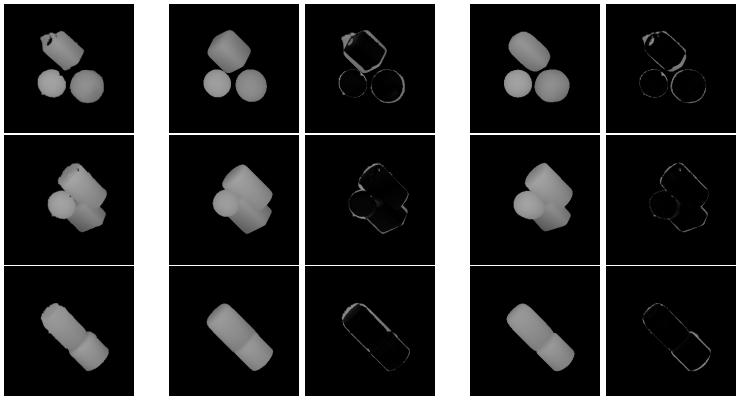


Figure 6: Qualitative comparison with the state-of-the-art: Input range images of (scanned) real-world objects (first column), Our reconstructions (second column), Absolute difference between the ground truth and our reconstruction (third column), Reconstructions by Leonardis et. al. [12, 10] (fourth column), Absolute difference between the ground truth and reconstruction by Leonardis et. al. [12, 10] (last column).

dicted segmentations. The distributions move away from a Gaussian shape quickly when more than one superquadric is present in the scene. The tails become larger when we increase the number of objects in the scene. As mentioned earlier, this can be explained by the inefficiency of the segmentation model, as the model also performs worse with greater numbers of objects in the scene - the IoU distribution becomes more and more skewed, with a heavier tail.

We also compare our approach to the state-of-the-art segmentation and superquadric recovery method from [12, 10] on range-images of real objects. For this experiment, we used range-image scans of real objects taken by Oblak et. al. for their work in [18]. We constructed range image scenes of multiple object by shifting the original images in pixel space and combining them using the max operator. The original range images, and their superquadric reconstructions using our approach and the state-of-the-art method from [12, 10] are shown in Figure 6. The iterative method from [12, 10] performs comparably to our solution, as we can see from the examples. Our method achieved 2.79 MAE calculated over all pixels differences from all pairs of ground truth and reconstructed images while [12, 10] scored 1.78. However, we note that the iterative algorithm of the original

method results in much higher processing times. Our method performs similarly in terms of reconstruction quality, but computes the segmentations and parameter predictions with a 100 speed up over the state-of-the-art approach. Specifically, the iterative method converges in about 10 s on one image while our method needs 0.11 s on a GPU. While our methods advantage against [12, 10] is that we can parallelize its computations, it still performs faster on a single threaded CPU with about 5 s per image.

6. Conclusion

We have presented a CNN-based solution for segmentation and recovery of multiple superquadrics from range images. We have shown that the designed solution is able to efficiently decompose complex depth scenes into smaller parts that can be modelled by superquadric models. Our approach was shown to produce scene reconstruction on par with a state-of-the-art method from the literature, while ensuring a significant speed up in processing times. As part of our future work, we will extend the solution to account for rotated superquadrics as well.

Acknowledgements

This research was supported in parts by the ARRS (Slovenian Research Agency) Project J2-9228 "A neural network solution to segmentation and recovery of superquadric models from 3D image data", ARRS Research Program P2-0250 (B) "Metrology and Biometric Systems" and the ARRS Research Program P2-0214 (A) "Computer Vision".

References

[1] R. Bajcsy and F. Solina. Three dimensional object representation revisited. In *ICCV*, pages 231–240, 1987.

[2] F. P. Ferrie, J. Lagarde, and P. Whaite. Darboux frames, snakes, and super-quadrics: geometry from the bottom up. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(8):771–784, Aug 1993.

[3] R. Girshick. Fast R-CNN. In *ICCV*, Dec 2015.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, June 2014.

[5] A. Gupta and R. Bajcsy. Volumetric segmentation of range images of 3d objects using superquadric models. *CVGIP: Image Understanding*, 58(3):302 – 326, 1993.

[6] A. Gupta, G. Funka-Lea, and K. Wohn. Segmentation, Modeling And Classification Of The Compact Objects In A Pile. In D. P. Casasent, editor, *Intelligent Robots and Computer Vision VIII: Algorithms and Techniques*, volume 1192, pages 98 – 109. SPIE, 1990.

[7] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *ICCV*, pages 2980–2988, Oct 2017.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[9] T. Horikoshi and S. Suzuki. 3D parts decomposition from sparse range data using information criterion. In *CVPR*, pages 168–173, June 1993.

[10] A. Jaklič, A. Leonardis, and F. Solina. *Segmentation and recovery of superquadrics*. Kluwer, 2000.

[11] H. Jiang and J. Xiao. A linear approach to matching cuboids in rgbd images. In *CVPR*, pages 2171–2178, 2013.

[12] A. Leonardis, A. Jaklič, and F. Solina. Superquadrics for segmenting and modeling range data. *IEEE TPAMI*, 19(11):1289–1295, 1997.

[13] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun. Towards fully autonomous driving: Systems and algorithms. In *2011 IEEE IV*, 2011.

[14] R. Li, X. Jia, J. H. Lewis, X. Gu, M. Folkerts, C. Men, and S. B. Jiang. Real-time volumetric image reconstruction and 3D tumor localization based on a single x-ray projection image for lung cancer radiotherapy. *Medical Physics*, 37(6Part1):2822–2826, 2010.

[15] T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 936–944, July 2017.

[16] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common Objects in Context. In *ECCV*, 2014.

[17] C. Niu, J. Li, and K. Xu. Im2Struct: Recovering 3D Shape Structure from a Single RGB Image. In *CVPR*, 2018.

[18] T. Oblak, K. Grm, A. Jaklič, P. Peer, V. Štruc, and F. Solina. Recovery of Superquadrics from Range Images using Deep Learning: A Preliminary Study. In *IWOBI*, pages 45–52. IEEE, 2019.

[19] O. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *BMVC*, volume 1, page 6, 2015.

[20] D. Paschalidou, A. O. Ulusoy, and A. Geiger. Superquadrics revisited: Learning 3D shape parsing beyond cuboids. In *CVPR*, pages 10344–10353, 2019.

[21] L. Pedersen. Science target assessment for Mars rover instrument deployment. In *IROS*, volume 1, Sep. 2002.

[22] A. P. Pentland. Automatic extraction of deformable part models. *International Journal of Computer Vision*, 4(2):107–126, 1990.

[23] N. Raja and A. Jain. Obtaining generic parts from range images using a multi-view representation. *CVGIP: Image Understanding*, 60(1):44 – 64, 1994.

[24] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[25] J. Slabanja, B. Meden, P. Peer, A. Jaklič, and F. Solina. Segmentation and reconstruction of 3D models from a point cloud with deep neural networks. In *ICTC*, 2018.

[26] F. Solina and R. Bajcsy. Recovery of parametric models from range images: The case for superquadrics with global deformations. *IEEE TPAMI*, 12(2):131–147, 1990.

[27] S. Tulsiani, H. Su, L. J. Guibas, A. A. Efros, and J. Malik. Learning shape abstractions by assembling volumetric primitives. In *CVPR*, pages 1466–1474, July 2017.

[28] Y. Zhou, K. Yin, H. Huang, H. Zhang, M. Gong, and D. Cohen-Or. Generalized cylinder decomposition. *ACM Trans. Graph.*, 34(6):171:1–171:14, Oct. 2015.