Supplementary Material for Learning Unsupervised Hierarchical Part Decomposition of 3D Objects from a Single RGB Image

Despoina Paschalidou[1,3,5] Luc van Gool[3,4,5] Andreas Geiger[1,2,5]
[1]Max Planck Institute for Intelligent Systems Tübingen
[2]University of Tübingen [3]Computer Vision Lab, ETH Zürich [4]KU Leuven
[5]Max Planck ETH Center for Learning Systems
{firstname.lastname}@tue.mpg.de vangool@vision.ee.ethz.ch

Abstract

*In this **supplementary document**, we first present examples of our occupancy function. In addition, we present a detailed overview of our network architecture and the training procedure. We then discuss how various components influence the performance of our model on the single-view 3D reconstruction task. Finally, we provide additional experimental results on more categories from the ShapeNet dataset [2] and on the D-FAUST dataset [1] together with the corresponding hierarchical structures. The **supplementary video** shows 3D animations of the predicted structural hierarchy for various objects from the ShapeNet dataset as well as humans from the D-FAUST.*

1. Occupancy Function

In this section, we provide illustrations of the occupancy function g for different primitive parameters and for different sharpness values. For any point $\mathbf{x} \in \mathbb{R}^3$, we can determine whether it lies inside or outside a superquadric using its implicit surface function which is commonly referred to as the *inside-outside function*:

$$f(\mathbf{x};\lambda) = \left(\left(\frac{x}{\alpha_1} \right)^{\frac{2}{\epsilon_2}} + \left(\frac{y}{\alpha_2} \right)^{\frac{2}{\epsilon_2}} \right)^{\frac{\epsilon_2}{\epsilon_1}} + \left(\frac{z}{\alpha_3} \right)^{\frac{2}{\epsilon_1}} \tag{1}$$

where $\alpha = [\alpha_1, \alpha_2, \alpha_3]$ determine the size and $\epsilon = [\epsilon_1, \epsilon_2]$ determine the shape of the superquadric. If $f(\mathbf{x};\lambda) = 1.0$, the given point $\mathbf{x}$ lies on the surface of the superquadric, if $f(\mathbf{x};\lambda) < 1.0$ the corresponding point lies inside and if $f(\mathbf{x};\lambda) > 1.0$

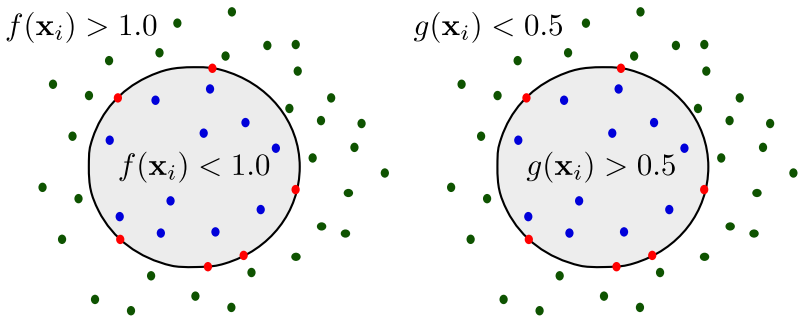
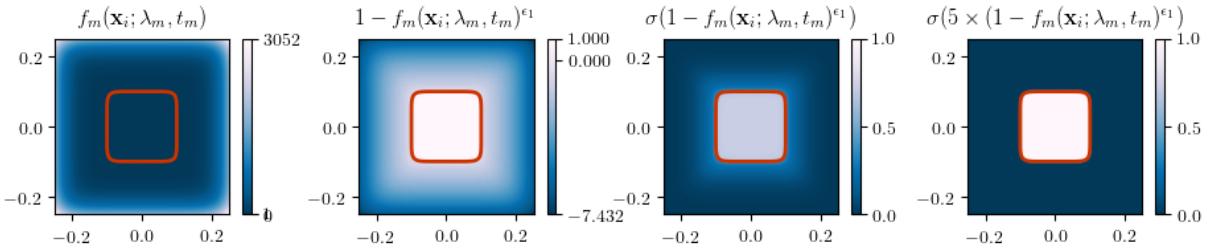


Figure 1: **Implicit surface function of superquadrics.** We visualize the 2D slice of $f(\mathbf{x}_i)$ and $g(\mathbf{x}_i)$ for a superquadric with $\alpha_1 = \alpha_2 = \alpha_3 = \epsilon_1 = \epsilon_2 = 1$.

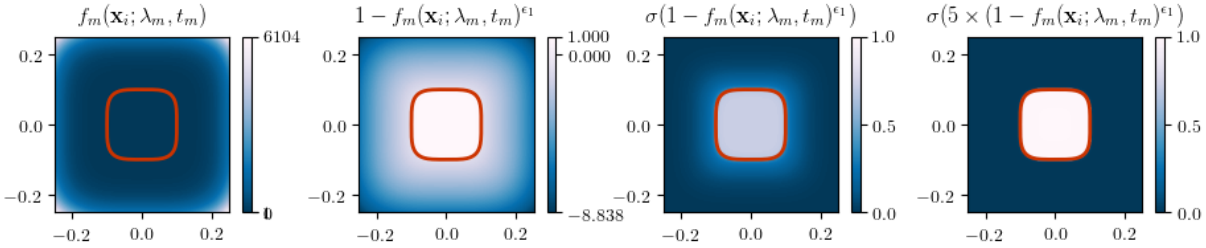
the point lies outside the superquadric. To account for numerical instabilities that arise from the exponentiations in (1), instead of directly using $f(\mathbf{x};\lambda)$, we follow [8] and use $f(\mathbf{x};\lambda)^{\epsilon_1}$. In addition, we also convert the inside-outside function to an *occupancy function*, $g : \mathbb{R}^3 \rightarrow [0,1]$:

$$g(\mathbf{x};\lambda) = \sigma\left(s\left(1 - f(\mathbf{x};\lambda)^{\epsilon_1}\right)\right) \tag{2}$$

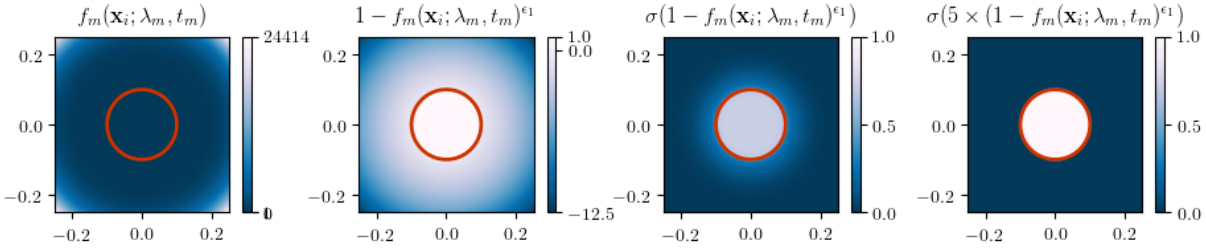
that results in per-point predictions suitable for the classification problem we want to solve. $\sigma(\cdot)$ is the sigmoid function and s controls the sharpness of the transition of the occupancy function. As a result, if $g(\mathbf{x};\lambda) < 0.5$ the corresponding point lies outside and if $g(\mathbf{x};\lambda) > 0.5$ the point lies inside the superquadric. Fig. 1 visualizes the range of the implicit surface function of superquadrics of (1) and (2). Fig. 2+3+4 visualize the implicit surface function for different values of ϵ_1 and ϵ_2 and different values of sharpness s. We observe that without applying the sigmoid to (1) the range of values of (1) varies significantly for different primitive parameters.



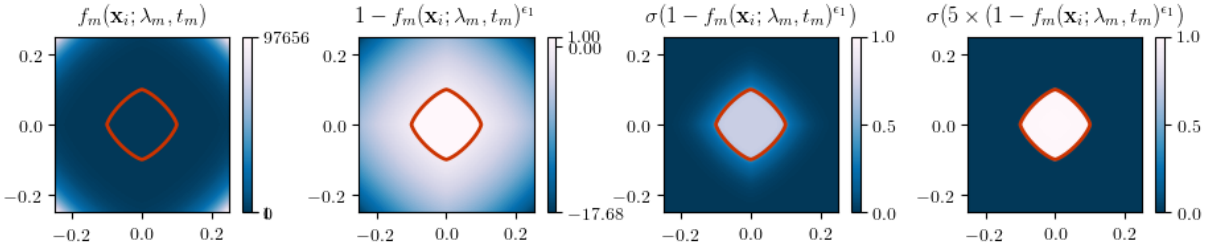
(a) $\epsilon_1 = 0.25$, and $\epsilon_2 = 0.25$



(b) $\epsilon_1 = 0.25$, and $\epsilon_2 = 0.5$

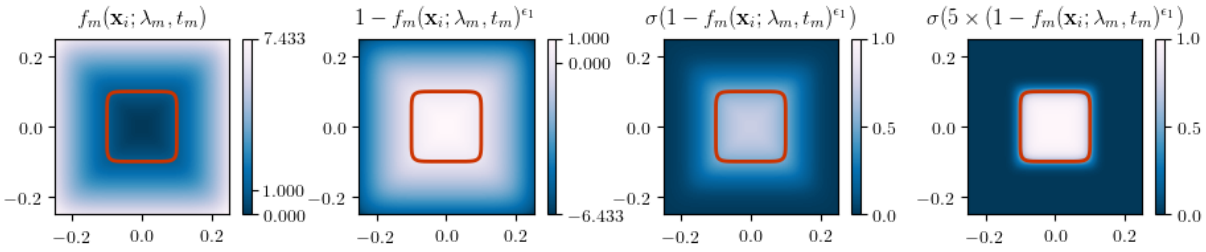


(c) $\epsilon_1 = 0.25$, and $\epsilon_2 = 1.0$

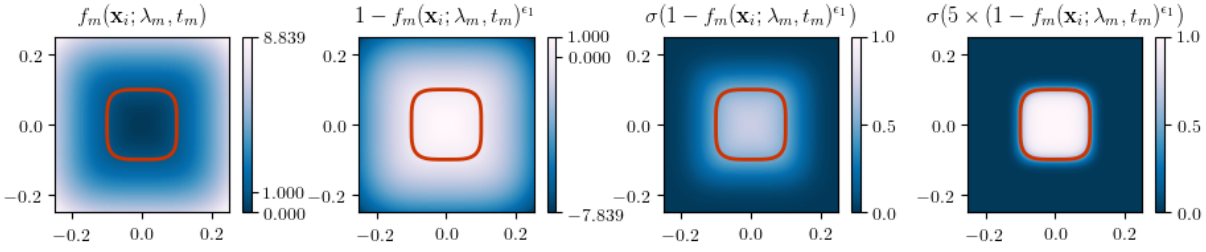


(d) $\epsilon_1 = 0.25$, and $\epsilon_2 = 1.5$

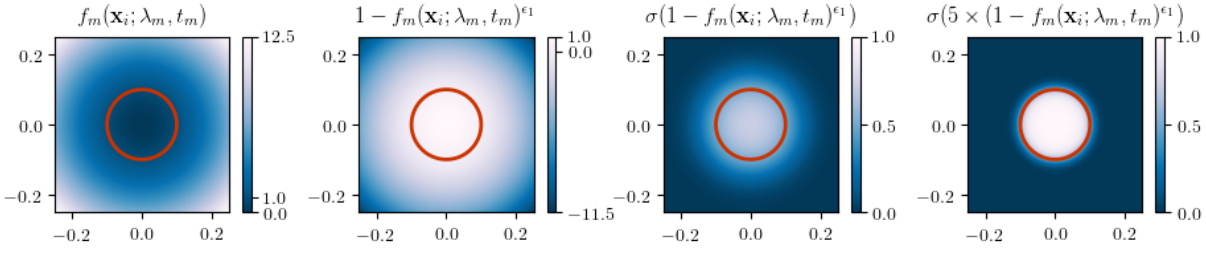
Figure 2: **Implicit surface function** We visualize the implicit surface function for different primitive parameters and for different sharpness values. The surface boundary is drawn with red.



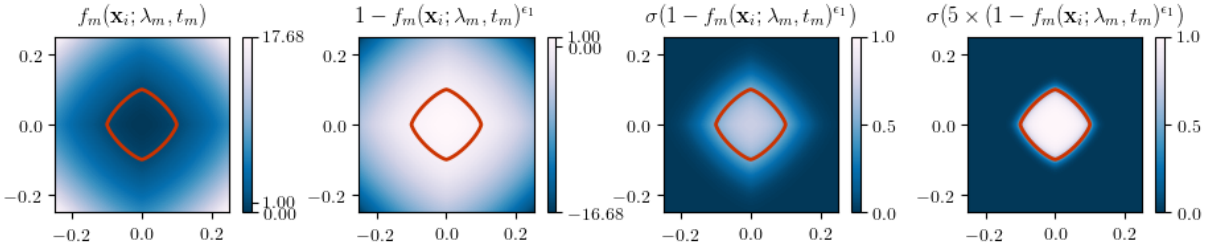
(a) $\epsilon_1 = 1.0$, and $\epsilon_2 = 0.25$



(b) $\epsilon_1 = 1.0$, and $\epsilon_2 = 0.5$

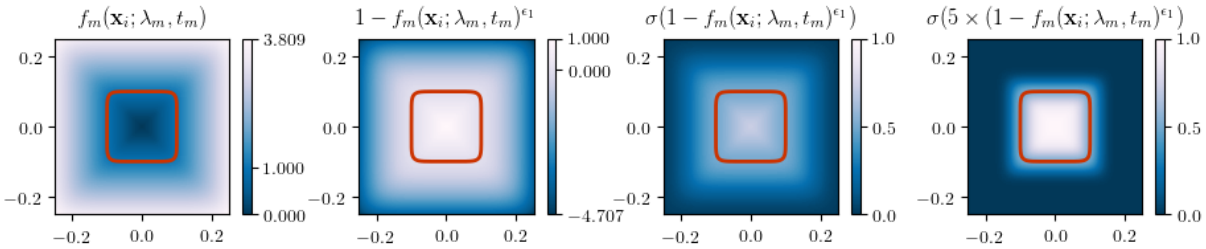


(c) $\epsilon_1 = 1.0$, and $\epsilon_2 = 1.0$

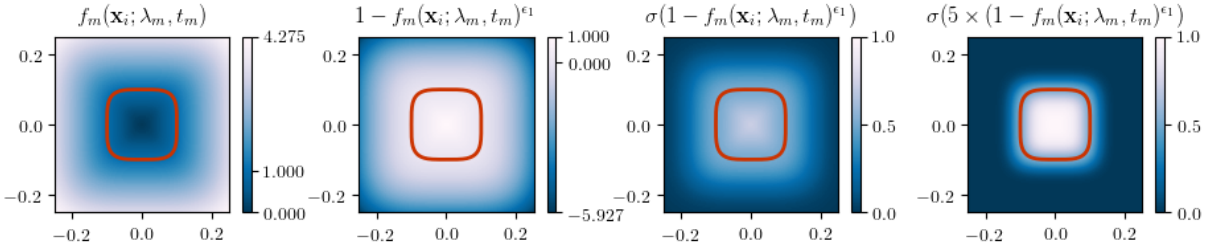


(d) $\epsilon_1 = 1.0$, and $\epsilon_2 = 1.5$

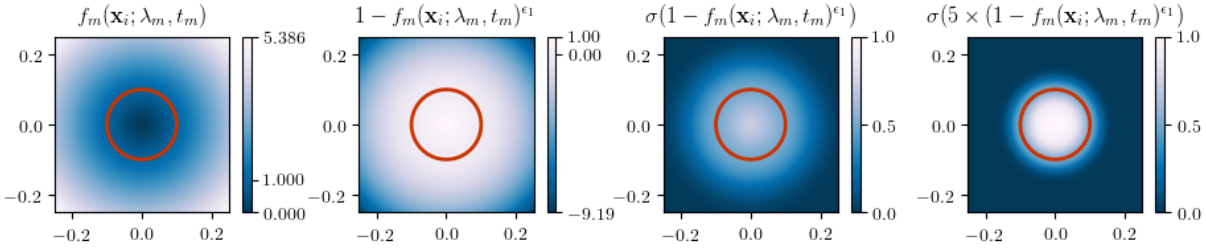
Figure 3: **Implicit surface function** We visualize the implicit surface function for different primitive parameters and for different sharpness values. The surface boundary is drawn with red.



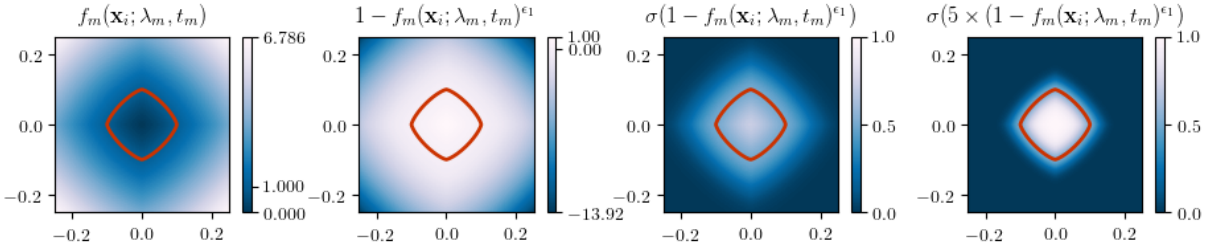
(a) $\epsilon_1 = 1.5$, and $\epsilon_2 = 0.25$



(b) $\epsilon_1 = 1.5$, and $\epsilon_2 = 0.5$



(c) $\epsilon_1 = 1.5$, and $\epsilon_2 = 1.0$



(d) $\epsilon_1 = 1.5$, and $\epsilon_2 = 1.5$

Figure 4: **Implicit surface function** We visualize the implicit surface function for different primitive parameters and for different sharpness values. The surface boundary is drawn with red.

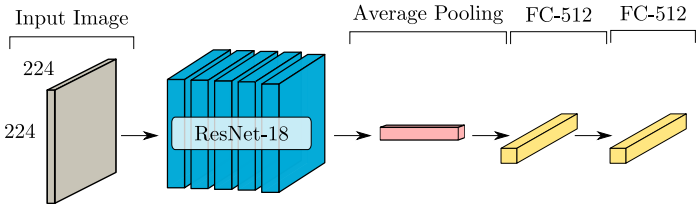
2. Implementation Details

In this section, we provide a detailed description of our network architecture. We then describe our sampling strategy and provide details on the metrics we use both for training and testing. Finally, we show how various components influence the performance of our model on the single-view 3D reconstruction task.

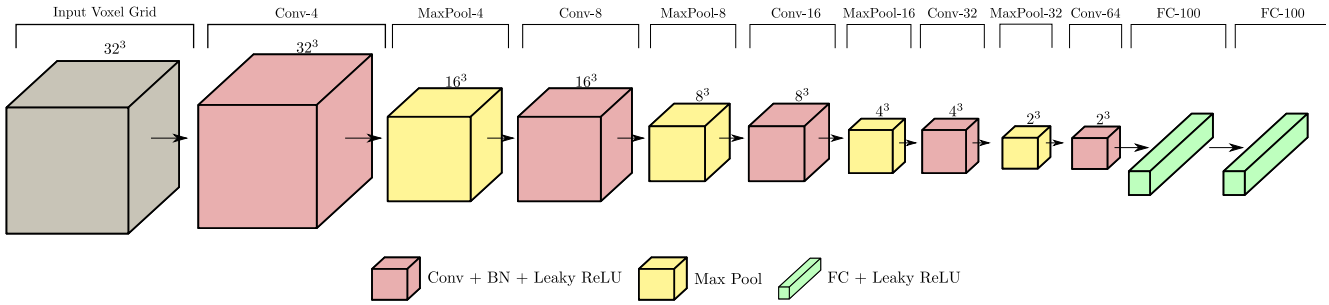
2.1. Network Architecture

Here we describe the architecture of each individual component of our model, shown in Figure 3 of our main submission.

Feature Encoder: The feature encoder depends on the type of the input, namely whether it is an image or a binary occupancy grid. For the single view 3D reconstruction task, we use a ResNet-18 architecture [7] (Fig. 5a), which was pretrained on the ImageNet dataset [5]. From the original design, we ignore the final fully connected layer keeping only the feature vector of length $F = 512$ after average pooling. For the volumetric 3D reconstruction task, where the input is a binary occupancy grid, we use the feature encoder proposed in [11](Fig. 5b). Note that the feature encoder is used as a generic feature extractor from the input representation.



(a) Single-view 3D Reconstruction



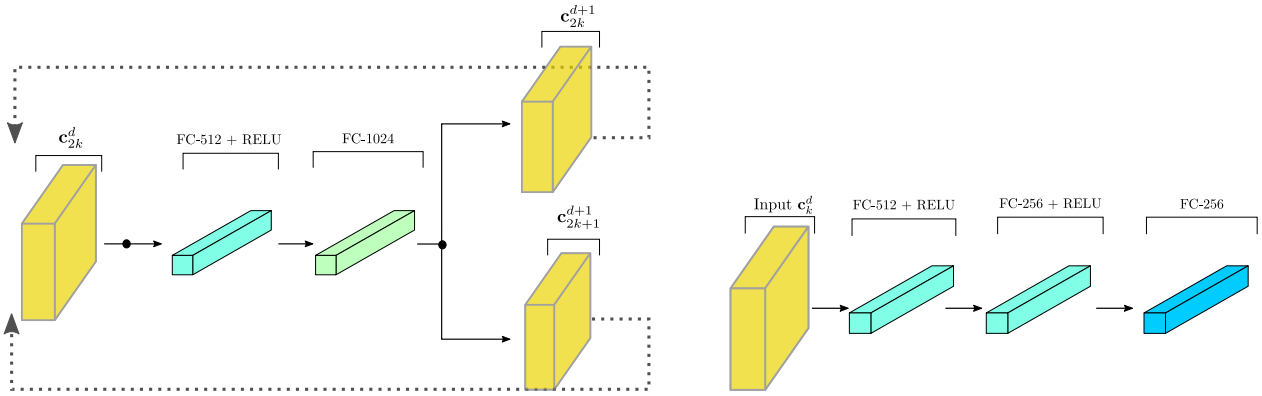
(b) Volumetric 3D Reconstruction

Figure 5: **Feature Encoder Architectures.** Depending on the type of the input, we employ two different network architectures. (a) For the single view 3D reconstruction task we use a ResNet-18 architecture [7] (b) For a binary occupancy grid as an input, we leverage the network architecture of [11].

Partition Network: The partition network implements a function $p_\theta : \mathbb{R}^F \rightarrow \mathbb{R}^{2F}$ that recursively partitions the feature representation $\mathbf{c}_k^d$ of node p_k^d into two feature representations, one for each child $\{p_{2k}^{d+1}, p_{2k+1}^{d+1}\}$. The partition network (Fig. 6a) comprises two fully connected layers followed by RELU non linearity.

Structure Network: The structure network maps each feature representation $\mathbf{c}_k^d$ to $\mathbf{h}_k^d$ a spatial location in $\mathbb{R}^3$. The structure network (Fig. 6b) consists of two fully connected layers followed by RELU non linearity.

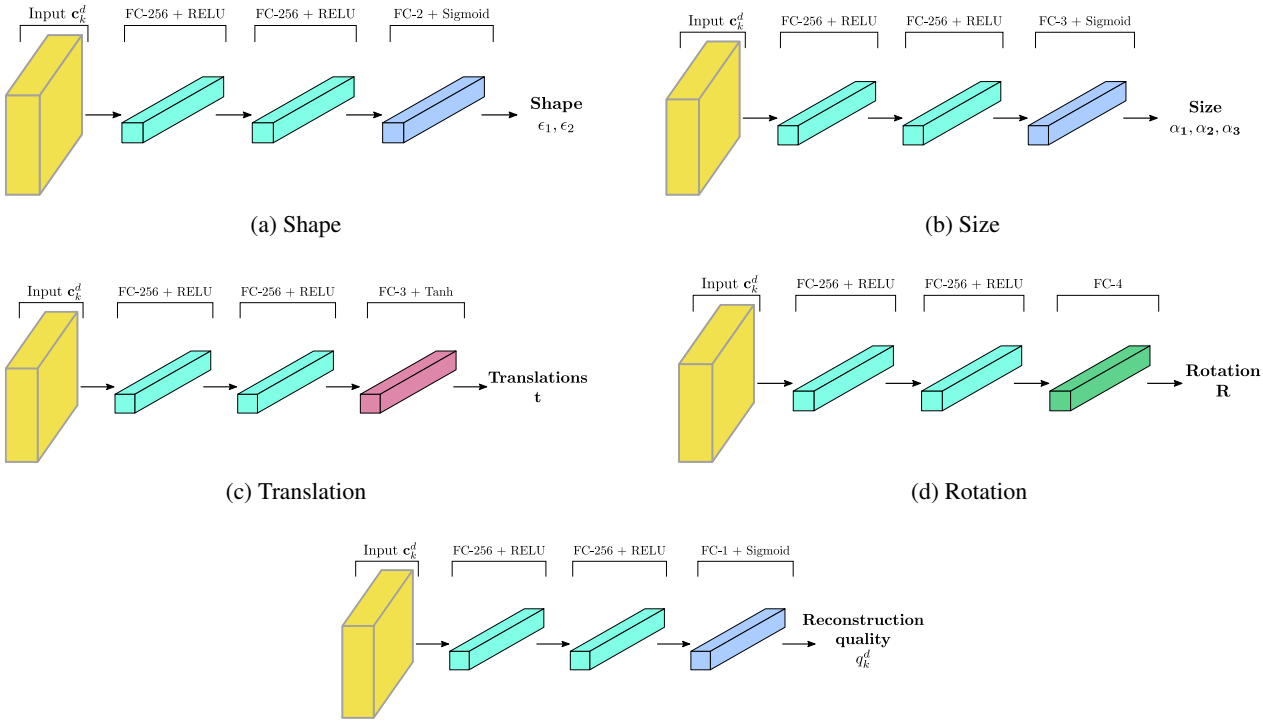
Geometry Network: The geometry network learns a function $r_\theta : \mathbb{R}^F \rightarrow \mathbb{R}^K \times [0, 1]$ that maps the feature representation $\mathbf{c}_k^d$ to its corresponding primitive parametrization λ_k^d and the reconstruction quality prediction q_k^d. In particular, the geometry network consists of five regressors that predict the parameters of the superquadrics (size α, shape ϵ and pose as translation



(a) Partition Network

(b) Structure Network

Figure 6: **Network Architecture Overview.** The *partition network* (6a) is simply one hidden layer fully connected network with RELU non linearity. The gray dotted lines indicate the recursive partition of the feature representation. Similarly, the *structure network* (6b) consists of two fully connected layers followed by RELU non linearity.



(a) Shape

(b) Size

(c) Translation

(d) Rotation

(e) Reconstruction quality

Figure 7: **Geometry Network.** We detail the specifics of each regressor for predicting the primitive parameters λ_k^d and the reconstruction quality q_k^d.

$\mathbf{t}$ and rotation $\mathbf{R}$) in addition to the reconstruction quality q_k^d. Fig. 7 presents the details of the implementation of each regressor.

2.2. Training

In all our experiments, we use the Adam optimizer [9] with learning rate 0.0001 and no weight decay. For other hyper-parameters of Adam we use the PyTorch defaults. We train all models with a batch size of 32 for 40k iterations. We do not

perform any additional data augmentation. We weigh the loss terms of Eq. 9 in our main submission with 0.1, 0.01, 0.01 and 0.1 respectively, in order to enforce that during the first stages of training the network will focus primarily on learning the hierarchical decomposition of the 3D shape ($\mathcal{L}_s + \mathcal{L}_p$). In this way, after the part decomposition is learned, the network also focuses on the part geometries ($\mathcal{L}_r$). We also experimented with a two-stage optimization scheme, where we first learn the hierarchical part decomposition and then learn the hierarchical representation, but we observed that this made learning harder.

2.3. Sampling Strategy

Sampling a point inside the target mesh has a probability proportional to the volume of the mesh. This yields bad reconstructions for thin parts of the object, such as legs of chairs and wings of aeroplanes. In addition, biasing the sampling towards the points inside the target mesh, results in worse reconstructions as also noted in [10]. To address the first issue (properly reconstructing thin parts), we use an unbalanced sampling distribution that, in expectation, results in sampling an equal number of points inside and outside the target mesh. To counter the second (biased sampling), we construct an unbiased estimator of the loss by weighing the per-point loss inversely proportionally to its sampling probability. We refer to our sampling strategy as *unbiased importance sampling*. Note that throughout all our experiments, we sample 10k points in the bounding box of the target mesh using our sampling strategy.

2.4. Metrics

We evaluate our model and our baselines using the volumetric Intersection over Union (IoU) and the Chamfer-L_1 distance. Note that as our method does not predict a single mesh, we sample points from each primitive proportionally to its area, such that the total number of sampled points from all primitives is equal to 100k. For a fair comparison, we do the same for [11,12]. Below, we discuss in detail the computation of the volumetric IoU and the Chamfer-L_1.

Volumetric IoU is defined as the quotient of the volume of the intersection of the target $S_{target}$and the predicted S_{pred} mesh and the volume of their union. We obtain unbiased estimates of the volume of the intersection and the union by randomly sampling 100k points from the bounding volume and determining if the points lie inside or outside the target / predicted mesh,

$$\text{IoU}(S_{pred}, S_{target}) = \frac{\mid V(S_{pred} \cap S_{target}) \mid}{\mid V(S_{pred} \cup S_{target}) \mid} \tag{3}$$

where $V(.)$ is a function that computes the volume of a mesh.

We obtain an unbiased estimator of the Chamfer-L_1 distance by sampling 100k points on the surface of the target S_{target} and the predicted S_{pred} mesh. We denote $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^N$ the set of points sampled on the surface of the target mesh and $\mathcal{Y} = \{\mathbf{y}_i\}_{i=1}^N$ the set of points sampled on the surface of the predicted mesh. We compute the Chamfer-L_1 as follows:

$$D_{\text{chamfer}}(\mathcal{X}, \mathcal{Y}) = \frac{1}{N} \sum_{\mathbf{x}_i \in \mathcal{X}} \min_{\mathbf{y}_j \in \cup \mathcal{Y}} \|\mathbf{x}_i - \mathbf{y}_j\| + \frac{1}{N} \sum_{\mathbf{y}_i \in \cup \mathcal{Y}} \min_{\mathbf{x}_j \in \mathcal{X}} \|\mathbf{y}_i - \mathbf{x}_j\| \tag{4}$$

The first term of (4) measures the *completeness* of the predicted shape, namely how far is on average the closest predicted point from a ground-truth point. The second term measures the *accuracy* of the predicted shape, namely how far on average is the closest ground-truth point from a predicted point.

To ensure a fair comparison with our baselines, we use the evaluation code of [10] for the estimation of both the Volumetric IoU and the Chamfer-L_1.

2.5. Empirical Analysis of Loss Formulation

In this section, we investigate the impact of how various components of our model affect the performance on the single-image 3D reconstruction task.

2.5.1 Impact of Sampling Strategy

We first discuss how the sampling strategy affects the performance of our model. Towards this goal, we evaluate our model on the single-view 3D reconstruction task using three different sampling strategies: (a) uniform sampling in the bounding box that contains the target object (b) biased sampling (namely sampling an equal number of points inside and outside the target mesh without reweighing) and (c) unbiased importance sampling as described in Section 2.3. All models are trained on

	IoU	Chamfer-L_1
Uniform	0.383	0.073
Biased	0.351	0.041
Importance	**0.491**	0.073

(a) Influence of sampling strategy

	IoU	Chamfer-L_1
Importance 2k	0.370	0.074
Importance 5k	0.380	0.076
Importance 10k	**0.491**	**0.073**

(b) Influence of number of sampled points.

Table 1: **Sampling Strategy.** We evaluate the performance of our model while varying the sampling scheme and the number of the sampled points inside the bounding box of the target mesh. We report the volumetric IoU (higher is better) and the Chamfer distance (lower is better) on the test set of the "chair category".

the "chair" object category of ShapeNet using the same network architecture, the same number of sampled points (N =10k) and the same maximum number of primitives ($D = 16$). The quantitative results on the test set of the "chair" category are shown in Table 1. We observe that the proposed importance sampling strategy achieves the best results in terms of IoU.

Furthermore, we also examine the impact of the number of sampled points on the performance of our model. In particular, we train our model on the "chair" category while varying the number of sampled points inside the bounding box that contains the target mesh. As expected, increasing the number of sampled points results in an improvement in reconstruction quality. We empirically found that sampling 10k points results in the best compromise between training time and reconstruction performance.

2.5.2 Impact of Proximity loss

In this section, we explain empirically the vanishing gradient problem that emerges from the use of the sigmoid in the occupancy function of (2). To this end, we train two variants of our model, one with and without the proximity loss of Eq. 15, in our main submission. For this experiment, we train both variants on D-FAUST for the single image 3D reconstruction task. Both models are trained for a maximum number of 32 primitives and $s = 10$ and for the same number of iterations.

(a) **Input** (b) **without** (c) **Ours** (d) **Input** (e) **without** (f) **Ours**

Figure 8: **Vanishing gradients.** We visualize the predictions of two two models, one trained with (**Ours**) and one **without** the proximity loss term. On the left, we visualize the input RGB image (a, d), in the middle the predictions without the proximity loss (b,c) and on the right the predictions of our model with this additional loss term.

	IoU	Chamfer-L_1
Ours w/o proximity loss	0.605	0.171
Ours	**0.699**	**0.098**

Table 2: **Proximity loss.** We investigate the impact of the proximity loss. We report the volumetric IoU and the Chamfer distance for two variants of our model, one with and without the proximity loss term.

Fig. 8 illustrates the predictions of both variants. We remark that the predictions of the model that was trained without the proximity loss are less accurate. Note that due to the vanishing gradient problem, the network is not able to properly "move" primitives and as a result, instead of reconstructing the hands of the humans using two or four primitives, the network uses only one. Interestingly, the reconstructions in some cases e.g. (e) do not even capture the human shape properly. However, even though the reconstruction quality is bad, the network is not able to fix it because the gradients of the reconstruction loss are small (even though the reconstruction loss itself is high). This is also validated quantitatively, as can be observed from Table 2.

3. Additional Results on ShapeNet

In this section, we provide additional qualitative results on various object types from the ShapeNet dataset [2]. Furthermore, we also demonstrate the ability of our model to predict semantic hierarchies, where the same node is used for representing the same part of the object. We compare our model qualitatively with [11]. In particular, we train both models on the single-view 3D reconstruction task, using the same image renderings and train/test splits as [3]. Both methods are trained for a maximum number of 64 primitives. For our method, we empirically observed that a sharpness value $s = 10$ led to good reconstructions. Note that we do not compare qualitatively with [4, 6] as they do not provide code. Finally, we also compare our model with [11, 12] on the volumetric reconstruction task, where the input to all networks is a binary voxel grid. For a fair comparison, all models leverage the same feature encoder architecture proposed in [11].

In Fig. 10+11, we qualitatively compare our predictions with [11] for various ShapeNet objects. We observe that our model yields more accurate reconstructions compared to our baseline. Due to the use of the reconstruction quality q_k^d, our model dynamically decides whether a node should be split or not. For example, our model represents the phone in Fig. 10 (a) using one primitive (root node) and the phone in Fig. 10 (b), that consists of two parts, with two primitives. This can be also noted for the case of the displays Fig. 10 (g+j). For more complicated objects, such as aeroplanes, tables and chairs, our network uses more primitives to accurately capture the geometry of the target object. Note that for this experiment we set the threshold for q_k^d to 0.8.

Our network associates the same node with the same part of the object, as it can be seen from the predicted hierarchies in Fig. 10+11. For example, for the displays the second primitive at the first depth level is used for representing the monitor of the display, for the aeroplanes the 4-th primitive in the second depth level is used for representing the front part of the aeroplanes.

3.1. Volumetric Reconstruction

Our model is closely related to the works of Tulsiani et al. [12] and Paschalidou et al. [11]. Both [11, 12] were originally introduced using a binary occupancy grid as an input to their model, thus we also compare our model with [11, 12] using a voxelized input of size $32 \times 32 \times 32$. We evaluate the modelling accuracy of these three methods on the *animal* class of the ShapeNet dataset. To ensure a fair comparison, we use the feature encoder proposed in [12] for all three. A qualitative evaluation is provided in Fig. 9.

Our model yields more detailed reconstructions compared to [11, 12]. For example, in our reconstructions the legs of the animals are not connected and the tails better capture the geometry of the target shape. Again, we observe that our network predicts semantic hierarchies, where the same node is used for representing the same part of the animal.

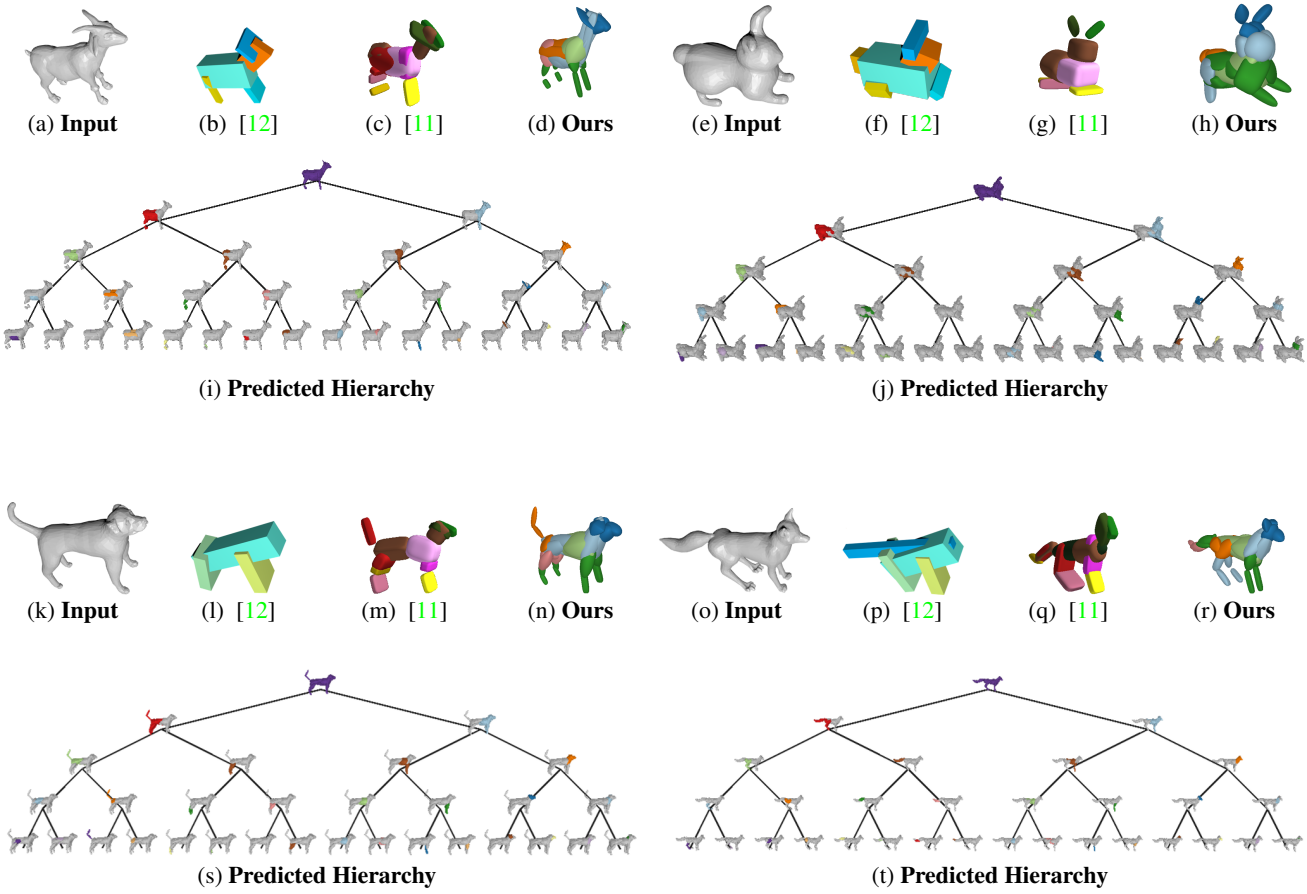
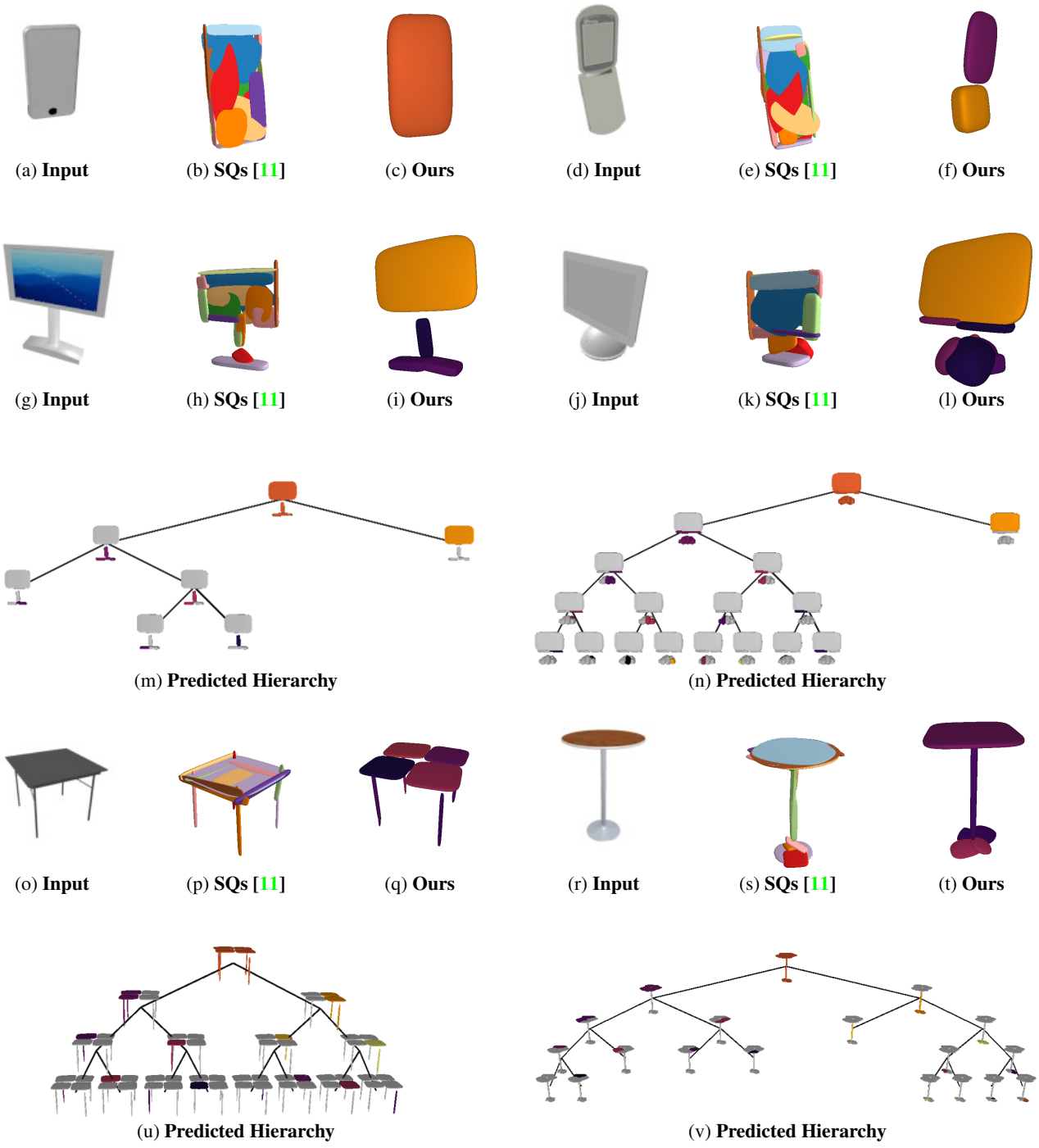


Figure 9: **Volumetric Reconstruction.** We note that our reconstructions are geometrically more accurate. In contrast to [11], our model yields reconstructions where the legs of the animals are not connected. Furthermore, our model accurately captures the ears and tails of the different animals.

4. Additional Results on D-FAUST

In this section, we provide additional qualitative results on the D-FAUST dataset [1]. Furthermore, we also demonstrate that the learned hierarchies are indeed semantic as the same node is used to represent the same part of the human body. Similar to the experiment of Section 4.2 in our main submission, we evaluate our model on the single-view 3D reconstruction task, namely given a single *RGB image as an input*, our network predicts its geometry as a *tree of primitives as an output*. We compare our model with [11]. Both methods were trained for a maximum number of 32 primitives until convergence. For our method, we set the sharpness value $s = 10$.

In Fig. 12+14, we qualitatively compare our predictions with [11]. We remark that even though [11] is more parsimonious, our predictions are more accurate. For example, we note that our shape reconstructions capture the details of the muscles of the legs that are not captured in [11]. For completeness, we also visualize the predicted hierarchy up to the fourth depth level. Another interesting aspect of our model, which is also observed in [11, 12] is related to the semanticness of the learned hierarchies. We note that our model consistently uses the same node for representing the same part of the human body. For instance, node $(4, 15)$, namely the 15-th node at the 4-th depth level, consistently represents the right foot, whereas, node $(4, 12)$ represents the left foot. This is better illustrated in Fig. 13. In this figure, we only color the primitive associated with a particular node, for various humans, and we remark that the same primitive is used for representing the same body part. Finally, another interesting characteristic of our model is related to its ability to use less primitives for reconstructing humans, with smaller bodies. In particular, while the lower part of the human body is consistently represented with the same set of



(a) **Input** (b) **SQs [11]** (c) **Ours** (d) **Input** (e) **SQs [11]** (f) **Ours**

(g) **Input** (h) **SQs [11]** (i) **Ours** (j) **Input** (k) **SQs [11]** (l) **Ours**

(m) **Predicted Hierarchy** (n) **Predicted Hierarchy**

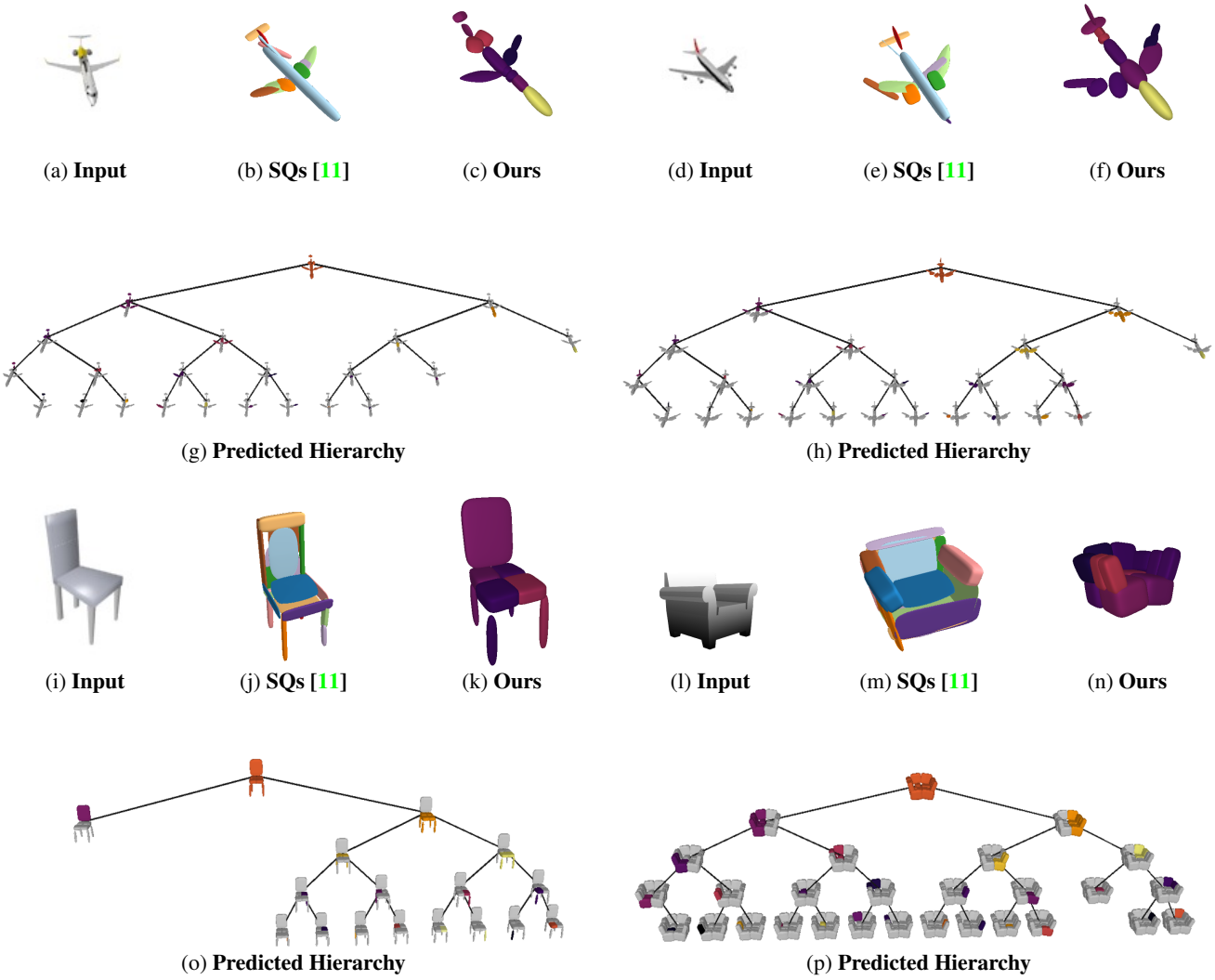
(o) **Input** (p) **SQs [11]** (q) **Ours** (r) **Input** (s) **SQs [11]** (t) **Ours**

(u) **Predicted Hierarchy** (v) **Predicted Hierarchy**

Figure 10: **Single Image 3D Reconstruction on ShapeNet.** We visualize the predictions of our model on various ShapeNet objects and compare to [11]. For objects that are represented with more than two primitives, we also visualize the predicted hierarchy.

primitives, the upper part can be represented with less depending on the size and the articulation of the human body. This is illustrated in Fig. 14, where we visualize the predictions of our model for such scenarios.

Below, we provide the full hierarchies of the results on D-FAUST from our main submission.



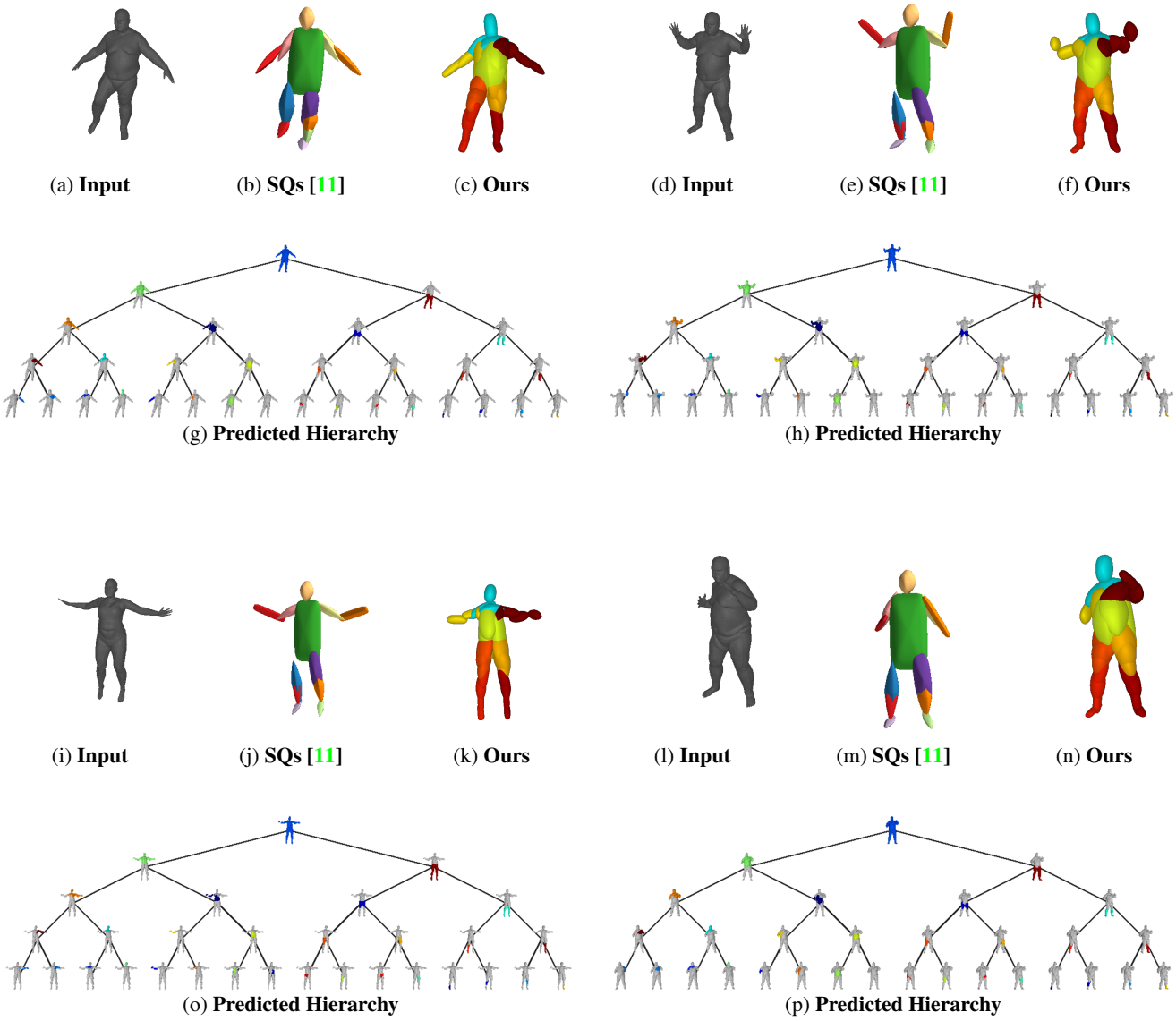
(a) **Input** (b) **SQs [11]** (c) **Ours** (d) **Input** (e) **SQs [11]** (f) **Ours**

(g) **Predicted Hierarchy** (h) **Predicted Hierarchy**

(i) **Input** (j) **SQs [11]** (k) **Ours** (l) **Input** (m) **SQs [11]** (n) **Ours**

(o) **Predicted Hierarchy** (p) **Predicted Hierarchy**

Figure 11: **Single Image 3D Reconstruction on ShapeNet.** We visualize the predictions of our model on various ShapeNet objects and compare to [11]. For objects that are represented with more than two primitives, we also visualize the predicted hierarchy.



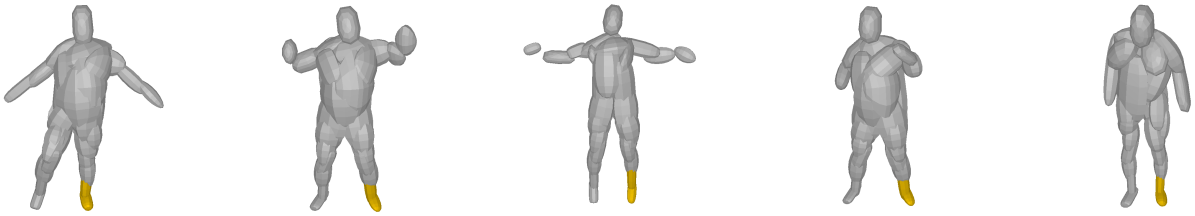
(a) **Input** (b) **SQs [11]** (c) **Ours** (d) **Input** (e) **SQs [11]** (f) **Ours**

(g) **Predicted Hierarchy** (h) **Predicted Hierarchy**

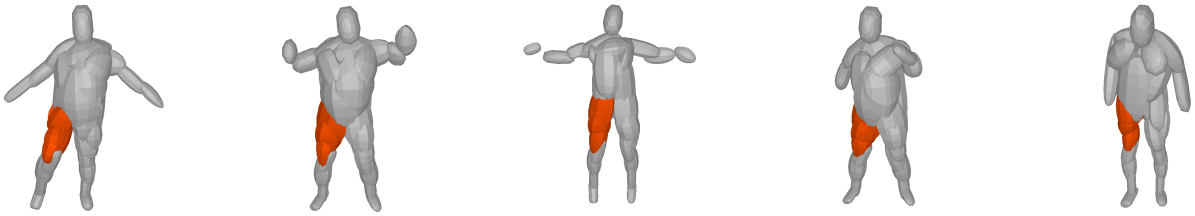
(i) **Input** (j) **SQs [11]** (k) **Ours** (l) **Input** (m) **SQs [11]** (n) **Ours**

(o) **Predicted Hierarchy** (p) **Predicted Hierarchy**

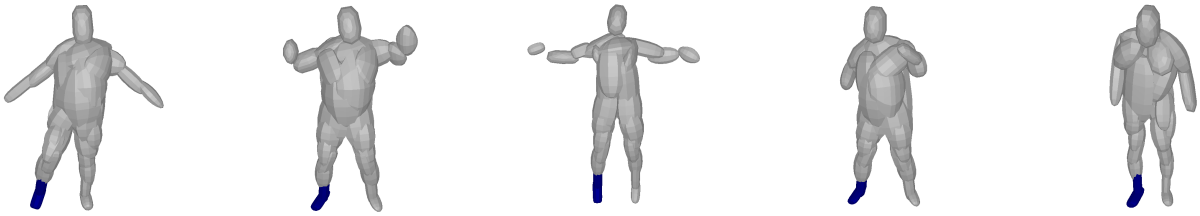
Figure 12: **Qualitative Results on D-FAUST.** Our network learns semantic mappings of body parts across different body shapes and articulations while being geometrical more accurate compared to [11].



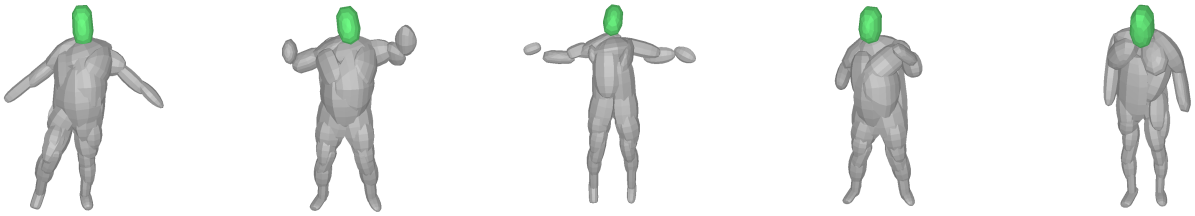
(a) Node $(4, 0)$



(b) Node $(3, 3)$



(c) Node $(4, 3)$



(d) Node $(4, 12)$

Figure 13: **Semantic Predictions on D-FAUST.** To illustrate that our model indeed learns semantic hierarchical layouts of parts, here we color a specific node of the tree for various humans and we observe that it consistently corresponds to the same body part.

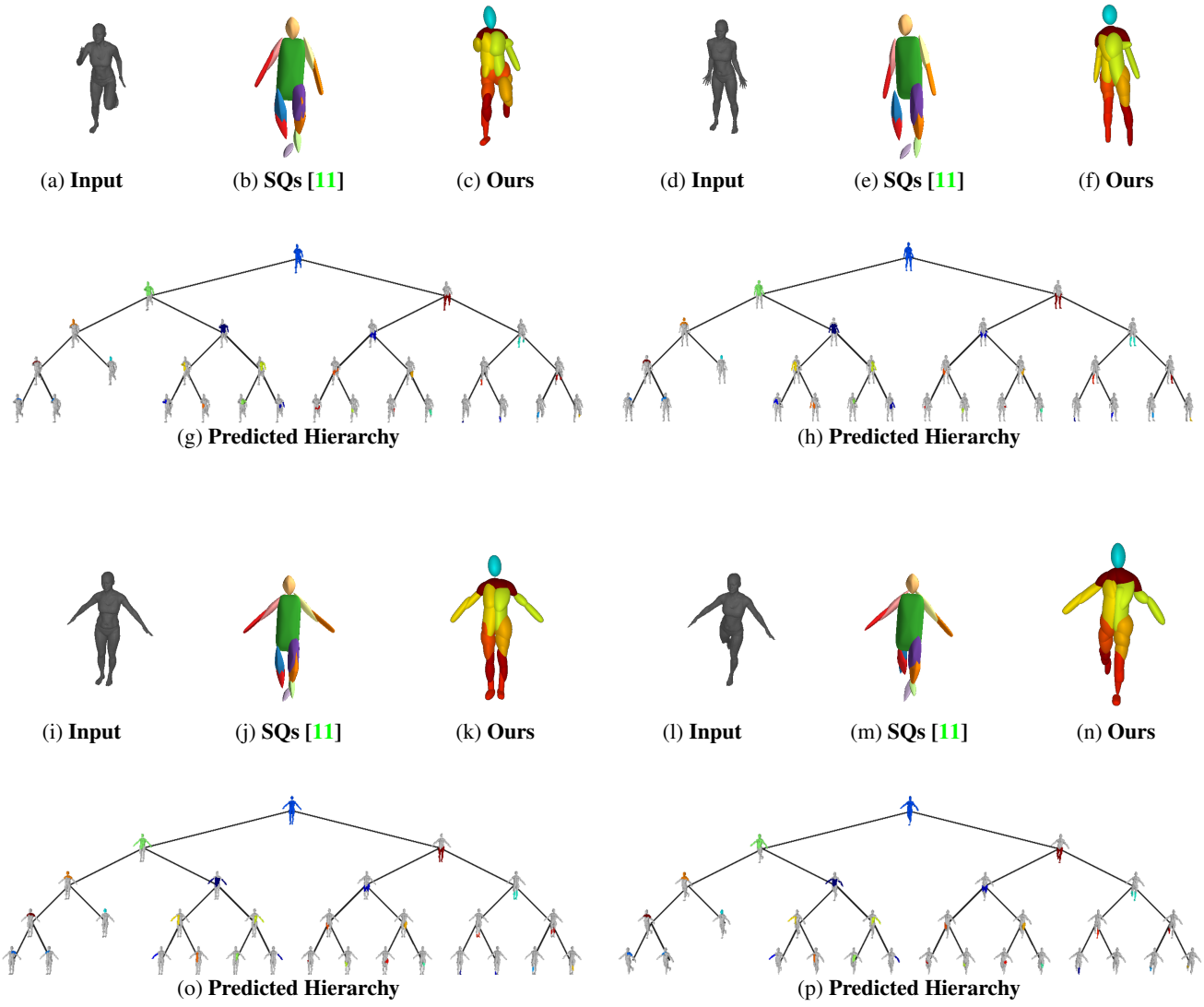


Figure 14: **Qualitative Results on D-FAUST.** Our network learns semantic mappings of body parts across different body shapes and articulations. Note that the network predicts less primitives for modelling the upper part of the human body.

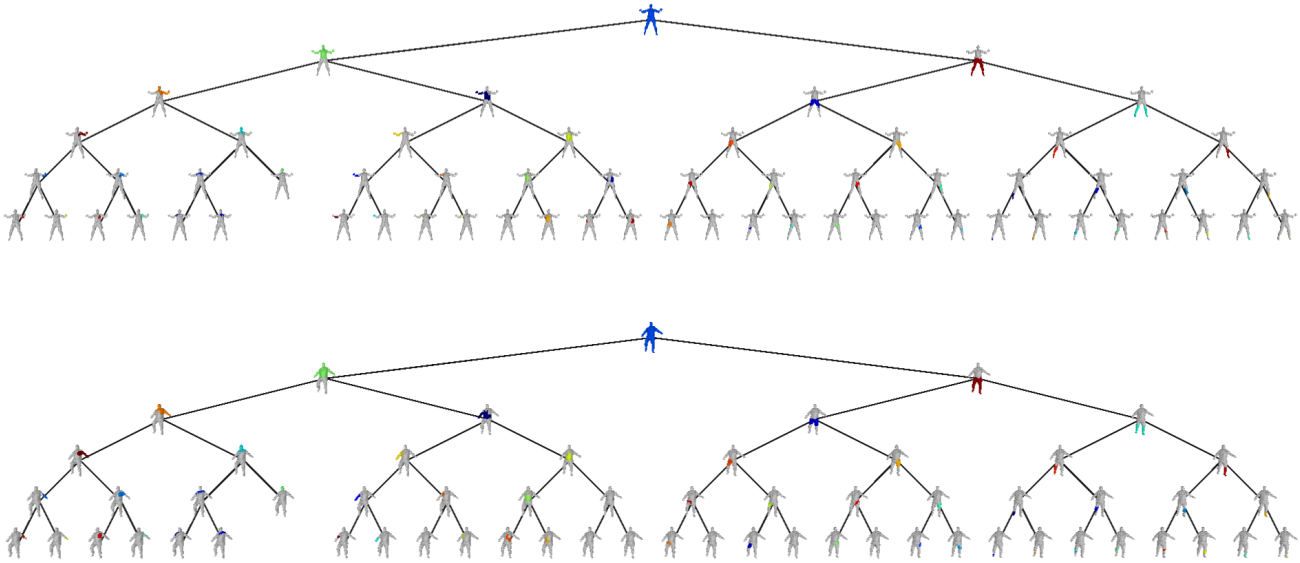


Figure 15: **Full hierarchies of Figure 6 in our main submission.** Please zoom-in for details

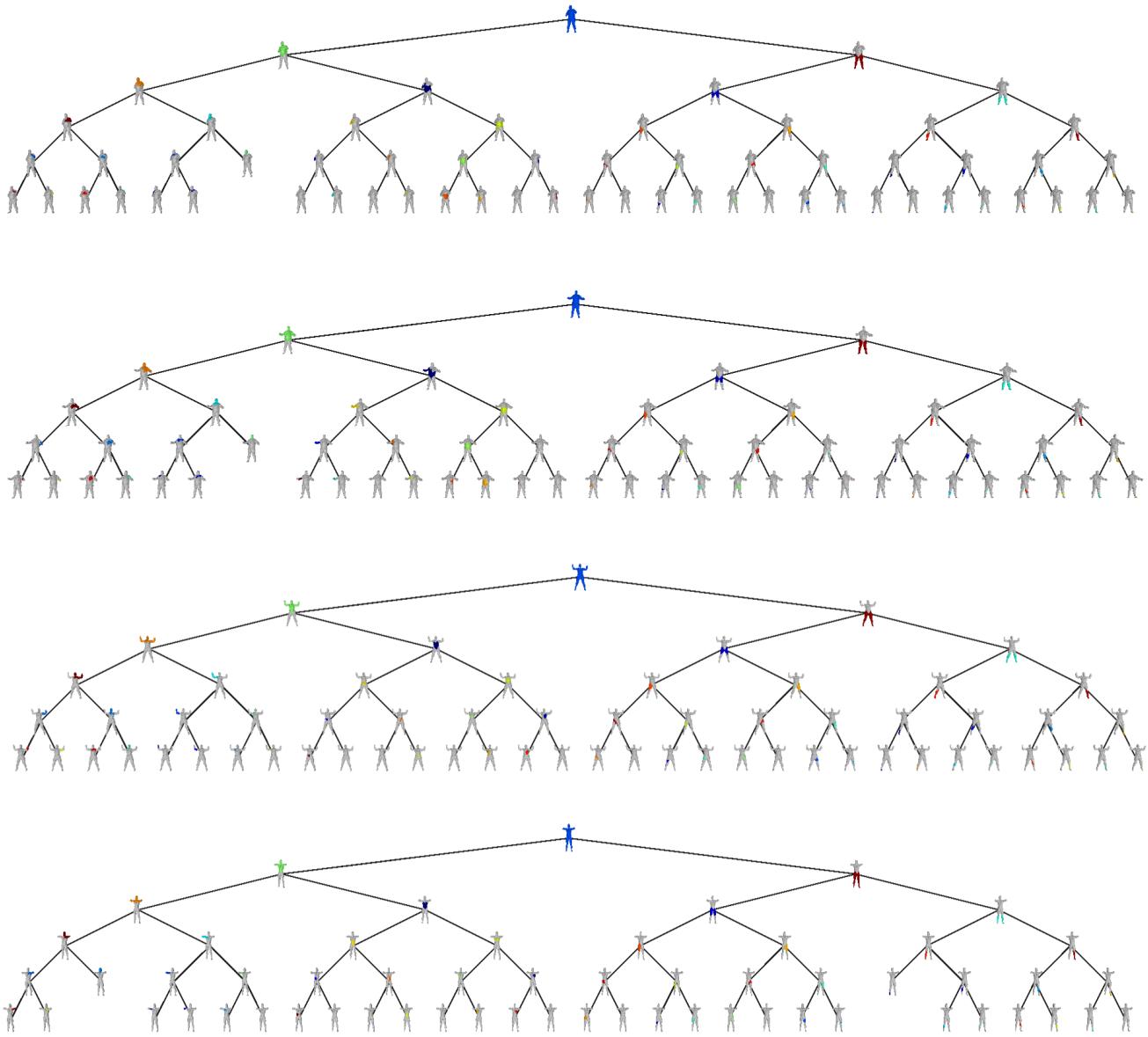


Figure 16: **Full hierarchies of Figure 8 in our main submission.** Please zoom-in for details.

References

[1] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Dynamic FAUST: registering human bodies in motion. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 12

[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, 11

[3] Christopher Bongsoo Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *Proc. of the European Conf. on Computer Vision (ECCV)*, 2016. 11

[4] Boyang Deng, Kyle Genova, Soroosh Yazdani, Sofien Bouaziz, Geoffrey Hinton, and Andrea Tagliasacchi. Cvxnets: Learnable convex decomposition. *arXiv.org*, 2019. 11

[5] Jia Deng, Wei Dong, Richard Socher, Li jia Li, Kai Li, and Li Fei-fei. Imagenet: A large-scale hierarchical image database. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2009. 6

[6] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T. Freeman, and Thomas A. Funkhouser. Learning shape templates with structured implicit functions. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 11

[7] 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. 6

[8] Ales Jaklic, Ales Leonardis, and Franc Solina. *Segmentation and Recovery of Superquadrics*, volume 20 of *Computational Imaging and Vision*. Springer, 2000. 2

[9] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. of the International Conf. on Learning Representations (ICLR)*, 2015. 7

[10] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 8

[11] Despoina Paschalidou, Ali Osman Ulusoy, and Andreas Geiger. Superquadrics revisited: Learning 3d shape parsing beyond cuboids. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 6, 8, 11, 12, 13, 14, 15, 17

[12] 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. 8, 11, 12