Supplementary Material for Superquadrics Revisited: Learning 3D Shape Parsing beyond Cuboids

Despoina Paschalidou[1,4] Ali Osman Ulusoy[2] Andreas Geiger[1,3,4]
[1]Autonomous Vision Group, MPI for Intelligent Systems Tübingen
[2]Microsoft [3]University of Tübingen [4]Max Planck ETH Center for Learning Systems
{firstname.lastname}@tue.mpg.de

Abstract

*In this **supplementary document**, we present a detailed derivation of the proposed analytical solution to the Chamfer loss, which avoids the need for computationally expensive reinforcement learning or iterative prediction. Moreover, we also present additional qualitative results on more complex object categories from the ShapeNet dataset [2] such as cars and motorbikes and on the SURREAL human body dataset [6]. Furthermore, we also show results on primitive prediction when using RGB images instead of 3D occupancy grids as input. Finally, we empirically demonstrate that our bi-directional Chamfer loss formulation indeed works better and results in less local minima than the original bi-directional loss formulation of Tulsiani et al. [5].*

1. Superquadrics

In this work, we propose superquadrics as a shape primitive representation. Their simple parametrization in combination to their ability to represent a diverse class of shapes makes superquadrics a natural choice for geometric primitives. Moreover, their continuous parametrization is suitable for deep learning as their shape varies continuously with their parameters. Superquadrics are fully modelled using a set of 11 parameters [1]. The **explicit superquadric equation** defines the surface vector $\mathbf{r}$

$$\mathbf{r}(\eta,\omega) = \begin{bmatrix} \alpha_1 \cos^{\epsilon_1}\eta \cos^{\epsilon_2}\omega \\ \alpha_2 \cos^{\epsilon_1}\eta \sin^{\epsilon_2}\omega \\ \alpha_3 \sin^{\epsilon_1}\eta \end{bmatrix} \quad \begin{matrix} -\pi/2 \leq \eta \leq \pi/2 \\ -\pi \leq \omega \leq \pi \end{matrix} \tag{1}$$

where $\alpha = [\alpha_1, \alpha_2, \alpha_3]$ determine the size and $\epsilon = [\epsilon_1, \epsilon_2]$ determine the global shape of the superquadric. Fig. 1 visualizes the shape of superquadrics for different values of ϵ_1 and ϵ_2. In addition to the shape parameters, we also associate a rigid body transformation with each superquadric. This transformation is represented by a translation vector $\mathbf{t} = [t_x, t_y, t_z]$ and a quaternion $\mathbf{q} = [q_0, q_1, q_2, q_3]$ that determines the coordinate system transformation $\mathcal{T}(\mathbf{x}) = \mathbf{R}(\lambda)\,\mathbf{x} + \mathbf{t}(\lambda)$ from world coordinates to local primitive-centric coordinates. This transformation as well as the angles η, ω and the scale parameters $\alpha_1, \alpha_2, \alpha_3$ are illustrated in Fig. 2.

2. Derivation of Pointcloud-to-Primitive Loss

This section provides the derivation of the pointcloud-to-primitive distance $\mathcal{L}_{X\rightarrow P}(\mathbf{X}, \mathbf{P})$ in Eq. 11 of the main paper. For completeness, we restate our notation briefly. We represent the target point cloud as a set of 3D points $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{N}$ and we approximate the continuous surface of the m^{th} primitive by a set of 3D points $\mathbf{Y}_m = \{\mathbf{y}_k^m\}_{k=1}^{K}$. We further denote $\mathcal{T}_m(\mathbf{x}) = \mathbf{R}(\lambda_m)\,\mathbf{x} + \mathbf{t}(\lambda_m)$ as the mapping from world coordinates to the local coordinate system of the m^{th} primitive.

The pointcloud-to-primitive distance, $\mathcal{L}_{X\rightarrow P}$, measures the distance from the point cloud to the primitives to ensure that each observation is explained by at least one primitive. It can be expressed as:

$$\mathcal{L}_{X\rightarrow P}(\mathbf{X}, \mathbf{P}) = \mathbb{E}_{p(\mathbf{z})}\left[\sum_{\mathbf{x}_i \in \mathbf{X}} \min_{m|z_m=1} \Delta_i^m\right] \tag{2}$$

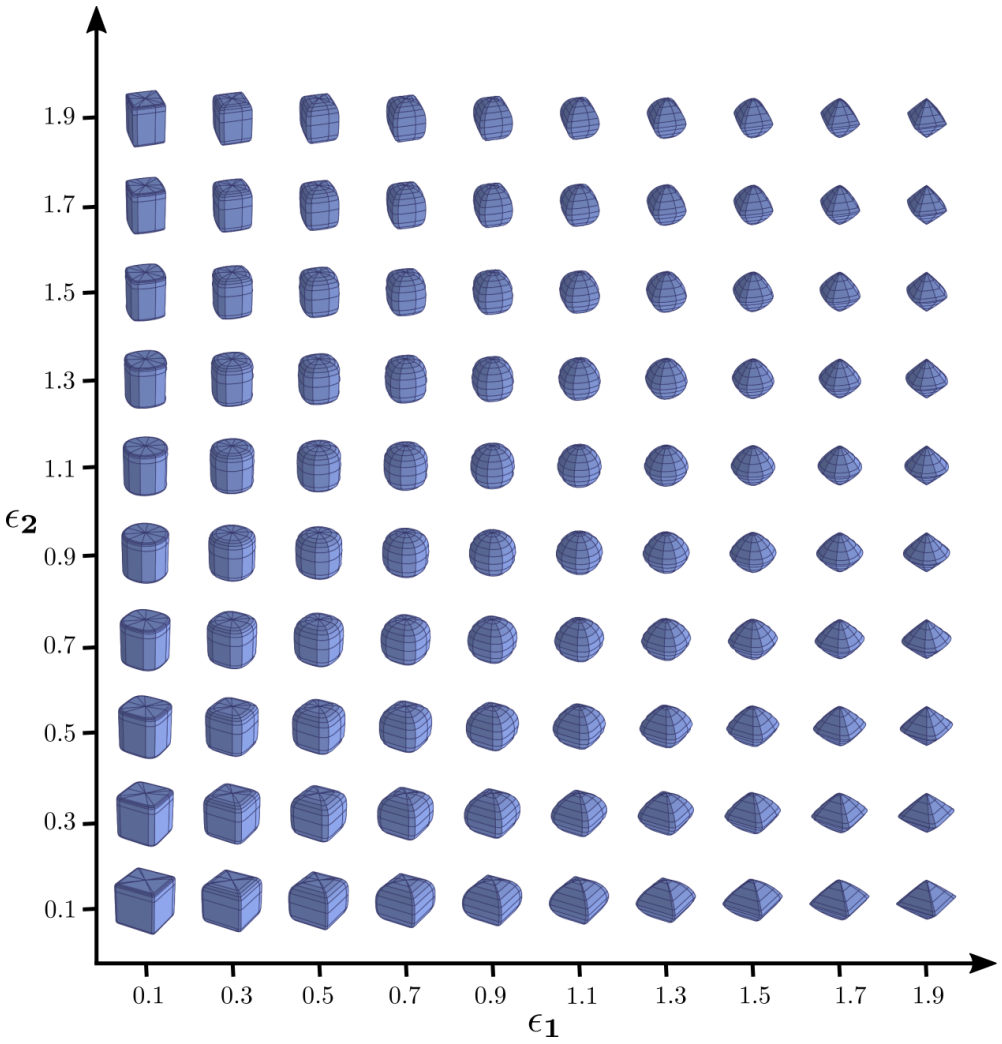


Figure 1: **Superquadric Shape Space.** Superquadrics are a parametric family of surfaces that can be used to describe cubes, cylinders, spheres, octahedral ellipsoids, etc. [1]. This figure visualizes superquadrics when varying the shape parameters ϵ_1 and ϵ_2, while keeping the size parameters α_1, α_2 and α_3 constant.

where Δ_i^m denotes the minimal distance from point $\mathbf{x}_i$ to the surface of the m'th primitive:

$$\Delta_i^m = \min_{k=1,..,K} \|\mathcal{T}_m(\mathbf{x}_i) - \mathbf{y}_k^m\|_2 \tag{3}$$

Assuming independence of the existence variables $p(\mathbf{z}) = \prod_m p(z_m)$, we can replace the expectations in (2) with summations as follows:

$$\mathcal{L}_{X \to P}(\mathbf{X}, \mathbf{P}) = \sum_{z_1} \cdots \sum_{z_M} \left[\sum_{\mathbf{x}_i \in \mathbf{X}} \min_{m|z_m=1} \Delta_i^m \right] p(\mathbf{z}) \tag{4}$$

Naïve computation of (4) has exponential complexity, i.e. for M primitives it requires evaluating the quantity inside the expectation 2^M times. Our key insight is that (4) can be evaluated in linear time if the distances Δ_i^m are sorted. Without loss of generality, we assume that the distances are sorted in ascending order. This allows us to state the following: if the first primitive exists, the first primitive will be the one closest to point $\mathbf{x}_i$ of the target point, if the first primitive does not exist and the second does, then the second primitive is closest to point $\mathbf{x}_i$ and so forth. More formally, this property can be stated

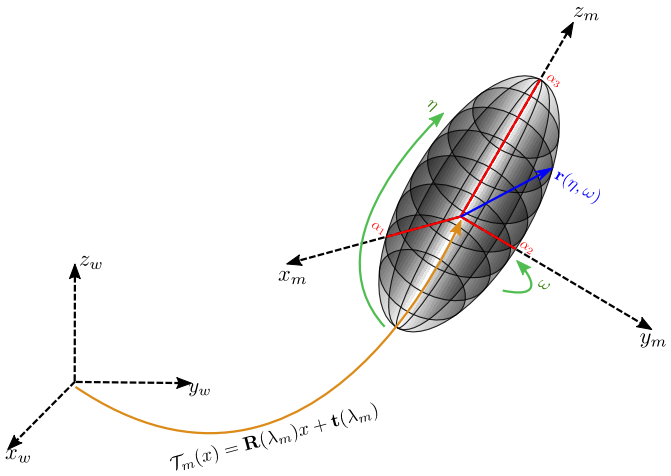


Figure 2: **Explicit Superquadric Equation**. A $3D$ vector $\mathbf{r}(\eta, \omega)$ defines a closed surface in space as η (latitude angle) and ω (longitude angle) change in the given intervals (1). The rigid body transformation $\mathcal{T}_m(x)$ maps a point from the world coordinate system to the local coordinate system of the m^{th} primitive.

as follows:

$$\min_{m|z_m=1} \Delta_i^m = \begin{cases} \Delta_i^1, & \text{if } z_1 = 1 \\ \Delta_i^2, & \text{if } z_1 = 0, z_2 = 1 \\ \vdots & \\ \Delta_i^M, & \text{if } z_m = 0, \ldots, z_M = 1 \end{cases} \tag{5}$$

Using (5) we can simplify (4) as follows. We start to carry out the summations over the existence variables one by one. Starting with the summations over z_1, (4) becomes:

$$\mathcal{L}_{X\to P}(\mathbf{X}, \mathbf{P}) = \sum_{\mathbf{x}_i \in \mathbf{X}} \left[\underbrace{\gamma_1 \sum_{z_2} \cdots \sum_{z_M} \Delta_i^1 \prod_{\bar{m}=2}^{M} p(z_{\bar{m}})}_{(\dagger)} + (1-\gamma_1) \sum_{z_2} \cdots \sum_{z_M} \left[\min_{m\geq 2|z_m=1} \Delta_i^m \right] \prod_{\bar{m}=2}^{M} p(z_{\bar{m}}) \right] \tag{6}$$

The expression, marked with (†) corresponds to the case for $z_1 = 1$, namely the 1^{st} primitive is part of the scene. From Eq. 5, we know that $\min_{m|z_m=1} \Delta_i^m = \Delta_i^1$ for $z_1 = 1$, thus the expression marked with (†), can be simplified as follows,

$$(\dagger) = \gamma_1 \Delta_i^1 \underbrace{\sum_{z_2} \cdots \sum_{z_M} \prod_{\bar{m}=2}^{M} p(z_{\bar{m}})}_{\text{this term evaluates to 1}} = \gamma_1 \Delta_i^1 \tag{7}$$

Following this strategy, we can iteratively simplify the remaining terms in (6) and arrive at the analytical form of the pointcloud-to-primitive distance stated in Eq. 11 in the main paper:

$$\begin{aligned} \mathcal{L}_{X\to P}(\mathbf{X}, \mathbf{P}) &= \sum_{\mathbf{x}_i \in \mathbf{X}} \left[\gamma_1 \Delta_i^1 + (1-\gamma_1)\gamma_2 \Delta_i^2 + \cdots + (1-\gamma_1)(1-\gamma_2)\ldots\gamma_M \Delta_i^M \right] \\ &= \sum_{\mathbf{x}_i \in \mathbf{X}} \sum_{m=1}^{M} \Delta_i^m \gamma_m \prod_{\bar{m}=1}^{m-1} (1-\gamma_{\bar{m}}) \end{aligned} \tag{8}$$

Note that our current formulation assumes that at least one primitive exists in the scene. However, this assumption can be easily relaxed by introducing a "virtual primitive" with a fixed distance to every 3D point on the target point cloud.

3. Qualitative Results on SURREAL

In this section, we provide additional qualitative results on the SURREAL human body dataset. In Fig. 3, we illustrate the predicted primitives of humans in various poses and articulations.

Figure 3: **Qualitative Results on SURREAL.** Our network learns semantic mappings of body parts across different body shapes and articulations. For instance, the network uses the same primitive for the left forearm across instances.

We remark that our model is able to accurately capture the various human body parts using superquadric surfaces. Another

interesting aspect of our model, which is also observed in [5], is related to the fact that our model uses the same primitive (highlighted with the same color) to represent the same actual human body part. For example, the head is typically captured using the primitive illustrated with red. For some poses these correspondences are lost. We speculate that this is because the network does not know whether the human is facing in front or behind.

4. Qualitative Results on ShapeNet

In this section, we provide additional qualitative results on various object types from the ShapeNet dataset [2]. We also demonstrate the ability of our model to capture fine details in more complicated objects such as *motorcycle-bikes* and *cars*. Due to their diverse shape vocabulary, superquadrics can accurately capture the structure of complex objects such as motorbikes and cars. We observe that our model successfully represents the wheels of all bikes using a flattened ellipsoid and the front fork using a pointy ellipsoid. Again, we note that our network consistently associates the same primitive with the same semantic part. For instance, for the motorcycles object category, the primitive colored in red is associated with the saddle, the primitive colored in green is associated with the front wheel etc. Fig. 6+7 demonstrate several predictions for both classes using superquadric surfaces as geometric primitives.

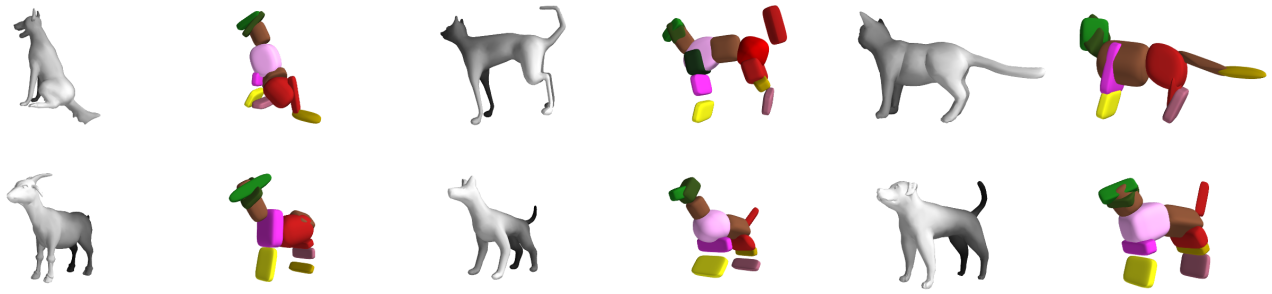


Figure 4: **Qualitative Results on *animals* from ShapeNet.** We visualize the predictions of our network on the *animals* class of the ShapeNet dataset. We remark the consistency across primitives and animal parts as well as the ability of our model to cpature details such as ears and tails of animals that could have not beeen captured using cuboidal primitives

Due to superquadrics' large shape vocabulary, our approach derives expressive scene abstractions that allow for differentiating between different types of vehicles both for motorcycles (scooter, racing bike, chopper etc.) (Fig. 6), cars (sedan, convertible, coupe, etc.) (Fig. 7), animals (dogs, cats) (Fig. 4) despite that our model leverages only up to 20 primitives per object. Note that for cars, wheels are not as easily recovered as for motorbikes due to the lack of supervision and since they are "geometrically occluded" by the body of the car.

Figure 5: **Qualitative Results on *chairs* from ShapeNet.** We visualize the predictions of our network on the *chairs* class of the ShapeNet dataset. We observe the consistency across corespondences between primitives and object parts as well as the ability of our model to capture the shape of rounded parts.

Fig. 4+5 depicts additional predictions on the *animal* and the *chair* object class of the ShapeNet dataset. We observe that for both categories our model consistently captures both the structure and the fine details of the depicted object. Note that chairs that have rounded legs are associated with flattened ellipsoids (Fig. 5), this would not have been possible only with cuboids.

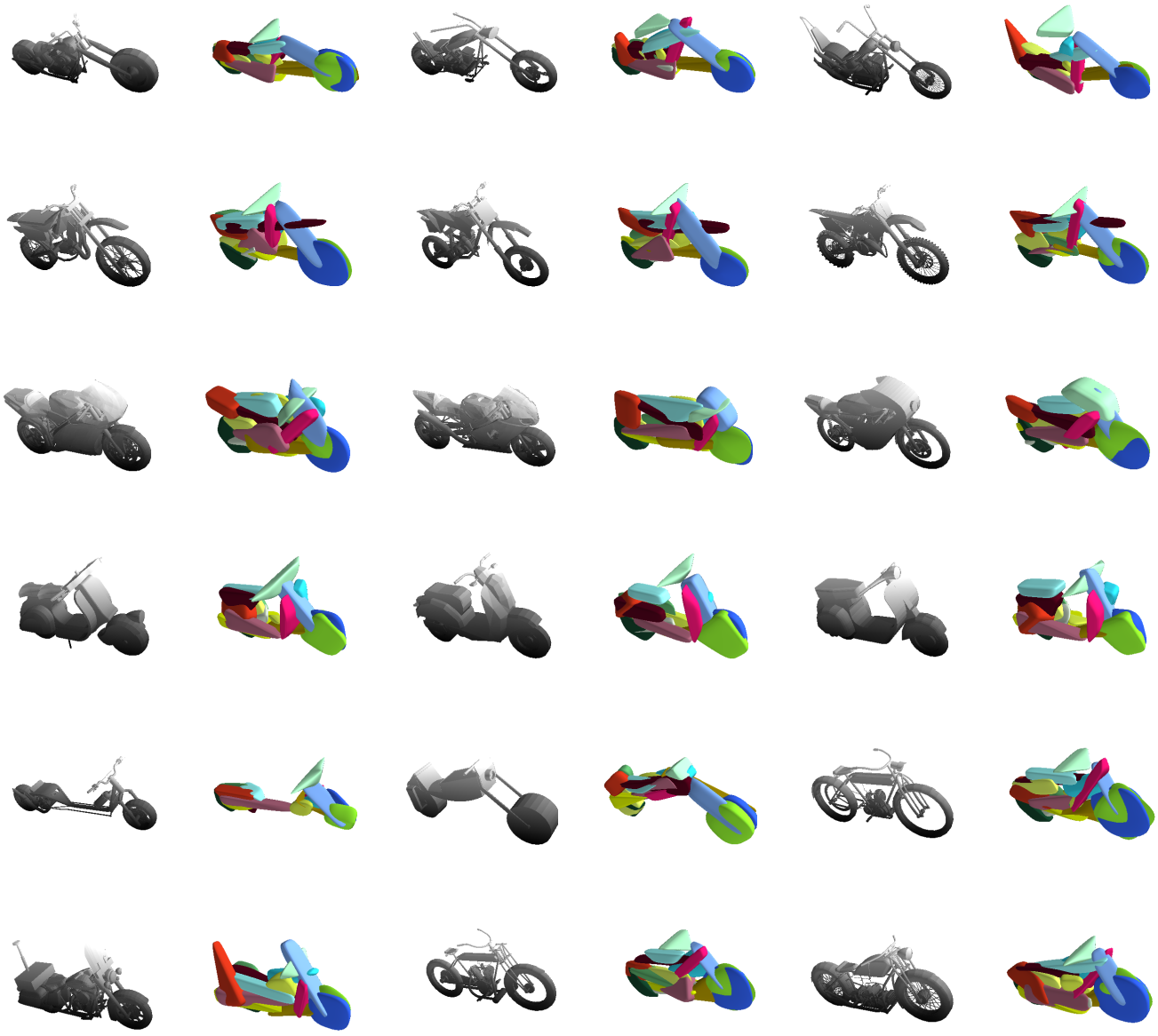


Figure 6: **Qualitative Results on *motorbikes* from ShapeNet.** Our network learns semantic mappings of various object parts of different objects within the same category. Our expressive shape abstractions allow for differentiating between different types of motorbikes (scooter, racing bike, chopper etc.), by sucessfully capturing the shape of various indicative parts such as the wheels or the front fork of the bike.

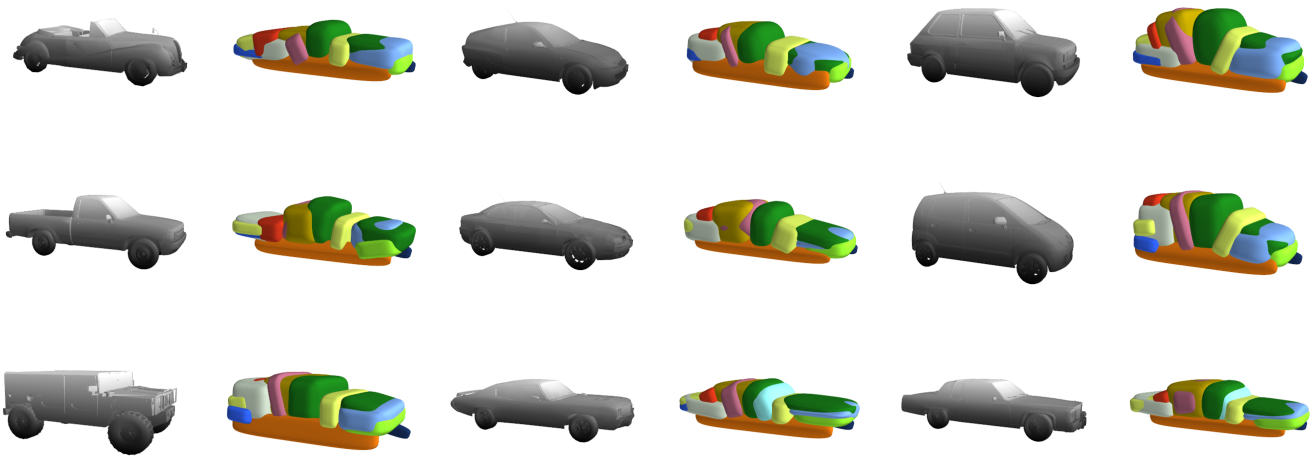


Figure 7: **Qualitative Results on *cars* from ShapeNet.** We visualize predictions for the object categories *car* from the ShapeNet dataset. Our expressive shape abstractions allow us to differentiate between different car types such as sedan, coupe etc. This would not have been possible with cuboidal primitives that cannnot model rounded surfaces

5. Network Architecture Details

In this section, we detail the network architecture used throughout our experimental evaluations. Our network comprises of two main parts, an encoder that learns a low-dimensional feature representation for the input and five regressors that predict the parameters of the superquadrics (size α, shape ϵ, translations $\mathbf{t}$, rotations $\mathbf{q}$ and γ probabilities of existence). As we already explained in our main submission, the encoder architecture is chosen based on the input type (image, voxelized input etc.). In our experiments, we consider a binary occupancy grid as an input and the sequence of layers comprising both the encoder and the regressors are depicted in Figure 8.

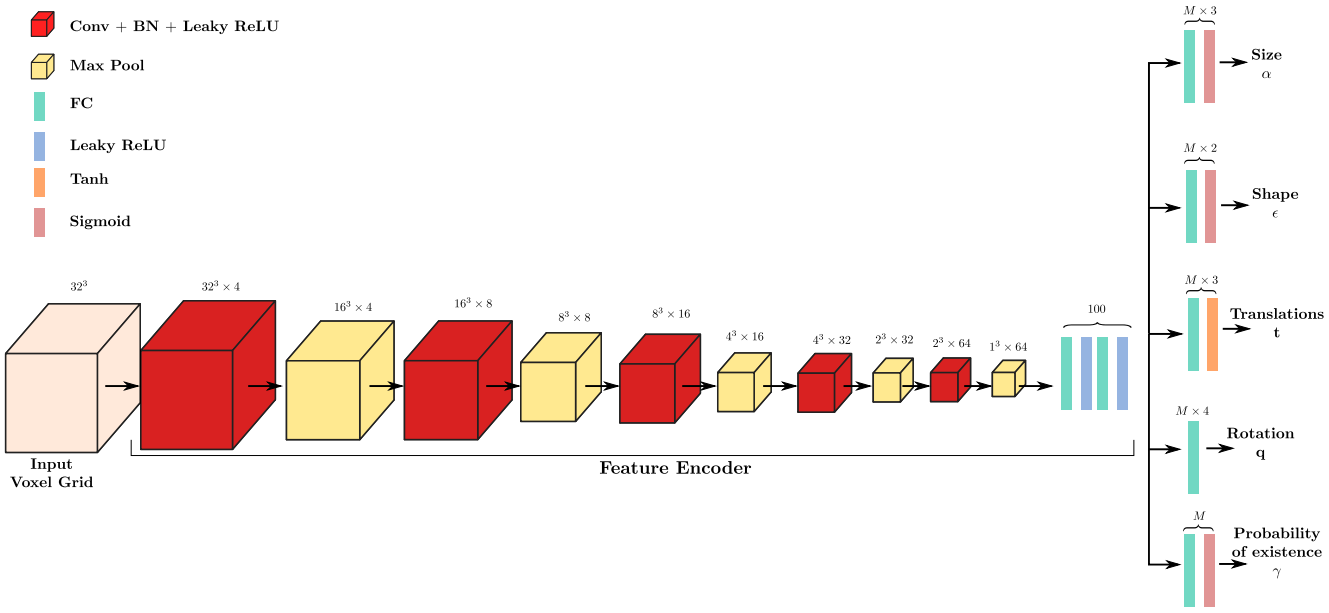


Figure 8: **Volume-based network architecture.** We visualize the layers that comprise our network architecture. Cubes denote operations that are conducted on 3-dimensional volumes, while rectangles correspond to K-dimensional features. The number above each shape (cube or rectangle) corresponds to the dimensionality of that layer. For instance, $16^3 \times 4$ denotes a feature map of size 16^3 and 4 channels. Following, our notation, M corresponds to the maximum number of primitives predicted.

Note that, for the image-based experiment of section 6 in the supplementary, where we consider an image as an input to our model, we replace the encoder architecture in Fig. 8 with a ResNet18 [3].

5.1. Parsimony Loss Details

We would also like to briefly provide some additional details for our parsimony loss. For completeness, we restate the parsimony loss of Equation 12 in our main submission,

$$\mathcal{L}_\gamma(\mathbf{P}) = \max\left(\alpha - \alpha \sum_{m=1}^{M} \gamma_m, 0\right) + \beta\sqrt{\sum_{m=1}^{M} \gamma_m} \tag{9}$$

Note that the $\sum_{m=1}^{M} \gamma_m$ corresponds to the expected number of primitives in the predicted parsing. As already mentioned, in our main submission, our model suffers from the trivial solution $\mathcal{L}_D(\mathbf{P}, \mathbf{X}) = 0$ which is attained for $\gamma_1 = \cdots = \gamma_m = 0$. To avoid this solution, we introduce the first term of Eq. 9 that penalizes the prediction when the expected number of primitives is less than 1. The second term penalizes the prediction when the expected number of primitives is large. Note that the maximum value of the second term is $\beta\sqrt{M}$, while the maximum value of the first term is α. Therefore, in order to allow the model to use more than one primitive, we set β to a value smaller than α. Typically $\alpha = 1.0$ and $\beta = 10^{-3}$.

6. Shape Abstraction from a Single RGB Image

In this section, we use the proposed reconstruction loss of Eq. 3, in the main submission, to extract shape primitives from RGB images instead of occupancy grids. Towards this goal, we render the ShapeNet models to images, and train an image-based network to minimize the same reconstruction loss also used for our volume-based architecture.

More specifically, we replace the encoder architecture, described in Section 3.4 in our main submission, with the ResNet18 architecture [3], without the last fully connected layer. The extracted features are subsequently passed to five independent heads that regress translation $\mathbf{t}$, rotation $\mathbf{q}$, size α, shape ϵ and probability of existence γ for each primitive. During training, we uniformly sample 1000 points, from the surface of the target object, as well as 200 points from the surface of every superquadric. For optimization, we use ADAM [4] with a learning rate of 0.001 and a batch size of 32 for 40k iterations. We observe that our model accurately captures shape primitives even from a single RGB image as input.

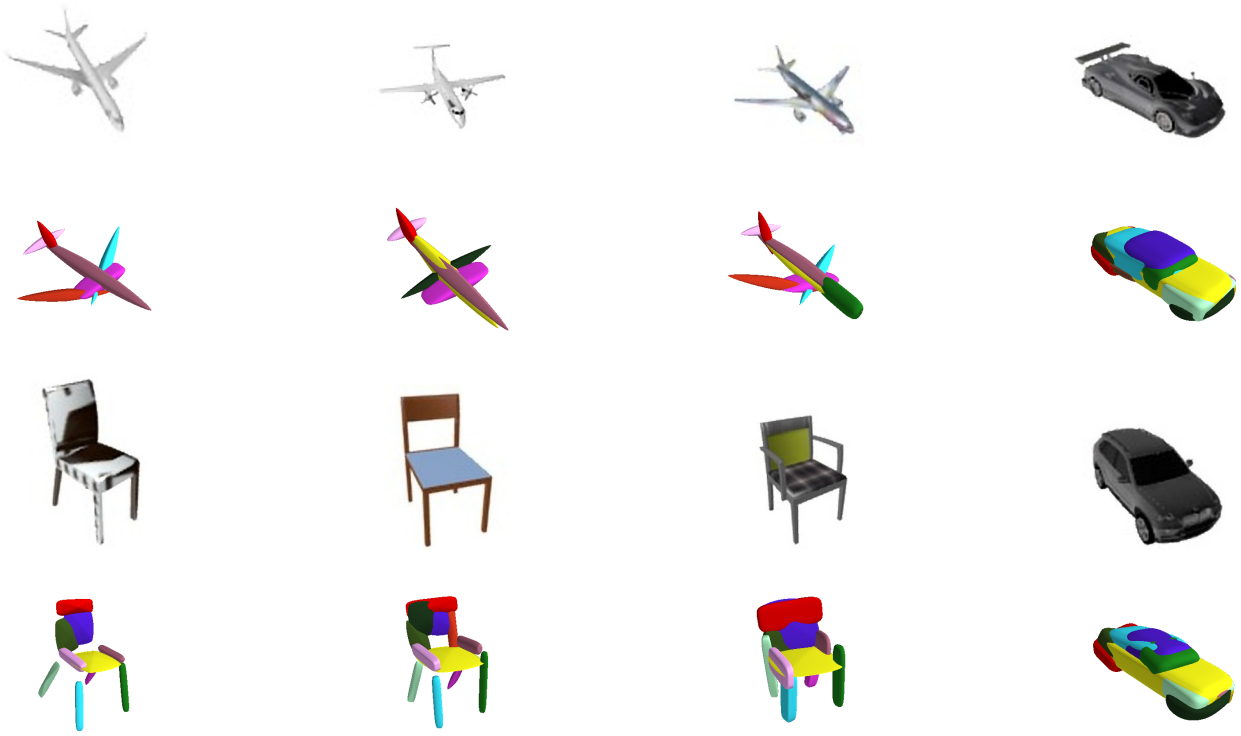


Figure 9: **Shape Abstraction from a Single RGB Image.** We visualize predictions for various ShapeNet object categories using a single RGB image as input to our model.

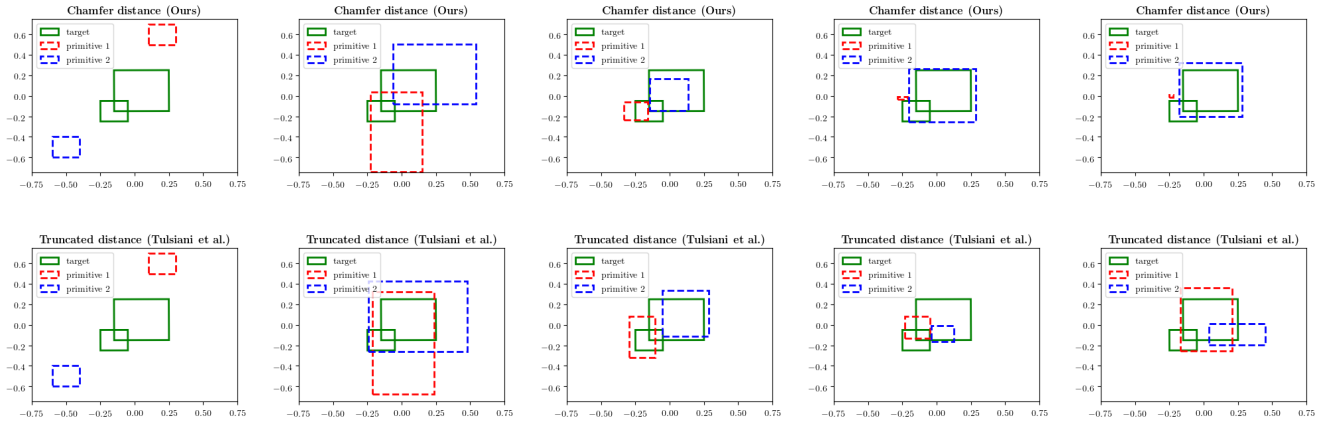
7. Quantitative Analysis

In this section, we provide additional details regarding the quantitative comparison of Table 1 in our main paper. For evaluation, we report two metrics the mean *Chamfer distance* and the mean *Volumetric IoU*. Volumetric IoU is defined as the quotient of the volume of the two meshes' intersection and the volume of their union. We obtain unbiased estimates of the volume of the intersection and the union by randomly sampling points from the bounding volume and determining if the points lie inside our outside the ground truth / predicted mesh. The computation of the Chamfer distance is discussed in detail in our main submission throughout Section 3. Regarding the comparison in Table 1 of our main submission, we want to mention that cuboids are a special case of superquadrics, thus fitting objects with cuboids is expected to lead to worse results compared to superquadrics.

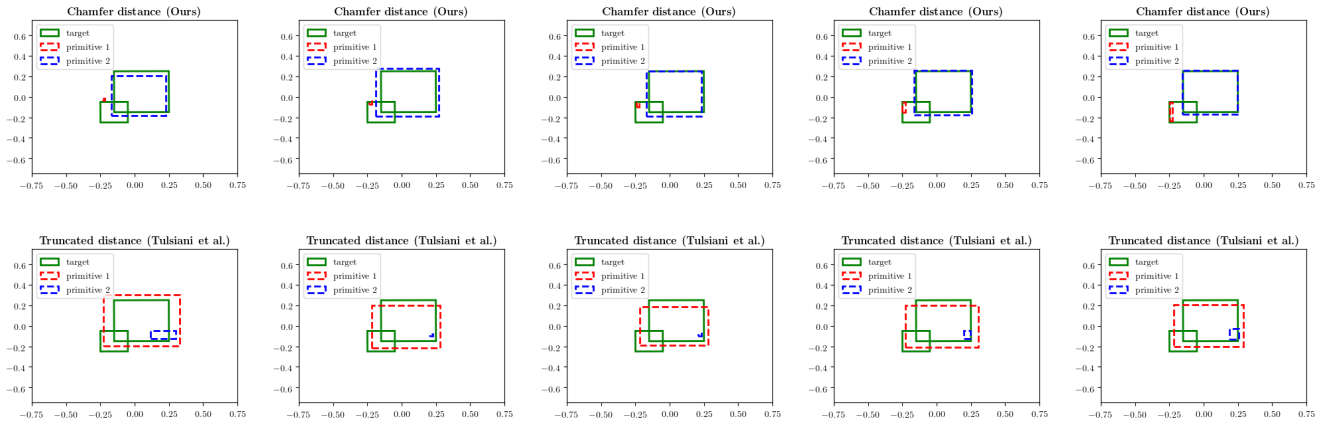
8. Empirical Analysis of Reconstruction Loss

In this section, we provide empirical evidence regarding our claim that our Chamfer-based reconstruction loss leads to more stable training compared to the truncated bi-directional loss of Tulsiani et al. [5]. Towards this goal, we directly opti-

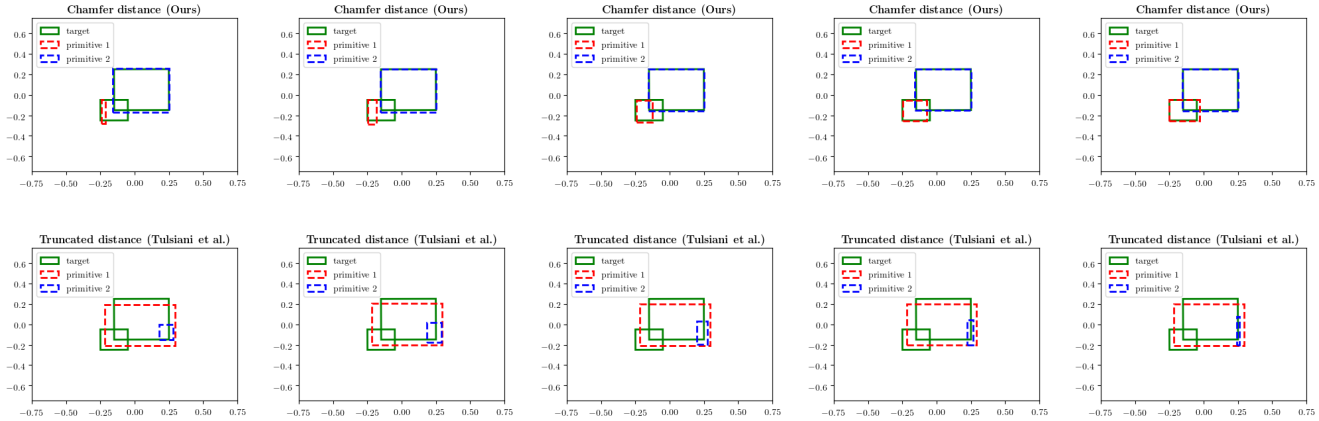
mize/train for the primitive parameters, i.e., not optimizing the weights of a neural network but directly fitting the primitives. We perform this experiment on a 2D toy example and compare the results when using the proposed loss to the results using the truncated distance formulation in [5]. We visualize the evolution of parameters for both optimization objectives as training progresses. We observe that the truncated loss proposed in [5] is more likely to converge to local minima (e.g. figures 11k-11o), while our loss consistently avoids them.



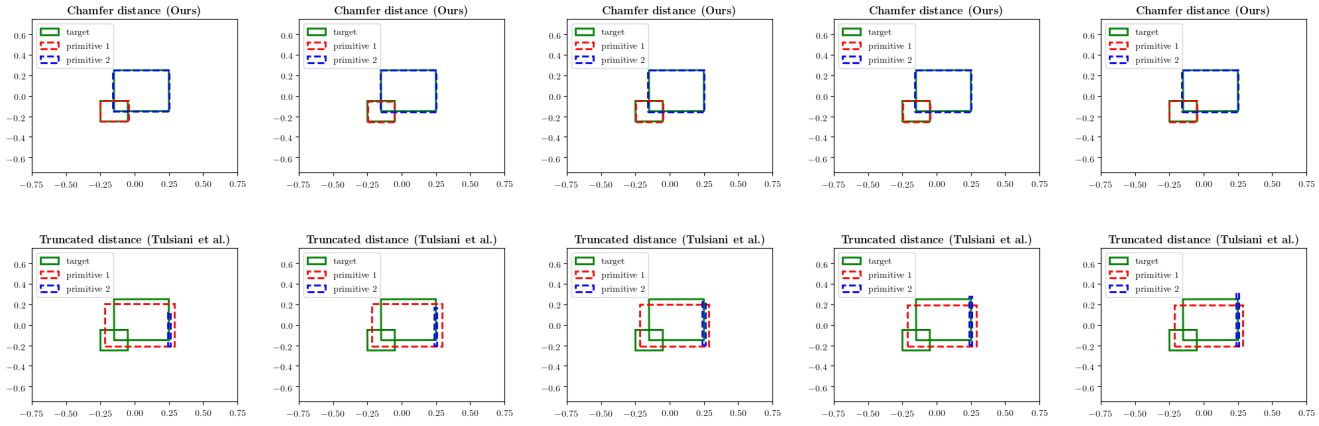
(a) **Iteration** 0 (b) **Iteration** 10 (c) **Iteration** 20 (d) **Iteration** 30 (e) **Iteration** 40



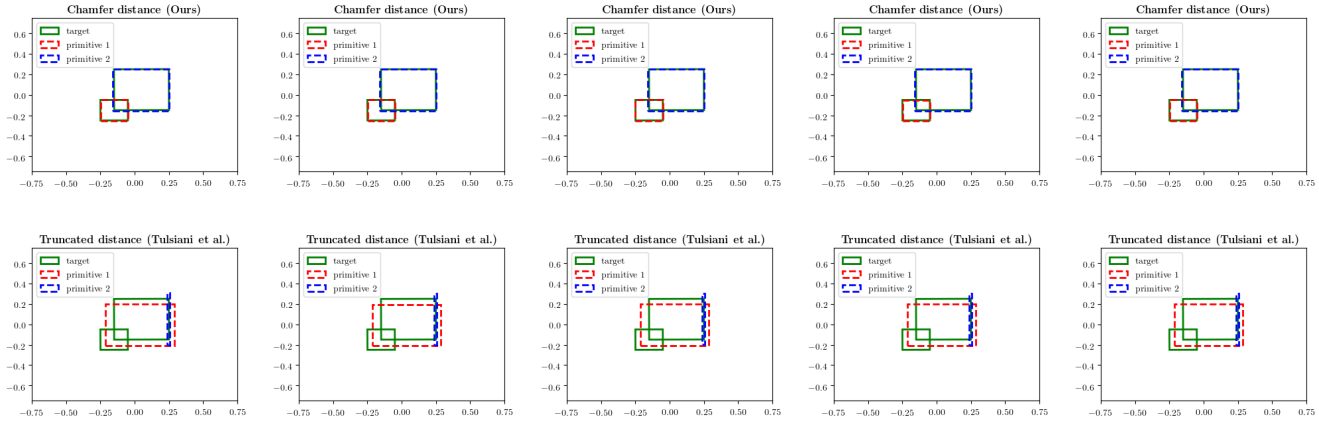
(f) **Iteration** 50 (g) **Iteration** 60 (h) **Iteration** 70 (i) **Iteration** 80 (j) **Iteration** 90



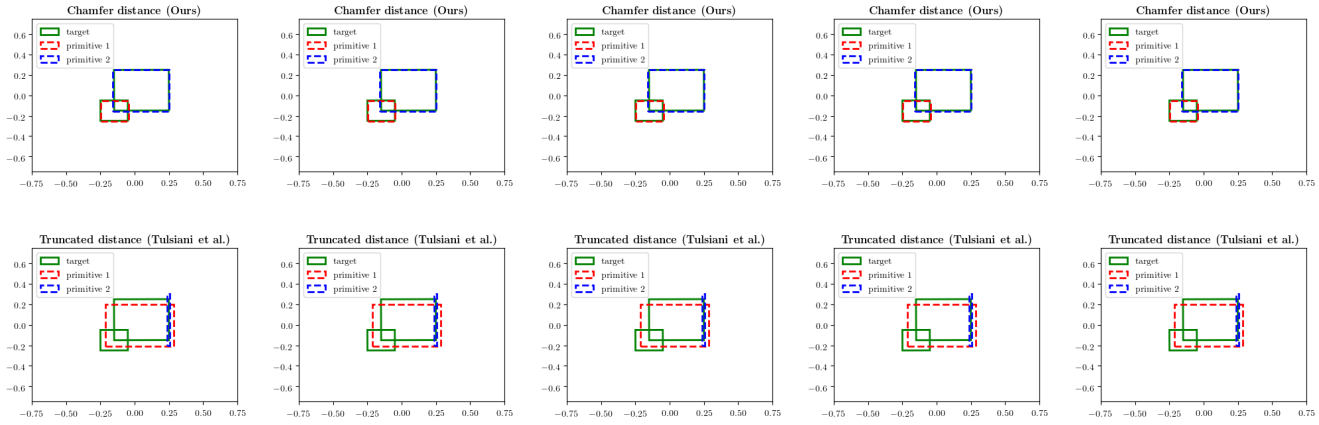
(k) **Iteration** 100 (l) **Iteration** 110 (m) **Iteration** 120 (n) **Iteration** 130 (o) **Iteration** 140



(a) **Iteration** 150 (b) **Iteration** 160 (c) **Iteration** 170 (d) **Iteration** 180 (e) **Iteration** 190



(f) **Iteration** 200 (g) **Iteration** 210 (h) **Iteration** 220 (i) **Iteration** 230 (j) **Iteration** 240



(k) **Iteration** 250 (l) **Iteration** 260 (m) **Iteration** 270 (n) **Iteration** 280 (o) **Iteration** 290

Figure 11: **Empirical Analysis of Reconstruction Loss**. We illustrate the evolution of two cuboid abstractions using our reconstruction loss with Chamfer distance and the truncated bi-directional loss of Tulsiani et al. [5].

References

[1] Alan H Barr. Superquadrics and angle-preserving transformations. *IEEE Computer Graphics and Applications (CGA)*, 1981. 1, 2

[2] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. *arXiv.org*, 1512.03012, 2015. 1, 5

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016. 8, 9

[4] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. of the International Conf. on Learning Representations (ICLR)*, 2015. 9

[5] Shubham Tulsiani, Hao Su, Leonidas J. Guibas, Alexei A. Efros, and Jitendra Malik. Learning shape abstractions by assembling volumetric primitives. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 5, 9, 10, 11

[6] Gül Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J. Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1