Recovery of superquadric parameters from range images using deep learning

Klemen Grm†, Tim Oblak‡, Aleš Jaklič‡, Vitomir Štruc†, Peter Peer‡, Franc Solina‡

University of Ljubljana; †Faculty of Electrical Engineering, ‡Faculty of Computer and Information Science

Abstract

With the recent advancements in deep neural computation, we devise a method to recover superquadric parameters from range images using a convolutional neural network. By training our simple, fully-convolutional architecture on synthetic data images, containing a single superquadric, we achieve encouraging results. In a fixed rotation scenario, the model could already be used in practice, but we still need to improve on prediction of arbitrary rotational parameters in the future.

Introduction

The reconstruction of 3D geometrical information from range images or point cloud data is an active challenge in the domain of computer vision. Real-world objects are usually complex, but can be closely approximated using simpler shape representations. A high-level geometric model is needed to describe such spatial entities without loosing too much information. Superquadrics are one of the possible choices for object representation in a 3D space. They are a broad family of parametric models, which can be controlled by a small number of parameters, while still covering a wide variety of 3D shapes.

Methods for fitting superquadrics to objects already exist, either by finding the parameters using least squares minimization [1], or by clustering the cloud data into smaller and smaller voxels [2], both being iterative solutions. In recent years we are witnessing an emerging trend of deep neural network approaches, which enables us to find patterns in a more unconstrained environment. The goal of our research is to find parameters of superquadrics by training an end-to-end deep convolutional network. Currently, we limit ourselves by assuming only a single shape is present in the scene.

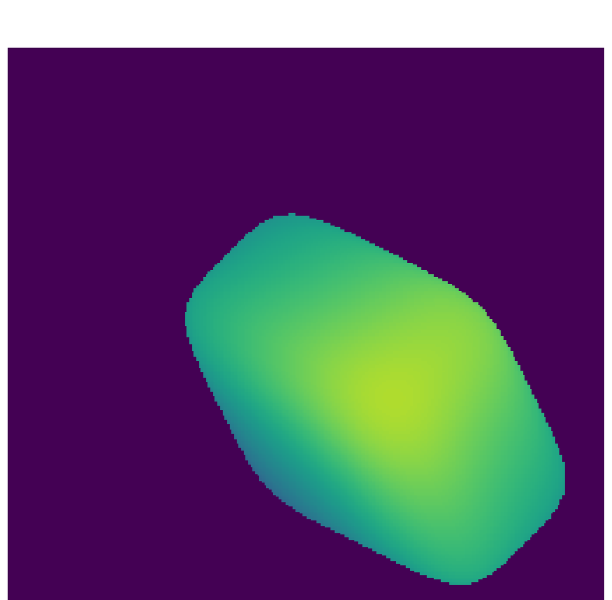


Figure 1: Example of a range image.

Dataset

Synthetic data is used to train and test the models. We generate the data in a controlled manner using a custom rendering tool, where superquadrics with arbitrary parameters can be created. This allows us to make a large dataset needed to train a deep neural network. Images are generated by rendering a single superquadric inside a $256 \times 256 \times 256$ grid, where first two dimensions are encoded as image width and height and the last one as depth, resulting in a 3D range image. Higher pixel values mean closer surfaces, while pixels with zero values represent the background.

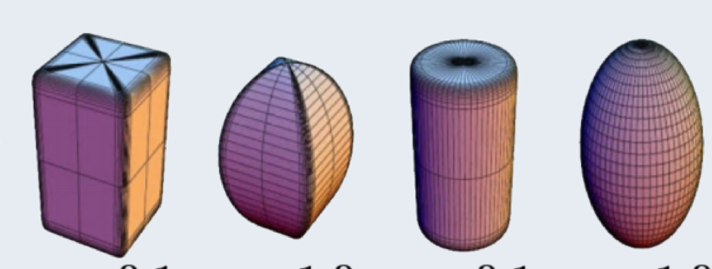
We use 100k images to train the network and 20k images to test the learned models.

Experiments

Three experiments were made. First model was trained on images, which included superquadrics in an isometric projection. The rotation was fixed and only shape, position and size were determined randomly. Second and third were trained on superquadrics with random rotations in all three axes. The main difference between the two was network output. In addition to block parameters (8 values), we also predict the rotation matrix (9 values) for the second model and quaternion coefficients (4 values) for the third. We use MSE to calculate the loss and Adam with learning rate of 0.01 for optimization. The learning rate is lowered by a factor of 10 after 250 and 750 epochs.

Superquadrics

$$\mathbf{F}(\mathbf{x},\mathbf{y},\mathbf{z}) = \left(\left(\frac{\mathbf{x}}{\mathbf{a_1}}\right)^{\frac{2}{\epsilon_2}} + \left(\frac{\mathbf{y}}{\mathbf{a_2}}\right)^{\frac{2}{\epsilon_2}}\right)^{\frac{\epsilon_2}{\epsilon_1}} + \left(\frac{\mathbf{z}}{\mathbf{a_3}}\right)^{\frac{2}{\epsilon_1}}$$



$\epsilon_1 = 0.1$	$\epsilon_1 = 1.0$	$\epsilon_1 = 0.1$	$\epsilon_1 = 1.0$
$\epsilon_2 = 0.1$	$\epsilon_2 = 0.1$	$\epsilon_2 = 1.0$	$\epsilon_2 = 1.0$

Architecture

We use a fully-convolutional architecture for our model. The architecture consists of 13 convolutional layers, each followed by a batch normalization layer and a ReLU activation layer.

The configuration of convolutional layers is as follows:

filters	size	stride
32	7	2
32	3	1
32	3	1
32	3	2
64	3	1
64	3	1
64	3	2
128	3	1
128	3	1
128	3	2
256	3	1
256	3	1
256	3	2

Table 1: Configuration of convolutional layers in our architecture.

The output of the final convolutional layer is flattened and used as an input to two separate fully-connected layers, each processing its own group of parameters (block + rotation).

Results

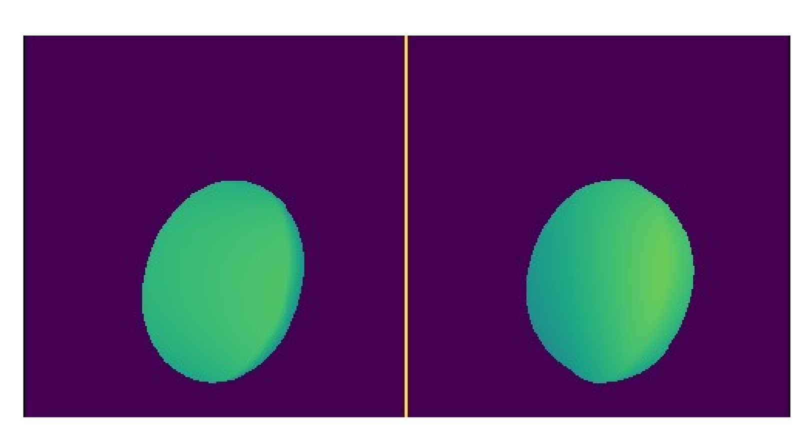


Figure 2: Comparison between a ground-truth (left) and predicted image (right).

The results for all three models are shown in table 1. Best results were achieved with model 1, which was trained on superquadrics in isometric projection. Size and position of shapes were predicted with an average error of a single pixel. With the addition of random rotations, some accuracy for predicting block parameters was lost.

attribute	range	model 1	model 2	model 3
size	[0 - 256]	**1.00**	11.76	9.42
shape	[0 - 1]	**0.02**	0.24	0.036
position	[0 - 256]	**1.13**	7.95	2.75
rot. (mat)	[0 - 1]	-	**0.66**	0.67
rot. (quat)	[0 - 1]	-	**0.45**	0.59

Table 2: Comparison of models. Mean absolute error for each parameter group is shown here.

Discussion

Model 1 already achieves satisfactory performance and could be used in practice if the object would be constrained to a single angle of projection. It is evident, however, that the prediction of rotational parameters with models 2 and 3 still presents a challenge for such network configuration. For example, an ellipsoid object with two or more axes with the same length could be described by an infinite amount of rotational parameters, while still being a perfect approximation of the original shape. Due to this ambiguous nature of rotational parameters, we believe that a custom loss function is needed to capture this geometric characteristic properly.

Conclusion

These preliminary results look promising. With a simple AlexNet-like architecture, we achieved pixel-close performance on a synthetically generated dataset of images consisting of a single superquadric in isometric projection. Recovery of rotational parameters still remains a challenge, which could be tackled by using a rotation-aware loss function. In the future, we also want to focus more on the segmentation and recovery of such parameters from superquadrics located in an unconstrained environment.

References

[1] F. Solina and R. Bajcsy. Recovery of parametric models from range images: The case for superquadrics with global deformations. *TPAMI*, 1990.

[2] K. Duncan, S. Sarkar, R. Alqasemi, and R. Dubey. Multi-scale superquadric fitting for efficient shape and pose recovery of unknown objects. In *ICRA*, 2013.

Contact

klemen.grm@fe.uni-lj.si
to1702@student.uni-lj.si
ales.jaklic@fri.uni-lj.si
vitomir.struc@fe.uni-lj.si
peter.peer@fri.uni-lj.si
franc.solina@fri.uni-lj.si

BMVA