CNN-based Superquadric Parameter Prediction via RGBD Image

Ryo Hachiuma, Yuko Ozasa and Hideo Saito
Graduate School of Science and Technology Keio University, Yokohama, Japan,
{ryo-hachiuma, yuko.ozasa, hs}@keio.jp

***Abstract*—Superquadrics represent various types of primitive shapes in a single equation with several parameters. Superquadric parameters of an object is estimated from 3D point cloud. However, the task of superquadric parameter estimation is computationally expensive and it is unacceptable for robot applications. Therefore, the concept of our research is to predict superquadric parameters directly from an image. We apply regression Convolutional Neural Network (CNN) that predicts parameters from images to superquadric parameter estimation. Moreover, we use not only depth but also RGB image to predict superquadric parameters for improve the accuracy of prediction. Experiments show that proposed CNN with the input of RGB-D image predicts more accurate parameters than baseline methods.**

1. Introduction

Approximating shape of objects into 3D primitives such as cuboids, cylinders and spheres is applied to the task of object grasping. Approaches to approximate objects into a single primitivie shape have been proposed [1], [2]. As the goal of these methods is to grasp specific objects, it is sufficient to approximate shape into limited kinds of primitives. On the other hand, Quispe *et al.* applied superquadric as primitive shape representation for complete shape representation from a single depth image to grasp household objects [3]. We employ superquadrics as primitive shape representation.

Superquadrics are parametrizable models that offer a large variety of different primitive shapes with a single equation [4]. Applying the superquadric to an object enables the object to be expressed by various primitives, such as cuboids, cylinders and spheres, with several parameters.

Superquadric parameters of an object are estimated with an image of 3D point cloud captured from a single viewpoint [5]. An equation obtained by substituting the 3D point cloud of an object into superquadric representation is regarded as a non-linear least squares problem. Superquadric parameters are estimated using the Levenberg-Marquardt (LM) algorithm [6].

However, estimating superquadric parameters of the 3D point cloud is a computationally expensive task and it is unacceptable for real-time robotics [7]. To address this issue, Duncan *et al.* proposed rapid superquadric parameter fitting by multi-scale voxelization [8]. They achieved reducing the computational time while maintaining the accuracy of superquadric parameter estimation.

Superquadric parameter estimation requires preprocessing of the captured depth image to generate 3D point cloud of the object. As object detection has become possible with high accuracy [9], we assume that the object detection is applied to the captured depth image beforehand in this paper. After the object is detected, background extraction and noise reduction are required to generate 3D point cloud of the object. These preprocessing tasks would be also time consuming. Therefore, the concept of our approach is to predict superquadric parameters directly from a depth image of an object.

Nowadays, there are methods which predict parameters from a single image using regression CNN [9], [10]. As superquadric parameter estimation can be regarded as a regression problem, we apply regression Convolutional Neural Network (CNN) to predict superquadric parameters from a depth image of object.

Moreover, it is known that the prediction accuracy improved at the task of object detection [11] or object recognition [12] by using not only depth but also color information as inputs of CNN. Therefore, we use both RGB and depth images (RGBD image) for superquadric prediction.

In this paper, we present CNN-based superquadric parameter prediction method from both RGBD image. In the experiment, we evaluate the prediction accuracy of our method verify the effectiveness of using RGBD image for the prediction. We also show that the prediction using our method is robust against the viewpoint in which the object was captured.

2. Superquadric parameters

Superquadrics have been introduced in computer graphics in 1981 [4], as a generalization of quadrics and has been well studied in graphics and computer vision [13]. The topic was started from superquadric parameter estimation from a depth image [5]. Thus, superquadrics have been used for object shape approximation [14], [15], object recognition [16], [17], object segmentation [18], and object grasping [19], [20], [21].

A Superquadric is defined by the inside-outside function with a shape parameter $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2)$ and a scale parameter $\boldsymbol{s} = (s_1, s_2, s_3)$:

$$F(x,y,z,\boldsymbol{q}) = \left\{ \left(\frac{x}{s_1}\right)^{\frac{2}{\varepsilon_2}} + \left(\frac{y}{s_2}\right)^{\frac{2}{\varepsilon_2}} \right\}^{\frac{\varepsilon_2}{\varepsilon_1}} + \left(\frac{z}{s_3}\right)^{\frac{2}{\varepsilon_1}}, \quad (1)$$

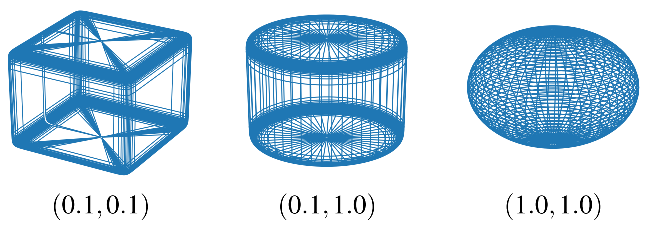


Figure 1: Some examples of superquadric surfaces according to $(\varepsilon_1, \varepsilon_2)$.

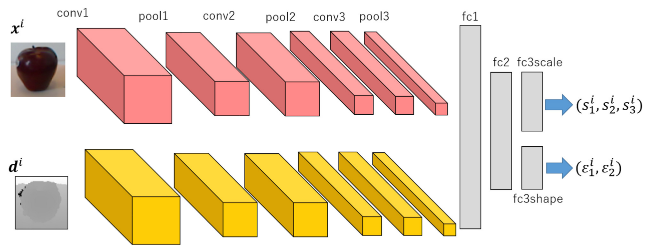


Figure 2: Two-stream regression convolutional neural network for superquadric parameter prediction. The input of the network is a pair of RGB and depth images.

where $\boldsymbol{q}$ is a 2-tuple as $(\boldsymbol{s}, \boldsymbol{\varepsilon})$. Parameters s_1, s_2, and s_3 are scale parameters that define the superquadric size at the x, y, and z coordinates, respectively. Parameters ε_1, ε_2 are shape parameters that express squareness along the z axis and the x-y plane. Fig. 1 shows some examples of superquadric surfaces according to shape parameters. For example, superquadric surface is spherical with $(\varepsilon_1, \varepsilon_2) = (1.0, 1.0)$.

Given a point (x, y, z), if $F < 1$, the point is inside the superquadric, if $F > 1$, the point is outside the superquadric, and if $F = 1$ the point lies on the surface of superquadric. Superquadric parameter $\boldsymbol{q}$ can be estimated from given K 3D points $\boldsymbol{p}_i = (x_i, y_i, z_i)$ of 3D point cloud. The minimization of the algebraic distance from points to the superquadric model can be solved by defining a non-linear least-squares minimization problem:

$$\min_{\boldsymbol{q}} \sum_{i=0}^{K} \left(\sqrt{s_1 s_2 s_3}\left(F(\boldsymbol{p}_i, \boldsymbol{q}) - 1\right)\right)^2, \tag{2}$$

where $(F(\mathbf{p}_i, \boldsymbol{q}) - 1)^2$ indicates the point to superquadric surface distance minimization, where the term $\sqrt{s_1 s_2 s_3}$ is proportional to superquadric volume, compensates for the fact that the previous equation is biased toward larger superquadric surfaces. In the literature, Eq. (2) can be solved via the Levenberg-Marquardt (LM) algorithm [6]. It is known that (2) is numerically unstable when $\varepsilon_1, \varepsilon_2 < 0.1$ and the superquadric surface has concavities with $\varepsilon_1, \varepsilon_2 > 2$. Constraints are applied [14] when minimizing (2) for shape parameters; $0.1 \leq \varepsilon_1, \varepsilon_2 \leq 2$ and for the scale parameters; $s_1, s_2, s_3 > 0$.

3. CNN-based superquadric parameter prediction

Our network aims at predicting superquadric parameters with the input of a pair of RGB and depth images. Network architecture is as simple as shown in Fig. 2. Our architecture consists of two convolutional network streams operating on color and depth information respectively. The network automatically learns to combine these two processing streams in a late fusion approach. The architecture of each RGB and depth stream is based on the architecture proposed by Li and Chan [22]. They use the regression CNN for human joint prediction, and their model performs well to predict joint position from an image. The architecture of late fusion is based on the idea proposed by Karpathy *et al.* [23]. Their late fusion network merged two streams in the first fully connected layer.

For training our model to predict superquadric parameters, training data, which consists of RGB images, depth images of objects and corresponding parameters, are required. We assume that the object detection is already applied to the scene. The superquadric parameters have to be computed beforehand to train the model.

Notation. Let T be a sample for training our model as

$$T = (\boldsymbol{x}, \boldsymbol{d}, \boldsymbol{q}), \tag{3}$$

where $\boldsymbol{x} \in \mathbb{R}^{H \times W}$ denotes the cropped RGB image with the image height H and width W, $\boldsymbol{d} \in \mathbb{R}^{H \times W}$ denotes the cropped depth image and $\boldsymbol{q} = (\boldsymbol{s}, \boldsymbol{\varepsilon})$ is a 2-tuple of superquadric parameters corresponding to the RGB and depth image. Also, $\boldsymbol{\varepsilon}$ and $\boldsymbol{s}$ denote a 2-tuple of shape parameters $(\varepsilon_1, \varepsilon_2)$ and a 3-tuple of scale parameters (s_1, s_2, s_3), respectively.

Loss Function. As the characteristics of shape parameters $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2)$ and scale parameters $\boldsymbol{s} = (s_1, s_2, s_3)$ are different, the loss of shape parameters and scale parameters are calculated, separately.

We use the mean squared error as the loss function. We define a loss for shape parameters as

$$L_{shape}(\hat{\boldsymbol{\varepsilon}}, \boldsymbol{\varepsilon}) = \frac{1}{n_1} \sum_{i}^{n_1} \|\hat{\varepsilon}_i - \varepsilon_i\|^2, \tag{4}$$

where $\boldsymbol{\varepsilon}$ is the ground truth of shape parameter, $\hat{\boldsymbol{\varepsilon}}$ is the predicted shape parameter, and n_1 is the number of elements of shape parameter, which is $n_1 = 2$. Likewise, we define a loss for scale parameters as

$$L_{scale}(\hat{\boldsymbol{s}}, \boldsymbol{s}) = \frac{1}{n_2} \sum_{i}^{n_2} \|\hat{s}_i - s_i\|^2, \tag{5}$$

where $\boldsymbol{s}$ is the ground truth of scale parameter, $\hat{\boldsymbol{s}}$ is the predicted scale parameter, and n_2 is the number of elements of scale parameter, which is $n_2 = 3$. With weighted summing for these loss functions, we define a loss as

$$L_R(\hat{\boldsymbol{q}}, \boldsymbol{q}) = L_{shape}(\hat{\boldsymbol{\varepsilon}}, \boldsymbol{\varepsilon}) + w_s L_{scale}(\hat{\boldsymbol{s}}, \boldsymbol{s}), \tag{6}$$

where $\boldsymbol{q}$ is the ground truth of superquadric parameter, $\hat{\boldsymbol{q}}$ is the predicted superquadric parameter, and w_s is the hyper-parameter for weighting between shape loss L_{shape} and scale loss L_{scale}.

Network Architecture. The architecture of proposed CNN is shown in Fig. 2. Our network consists of two streams - processing color and depth data independently - which are combined in a late fusion approach, and predicts superquadric parameters $\boldsymbol{q}$. Scale parameters $\hat{\boldsymbol{s}}$ are predicted at the output of layer fc3scale, and shape parameters $\hat{\boldsymbol{\varepsilon}}$ are predicted at the output of layer fc3shape. The loss is computed at each fc3scale, fc3shape layer by (6). For the output of convolutional and fully connected layers, the activation function is applied to learn non-linearity. All layers except fc3shape are activated with Rectified Linear Unit (ReLU) activation function [24], which showed that ReLU has good performance to train fast. For an activation function for fc3shape, we use the f_ε defined as

$$f_\varepsilon(u) = \frac{2}{1+exp(-u)}. \tag{7}$$

The function f_ε is the sigmoid function multiplied by 2. The reason to apply the this function instead of ReLU is that the range of shape parameters are set to from 0 to 2 when estimating superquadric parameter in (2).

Network Training. For training, dropout [25] is applied to the first fully-connected layers (fc1) to prevent over-fitting. Also, spatial dropout [26] is applied to the first convolutional layer (conv1), which extends the value across the entire feature map so that it improves the performance especially when the training set size is small.

4. Experimental Setup

For the evaluation of parameter prediction, the dataset used in the experiment, detailed information of training, and training loss and validation loss are described below.

4.1. Dataset

A dataset was constructed for the evaluation of the accuracy of superquadric parameter prediction. The dataset is composed of pairs of RGB-D images of objects and superquadric parameters of each object.

RGB and depth images are chosen from the RGB-D Object Dataset [27]. The RGB-D Object Dataset [27] contains 51 categories of objects, and it consists of RGB-D images, and 3D point clouds of 300 distinct objects. Each object is captured on a turntable from different viewpoints. These are common household objects, such as apples, cameras, food cans, keyboards, shampoo, and toothpastes.

We selected objects of nine categories from the RGB-D Object Dataset [27]. The dataset is composed of a combination of the RGB-D images and superquadric parameters estimated using the LM algorithm from 3D point clouds. For the training of our CNN model, RGB-D images and superquadric parameters are used. We splited training, validation and test data with the ratio of around $6:2:2$ per each category randomly. There are around $25,000$ RGB-D images and superquadric parameters for training, $8,100$ for the validation and $8,100$ for the test. As we randomly split train and test data per category, test set includes data which were captured from the viewpoints not included in the training set. We also evaluate the robustness of superquadric parameter prediction against viewpoints with the dataset.

layer type	kernel	stride	output size	activation fn
conv1	7×7	2×2	$72\times72\times32$	ReLU
pool1	–	2×2	$36\times36\times32$	–
conv2	5×5	1×1	$36\times36\times16$	ReLU
pool2	2×2	2×2	$18\times18\times16$	–
conv3	3×3	1×1	$16\times16\times16$	ReLU
pool3	2×2	1×1	$8\times8\times16$	–
fc1	–	–	256	ReLU
fc2	–	–	256	ReLU
fc3scale	–	–	3	ReLU
fc3shape	–	–	2	f_ε

TABLE 1: Network parameters of each layer. The name in the column of layer type corresponds to the layer visualized in Fig. 2.

4.2. Training details

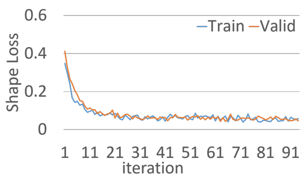
The network detail of our model is shown in Table 1. At the row of layer type, conv, pool, and fc is the abbreviation of convolutional, pooling, and fully connected layer, respectively. The row of layer type corresponds to the layer named in Fig. 2.

We assume that the objects are detected beforehand. The bounding box of objects are given at the RGB-D Object Dataset [27]. The input is the cropped RGB and depth image of objects, and the input of RGB and depth images is all rescaled to 149x149.

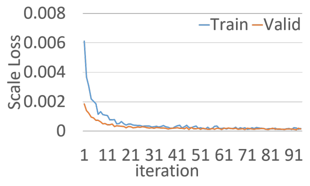
We set the size of the data batch to 256. Weights of the convolutional and fully-connected layers were initialized by a zero-mean Gaussian distribution with a standard deviation of 0.1. We use Adam optimizer [28] with the learning rate 0.00001 to train our CNN according to the loss described in (6). The parameter weight w_s between shape loss and scale loss is tuned to 100.0 in (6). The dropout probability of dropout and spatial dropout is set to be 0.5 in the experiments. We applied early stopping [29] technique that stops training the network when the validation loss is no longer decreased. Training and evaluations are performed on a PC equipped with NVIDIA GTX 1080 GPU.

4.3. Training Loss and validation loss

To confirm the convergence of the training and validation loss of proposed CNN, Fig. 3 shows the train and validation loss of both shape and scale parameters. For both scale and shape losses are decreased over training iterations. Early stopping is applied after 95 iterations where the loss had begun to increase.



(a)



(b)

Figure 3: Plots of the shape and scale losses of proposed CNN during training iterations. (a): loss of training shape parameters, (b): loss of training scale parameters.

5. Experimental Results

We set two baseline methods to evaluate the robustness of proposed CNN against viewpoints. The first is the network with the input of RGB image, and the second is the network with the input of depth image.

5.1. Visualization of predicted parameters

Fig. 4 shows the qualitative results of superquadric parameter prediction. The first row shows the RGB images of objects, and the second row shows the superquadric surface estimated by LM algorithm, the third row shows the surface predicted by CNN with the input of RGB image, the fourth row shows the surface predicted by CNN with the input of depth image, and the last row shows the surface predicted by CNN with the input of both RGB and depth image. Predicted and estimated superquadric parameters are shown below each superquadric surface. For example, scale parameters of apple predicted by proposed CNN are $(0.033, 0.041, 0.048)$ and shape parameters are $(0.774, 0.758)$.

Visualization of the superquadric surface of the predicted parameters by proposed CNN with the input of RGBD image can demonstrate that primitive shapes were successfully approximated objects' shape. For example, spherical surface is predicted at the category of apple, cubical surface is predicted at the category of food box and kleenex, and cylinderical surface is predicted at the category of flashlight and food can.

5.2. Prediction error comparison

Prediction error of scale and shape parameters are evaluated using mean absolute error. Table 2 shows the prediction error of shape and scale parameters per category with two baselines. The mean prediction error of both shape and scale parameters of proposed CNN is less than it of baselines. Our proposed CNN predicted accurate parameters against data captured from different viewpoints in the training dataset.

Moreover, to compare predicted shape and scale parameters as superquadric parameters, we employ the error of volume of predicted superquadric surface. The volume of superquadric can be calculated from both scale and shape parameters [30]. Volume error of superquadric surface is evaluated using mean absolute error between ground truth surface and predicted surface. The volume error is summarized in Table 3. It also shows the robustness of proposed CNN against viewpoints.

6. Conclusion

In this paper, we have presented an approach for superquadric parameter prediction from RGBD image. Our proposed approach is the two-stream regression CNN with the input of a pair of RGB and depth images. It consists of two convolutional network streams operating on color and depth information, respectively. The network trains to combine these two processing streams in a late fusion approach. The experimental results showed the effectiveness of using RGBD image for the parameter prediction. Additionally, the robustness against the viewpoint in which the object was captured is evaluated. We are currently exploring an extension to our research to predict not only superquadric parameters but also rotation and translation parameters of object.

References

[1] K. Harada, K. Nagata, T. Tsuji, N. Yamanobe, A. Nakamura, and Y. Kawai, "Probabilistic approach for object bin picking approximated by cylinders," in *ICRA*, 2013, pp. 3742–3747.

[2] K. Huebner and D. Kragic, "Selection of robot pre-grasps using box-based shape approximation," in *IROS*, 2008, pp. 1765–1770.

[3] A. H. Quispe, B. Milville, M. A. Gutirrez, C. Erdogan, M. Stilman, H. Christensen, and H. B. Amor, "Exploiting symmetries and extrusions for grasping household objects," in *ICRA*, 2015, pp. 3702–3708.

[4] A. H. Barr, "Superquadrics and angle-preserving transformations," *IEEE Computer graphics and Applications*, vol. 1, no. 1, pp. 11–23, 1981.

[5] F. Solina and R. Bajcsy, "Range image interpretation of mail pieces with superquadrics," in *AAAI*, 1987, pp. 733–737.

[6] J. J. Moré, "The levenberg-marquardt algorithm: implementation and theory," in *Numerical analysis*, 1978, pp. 105–116.

[7] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz, "Close-range scene segmentation and reconstruction of 3d point cloud maps for mobile manipulation in domestic environments," in *IROS*, 2009, pp. 1–6.

[8] 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, pp. 4238–4243.

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *CVPR*, 2016, pp. 779–788.

[10] S. Li, Z.-Q. Liu, and A. B. Chan, "Heterogeneous multi-task learning for human pose estimation with deep convolutional neural network," in *CVPR*, 2014, pp. 482–489.

[11] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from rgb-d images for object detection and segmentation," in *ECCV*. Springer, 2014, pp. 345–360.

[12] A. Eitel, J. T. Springenberg, L. Spinello, M. A. Riedmiller, and W. Burgard, "Multimodal deep learning for robust RGB-D object recognition," *CoRR*, vol. abs/1507.06821, 2015. [Online]. Available: http://arxiv.org/abs/1507.06821

[13] A. P. Pentland, "Perceptual organization and the representation of natural form," *Artificial Intelligence*, vol. 28, no. 3, pp. 293–331, 1986.

Error	input	apple	cereal box	flashlight	food box	food can	kleenex	lime	soda can	tomato	mean
shape	RGB	0.186	0.071	0.159	0.055	0.181	0.074	0.259	0.182	0.238	0.156
	Depth	0.197	0.080	0.164	0.085	0.191	0.108	**0.265**	0.192	0.245	0.170
	RGB-D	**0.183**	**0.048**	**0.149**	**0.042**	**0.172**	**0.070**	0.269	**0.165**	**0.237**	**0.148**
scale $(\times 10^{-1})$	RGB	0.053	0.193	0.124	0.126	0.075	0.115	**0.051**	0.080	0.057	0.097
	Depth	0.076	0.240	0.155	0.150	0.085	0.168	0.055	0.110	0.067	0.123
	RGB-D	**0.050**	**0.166**	**0.119**	**0.092**	**0.064**	**0.103**	0.052	**0.075**	**0.055**	**0.086**

TABLE 2: Error of predicted shape and scale comparison.

input	apple	cereal box	flashlight	food box	food can	kleenex	lime	soda can	tomato	mean
RGB	0.014	0.282	0.023	0.117	0.027	0.086	**0.003**	0.023	0.009	0.065
Depth	0.021	0.342	0.034	0.015	0.037	0.151	0.004	0.037	0.170	0.087
RGB-D	**0.012**	**0.223**	**0.021**	**0.072**	**0.023**	**0.062**	0.004	**0.018**	**0.008**	**0.049**

TABLE 3: Error of predicted volume $(\times 10^{-2})$ comparison.

[14] F. Solina and R. Bajcsy, "Recovery of parametric models from range images: The case for superquadrics with global deformations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 2, pp. 131–147, 1990.

[15] M. Strand, Z. Xue, M. Zoellner, and R. Dillmann, "Using superquadrics for the approximation of objects and its application to grasping," in *ICRA*, 2010, pp. 48–53.

[16] W. Xing, W. Liu, and B. Yuan, "Superquadric-based geons recognition utilizing support vector machines," in *ICSP*, 2004, pp. 1264–1267.

[17] R. Hachiuma, Y. Ozasa, and H. Saito, "Primitive shape recognition via superquadric representation using large margin nearest neighbor classifier," in *VISAPP*, 2017, pp. 325–332. [Online]. Available: https://doi.org/10.5220/0006153203250332

[18] L. Chevalier, F. Jaillet, and A. Baskurt, "Segmentation and superquadric modeling of 3d objects," in *WSCG*, 2003, pp. 232–239.

[19] A. T. Miller and P. K. Allen, "Graspit! a versatile simulator for robotic grasping," *IEEE Robotics Automation Magazine*, vol. 11, no. 4, pp. 110–122, 2004.

[20] K. M. Varadarajan and M. Vincze, "Object part segmentation and classification in range images for grasping," in *ICAR*, 2011, pp. 21–27.

[21] G. Vezzani, U. Pattacini, and L. Natale, "A grasping approach based on superquadric models," in *ICRA*, 2017, pp. 1579–1586.

[22] S. Li and A. B. Chan, "3d human pose estimation from monocular images with deep convolutional neural network," in *ACCV*, 2014, pp. 332–347.

[23] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp. 1725–1732.

[24] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010, pp. 807–814.

[25] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[26] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *CVPR*, 2015, pp. 648–656.

[27] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view rgb-d object dataset," in *ICRA*, 2011, pp. 1817–1824.

[28] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[29] L. Prechelt, *Early Stopping — But When?* Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 53–67.

[30] A. Jaklic, A. Leonardis, and F. Solina, *Segmentation and recovery of superquadrics.* Springer Science & Business Media, 2013, vol. 20.

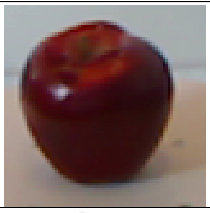
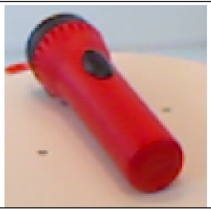
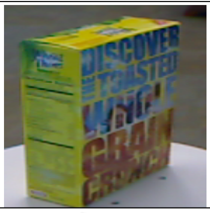
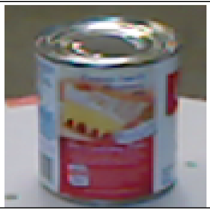
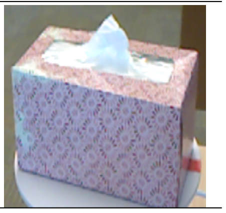
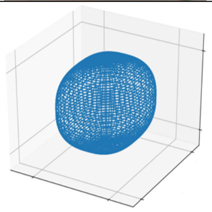
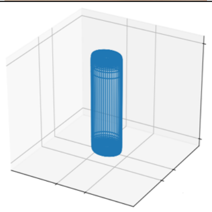
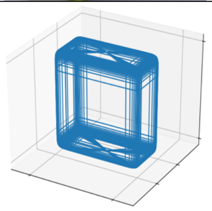
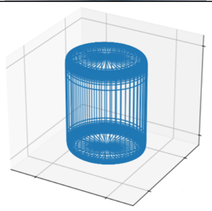
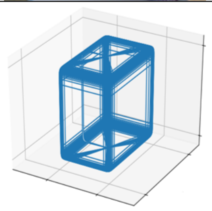
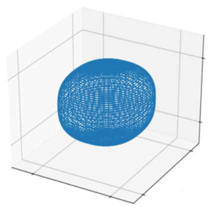
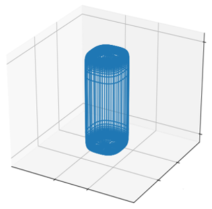
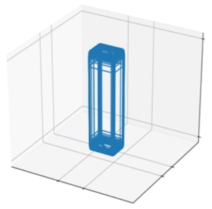
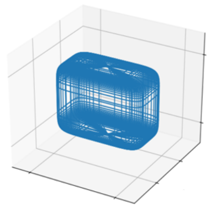
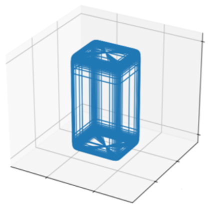
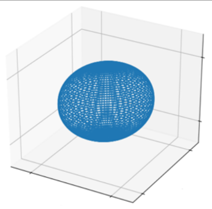
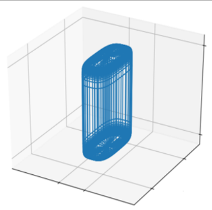
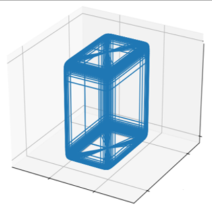
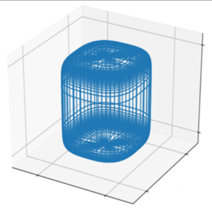
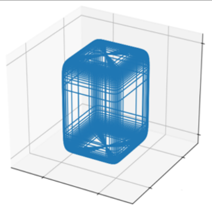
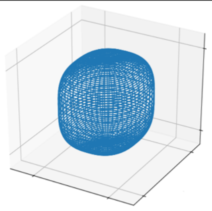
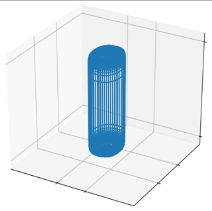
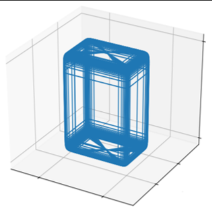
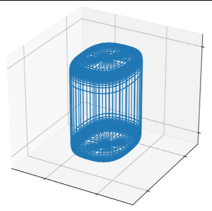
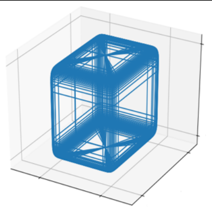
	apple	flashlight	food box	food can	kleenex
RGB image					
Ground Truth surface					
(s_1, s_2, s_3) $(\varepsilon_1, \varepsilon_2)$	$(0.027, 0.032, 0.033)$ $(0.604, 0.788)$	$(0.025, 0.019, 0.092)$ $(0.107, 1.023)$	$(0.085, 0.039, 0.110)$ $(0.118, 0.131)$	$(0.040, 0.037, 0.055)$ $(0.100, 1.02)$	$(0.031, 0.050, 0.062)$ $(0.100, 0.100)$
CNN with RGB					
(s_1, s_2, s_3) $(\varepsilon_1, \varepsilon_2)$	$(0.027, 0.030, 0.025)$ $(0.722, 0.857)$	$(0.024, 0.026, 0.075)$ $(0.140, 0.695)$	$(0.033, 0.024, 0.140)$ $(0.098, 0.096)$	$(0.027, 0.014, 0.024)$ $(0.251, 0.319)$	$(0.036, 0.031, 0.079)$ $(0.105, 0.172)$
CNN with depth					
(s_1, s_2, s_3) $(\varepsilon_1, \varepsilon_2)$	$(0.028, 0.032, 0.026)$ $(0.965, 0.937)$	$(0.015, 0.035, 0.058)$ $(0.114, 0.604)$	$(0.043, 0.072, 0.102)$ $(0.100, 0.159)$	$(0.033, 0.036, 0.047)$ $(0.215, 0.673)$	$(0.036, 0.040, 0.066)$ $(0.209, 0.184)$
CNN with RGBD					
(s_1, s_2, s_3) $(\varepsilon_1, \varepsilon_2)$	$(0.033, 0.041, 0.038)$ $(0.774, 0.758)$	$(0.022, 0.028, 0.081)$ $(0.129, 0.838)$	$(0.064, 0.041, 0.107)$ $(0.114, 0.120)$	$(0.024, 0.032, 0.044)$ $(0.120, 0.722)$	$(0.039, 0.055, 0.065)$ $(0.171, 0.129)$

Figure 4: Visualization of superquadric surfaces estimated by LM algorithm (ground truth), predicted by CNN with the input of RGB image, predicted by CNN with the input of depth image, and predicted by CNN with the input of RGBD image (proposed).