Multi-Category Mesh Reconstruction From Image Collections Supplementary Material

Alessandro Simoni*, Stefano Pini*, Roberto Vezzani, Rita Cucchiara
Department of Engineering "Enzo Ferrari", University of Modena and Reggio Emilia (Italy)
{alessandro.simoni, s.pini, roberto.vezzani, rita.cucchiara}@unimore.it

1. Introduction

In this supplementary material, we report architectural details of the proposed method in Section 2, followed by an analysis of the computational performance in Section 3. In section 4, we present additional ablation studies, including the impact of pre-training and foreground mask quality; an analysis of the meanshape learning process; the use of a different number of meanshapes on CUB; a study on the unsupervised shape selection module, in terms of classification accuracy and average meanshape weight. Finally, additional qualitative results and failure cases are reported in Section 5.

2. Architectural details

In this section, we firstly describe the architectural details of our method. Then, we report the weights used to balance the losses during the training process.

2.1. Network

Here, we report the implementation details of each module of the proposed framework.

Feature extraction. We use ResNet-18 [4] as visual encoder, replacing the classification layer with an additional convolutional layer with kernel size $k = 4$, stride $s = 2$, and 256 filters. Taking as input an RGB image $I \in \mathbb{R}^{3\times256\times256}$, the encoder outputs a feature map $f_{\text{tex}} \in \mathbb{R}^{256\times4\times4}$. These features are then flattened and given as input to a 256-d fully connected layer with batch normalization and a leaky ReLU activation function, obtaining a 256-d feature vector f_{shape}. The visual encoder is pre-trained on ImageNet [1]. We investigate the impact of pre-training on the unsupervised shape selection in Section 4.1.

Unsupervised shape selection. The unsupervised shape selection module is a network that smoothly approximates the argmax function over the N meanshapes. It is composed of two fully-connected layers: (i) a 64-d layer with batch normalization and leaky ReLU, (ii) a N-d layer followed by a softmax activation function that outputs the N weighting scores. The input of the module are the features f_{shape}.

Vertex deformation. Inspired by the work of Park *et al*. [7], the vertex deformation network is composed of four 512-d fully connected layers with weight normalization, random dropout of 0.2, and the ReLU activation function. An additional 3-d fully connected layer with a tanh activation function outputs the displacement $\Delta v_j = (\Delta x, \Delta y, \Delta z)$ of the vertex v_j, which is given as input along with the features f_{shape} and the weighting scores of the previous module. The input features (*i.e*. vertex location, f_{shape}, and weighting scores) are also concatenated to the output of the second layer, before applying the third one.

3D pose regression. The prediction of the object viewpoint is tackled as a regression problem using two fully connected layers: (i) a 64-d layer with batch normalization, random dropout of 0.5, and leaky ReLU, (ii) a 7-d layer that outputs the object pose $\hat{\pi} = (\hat{s}, \hat{t}, \hat{q}) \in (\mathbb{R}^1, \mathbb{R}^2, \mathbb{R}^4)$. The input of the module are the features f_{shape}.

Texture prediction. Inspired by the decoder of the SPADE architecture proposed by Park *et al*. [8], our texture decoder is composed of 6 upsampling steps with bilinear interpolation, in order to output a texture image $I_{\text{tex}} \in \mathbb{R}^{3\times256\times256}$. Differently from the original implementation, we use only

Dataset	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6	λ_7	λ_8	λ_9
Pascal3D+	100.0	6.0	1.8	0.05	20.0	2.0	0.03	0.05	0.8
CUB	20.0	1.2	0.18	0.005	2.0	0.1	0.12	0.02	3.2

Table 1: Loss weights on Pascal3D+ [10] and CUB [9].

Method	**Params** (M)	**Memory** (GB)	**Inference** (ms)
CMR [5]	84.25	2.28	3.56 ± 0.14
U-CMR [2]	19.89	3.38	5.10 ± 2.91
Ours	20.10	3.74	4.43 ± 0.19

Table 2: Performance analysis of our multi-category approach against open-sourced single-category competitors.

* Equal contribution.

Training classes	Segmentation Method	Number of meanshapes	3D IoU ↑	Mask IoU ↑		Texture metrics		
				Pred cam	GT cam	SSIM ↑	L1 ↓	FID ↓
aeroplane, car	Mask R-CNN	2	**0.556**	0.648	0.699	**0.739**	0.064	350.12
aeroplane, car	PointRend	2	0.552	**0.671**	**0.702**	0.737	**0.062**	**344.80**
bicycle, bus, car, motorbike	Mask R-CNN	4	0.530	0.677	0.756	0.605	0.098	390.55
bicycle, bus, car, motorbike	PointRend	4	**0.543**	**0.711**	**0.759**	**0.607**	**0.094**	**380.15**

Table 3: Evaluation on Pascal3D+ [10] using segmentation masks obtained with Mask R-CNN [3] or PointRend [6].

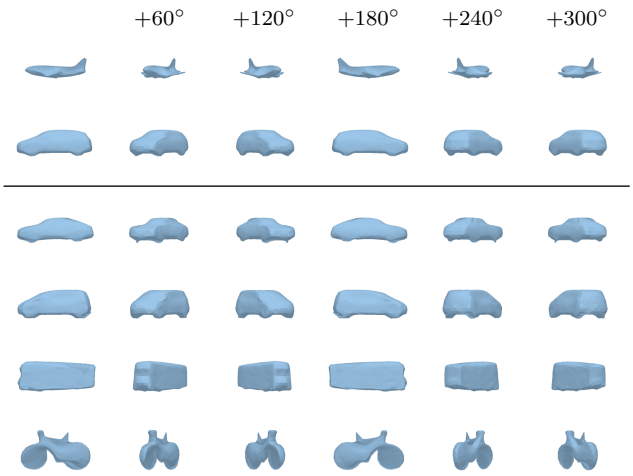

Figure 1: Meanshapes learned by our method trained on aeroplanes and cars, and on 4 automotive classes of Pascal3D+ [10] without encoder pre-training on ImageNet.

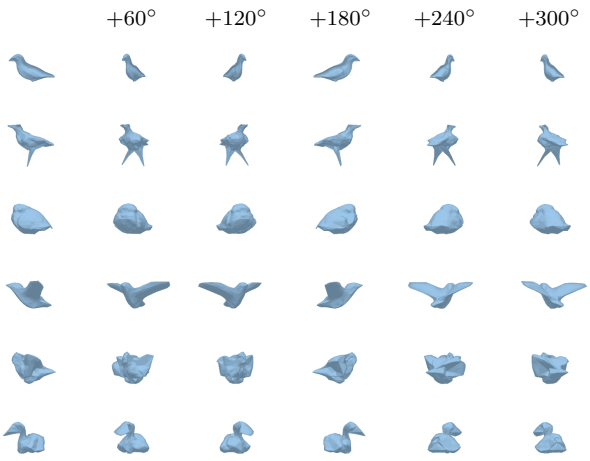

Figure 2: Some of the meanshapes learned by our method trained on CUB [9], using 14 meanshapes, without encoder pre-training on ImageNet.

the convolutional layers with skip connections and leaky ReLU activation functions. We test different types of normalization (*e.g.* batch, instance), but we obtain the best results without it. The decoder takes as input the features f_{tex} and the output is finally passed through a sigmoid activation function in order to obtain valid RGB color values.

2.2. Loss weights

In the following, we reintroduce the losses used during training in order to show their weighting parameters, whose values are reported in Table 1. We select different weights for each dataset, exploiting their validation set.

For the shape prediction, the loss is defined by:

$$\mathcal{L}_{\text{shape}} = \lambda_1 \mathcal{L}_{\text{mask}} + \lambda_2 \mathcal{L}^{\hat{M}}_{\text{smooth}} + \lambda_3 \mathcal{L}^{\Delta V}_{\text{smooth}} + \lambda_4 \mathcal{L}_{\text{def}} \quad (1)$$

where the smoothness prior is applied to both the vertices of deformed shape $\hat{M}$ and the predicted deformations ΔV. For the pose regression, the loss is defined as:

$$\mathcal{L}_{\text{cam}} = \lambda_5 \mathcal{L}_{\text{pose}} + \lambda_6 \mathcal{L}_{\text{pose\_reg}} \quad (2)$$

while the texture prediction loss is represented by:

$$\mathcal{L}_{\text{tex}} = \lambda_7 \mathcal{L}_{\text{color}} + \lambda_8 \mathcal{L}_{\text{style}} + \lambda_9 \mathcal{L}_{\text{percept}} \quad (3)$$

3. Computational Performance

In this section, we assess the computational requirements of our method and some open-sourced competitors. Compared to previous category-specific methods, our approach does not require an initial shape classifier and the training on N independent models, thus being faster and requiring less memory during inference. Indeed, our multi-category method has comparable network size, memory usage, and inference time with respect to the single-category competitors, as reported in Table 2. Their evaluation is conducted on a workstation with an *Intel Core* i7-7700K and a *Nvidia GeForce* GTX 1080 Ti.

4. Additional ablation studies

In this section, we present further experiments on the datasets Pascal3D+ and CUB.

4.1. Impact of pre-training on shape selection

Since our model exploits a visual encoder pre-trained on ImageNet [1], we investigate the impact of using pre-trained weights or training the encoder from scratch, with a particular focus on the unsupervised shape selection module. Indeed, we aim to verify that the proposed method is capa-

Training classes	ImageNet pre-train	Number of meanshapes	3D IoU ↑	Mask IoU ↑		Texture metrics		
				Pred cam	GT cam	SSIM ↑	L1 ↓	FID ↓
aeroplane, car	✔	2	**0.550**	**0.639**	**0.700**	**0.732**	**0.066**	**353.61**
aeroplane, car		2	0.541	0.599	0.675	0.728	0.069	357.47
bicycle, bus, car, motorbike	✔	4	**0.543**	**0.711**	**0.759**	**0.607**	**0.094**	**380.15**
bicycle, bus, car, motorbike		4	0.534	0.632	0.727	0.580	0.111	392.71

Table 4: Evaluation on Pascal3D+ [10] using a ResNet-18 encoder with or without pre-trained weights on ImageNet [1] (segmentation masks obtained with PointRend [6]).

Imagenet pre-train	Mask IoU ↑		Texture metrics		
	Pred cam	GT cam	SSIM ↑	L1 ↓	FID ↓
✔	**0.642**	**0.723**	**0.715**	**0.065**	**231.95**
	0.563	0.699	0.693	0.077	259.36

Table 5: Evaluation on CUB [9] using a ResNet-18 encoder with or without pre-trained weights on ImagetNet [1].

ble of learning meaningful meanshapes even without a pre-trained feature extractor. Quantitative results and learned meanshapes are reported (i) in Table 4 and Figure 1 for Pascal3D+ and (ii) in Table 5 and Figure 2 for the CUB dataset. IoU and texture metrics show that the pre-trained version obtains better scores in every setting. However, it is worth noting that the framework is capable of obtaining satisfactory results and learning meaningful meanshapes even without any pre-training of the encoder network, confirming the effectiveness of the proposed shape selection module.

4.2. Impact of finer foreground masks

In this section, we compare the scores obtained on Pascal3D+ using rough foreground masks, provided by Mask R-CNN [3], or more precise masks, obtained with PointRend [6]. Results are reported in Table 3. As expected, there is a clear advantage in using finer masks in the setting with 4 automotive classes. Indeed, PointRend produces accurate masks, which present fine details and sharp edges, that are leveraged by the framework during the training process. On the other hand, a relatively small improvement can be observed when training on just aeroplanes and cars. This may be due to a different quality of the aeroplane masks between Mask R-CNN and PointRend.

4.3. Meanshape learning during training

In order to evaluate the unsupervised learning of multiple meanshapes during the training process, we report the learned shapes at different epochs in Figure 3 (Pascal3D+) and in Figure 4 (CUB). These results show that the method distinguishes different object categories within the first few epochs and then progressively optimize each meanshape accordingly. While the classes are clearly disentangled in just tens of epochs on Pascal3D+, the same process requires more epochs on CUB. We believe that this difference is due to the class type: classes of different entities on Pascal3D+, different classes of the same entity "bird" on CUB. Nevertheless, the method progressively learns meaningful meanshapes in both settings.

Number of meanshapes	Mask IoU ↑		Texture metrics		
	Pred cam	GT cam	SSIM ↑	L1 ↓	FID ↓
1	**0.658**	0.721	0.717	0.064	**227.24**
10	0.657	0.721	**0.720**	**0.063**	232.84
14	0.642	0.723	0.715	0.065	231.95
18	0.648	**0.724**	0.715	0.065	228.24

Table 6: Evaluation on CUB [9] using different numbers of meanshapes (1, 10, 14, 18).

4.4. Number of meanshapes on CUB

The CUB dataset contains images of the same category "bird". However, the dataset can be split in many subcategories, for instance using the annotated bird type (200 different values) or one of the other annotated categorical attributes (*e.g.* the "has_shape" one provides 14 different values, including *duck-like*, *gull-like*, *hummingbird-like*, *long-legged-like*). Thus, in the paper we empirically set the number of meanshapes as the number of the "has_shape" attribute values. Here, we analyze the impact of using different numbers of meanshapes, testing the framework with 1, 10, 14, and 18 meanshapes and reporting the results in Table 6. Differently from the training on Pascal3D+, in this case there are no clear advantages, in terms of mask IoU and texture scores, in using a single or multiple meanshapes. However, as clearly shown in the paper and in Figure 4, the method can exploit the available meanshapes to learn meaningful base shapes in an unsupervised manner. These base shapes can then be used as representative shapes for the whole dataset or as bird templates in other tasks. In addition, we did not find an explicit pattern in using different numbers of meanshapes. This shows that the initialization of this hyper-parameter is not crucial for the learning process, in particular when the class division is not perfectly clear.

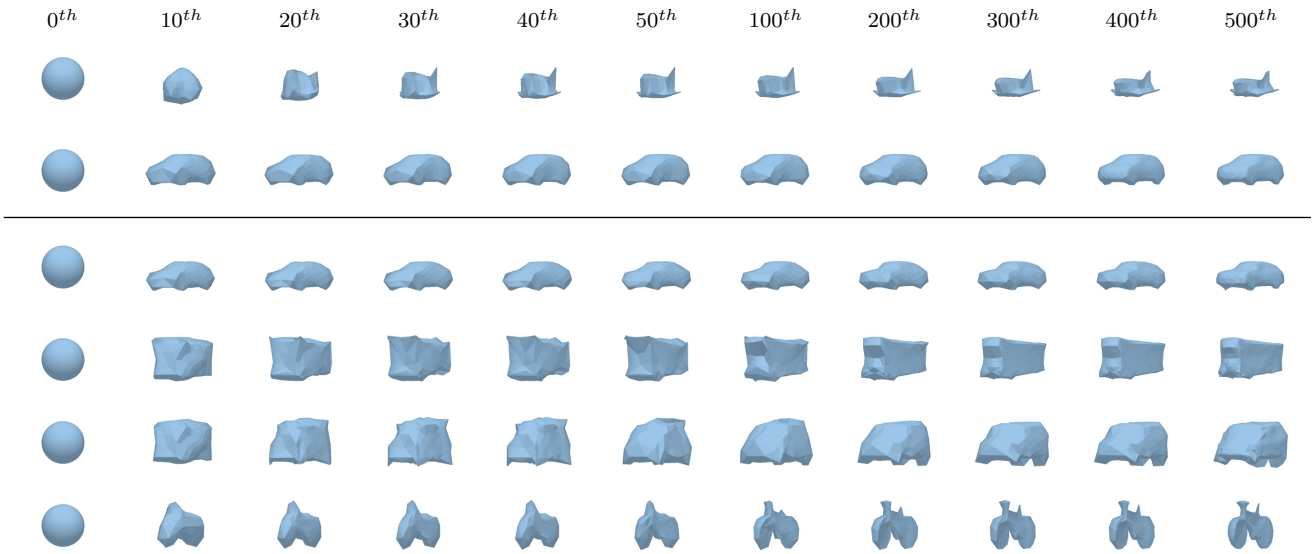

Figure 3: Meanshapes learned during different epochs of the training procedure on aeroplanes and cars (rows 1-2), and on 4 automotive classes (rows 3-6) of Pascal3D+ [10].

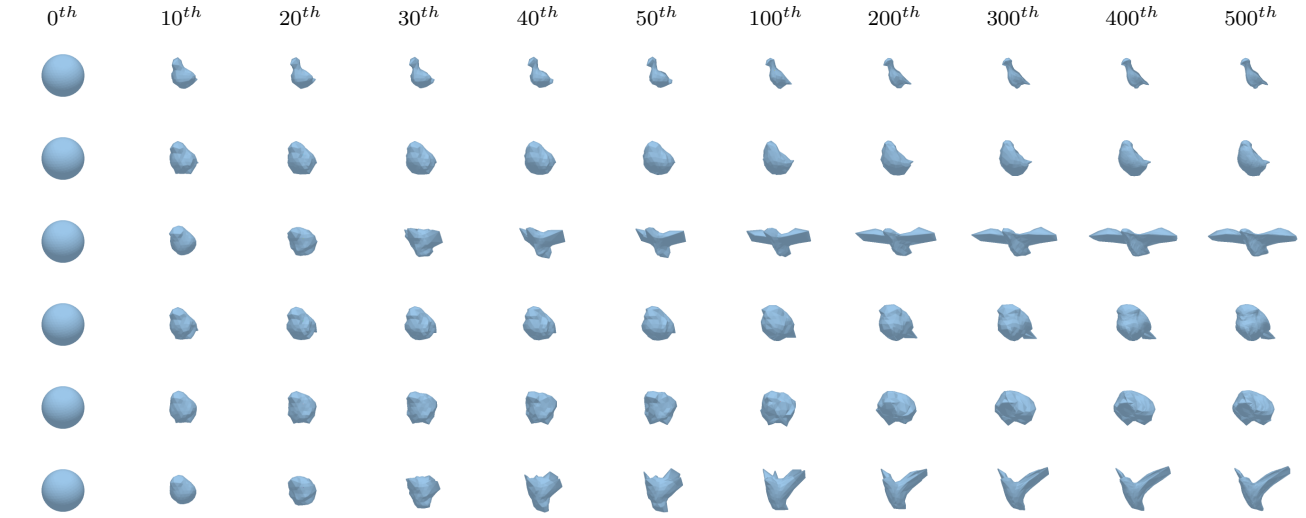

Figure 4: Some of the meanshapes learned during different epochs of the training procedure on CUB [9], using 14 meanshapes.

4.5. Unsupervised shape classification

In this section, we investigate the usage of the unsupervised shape selection module as classifier on the Pascal3D+ dataset. In particular, we evaluate whether the most weighted meanshape represents the object category. In the 2-class setting (aeroplane, car), the obtained classification accuracy is 98.82%; in the 4-class setting (bicycle, bus, car, motorbike), the classification accuracy is 93.45%. In the latter case, the classes bicycle and motorbike are considered a single class, given that the method learned a single meanshape that represents both.

4.6. Average meanshape weights

To evaluate the importance of each meanshape on the predicted shape, we compute the average meanshape weight predicted by the unsupervised shape selection module. Results are reported in Figure 5 for all the meanshapes of the experiments with aeroplanes (Pascal3D+) and birds (CUB). While we acknowledge that there are few learned meanshapes that do not correspond to a clear object category, these meanshapes have a marginal impact on the weighted meanshape. On the contrary, the most representative meanshapes have, on average, a major contribution

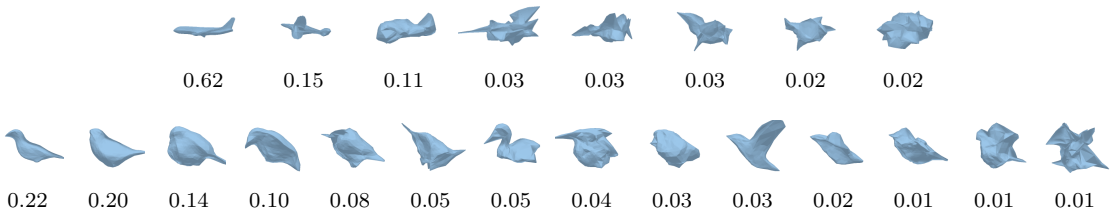

Figure 5: Average meanshape weights for Pascal3D+ (aeroplane class, 8 meanshapes) and CUB (14 meanshapes) ordered from the most weighted to the least one.

on the weighted one.

5. Additional qualitative results

We report additional qualitative results for the CUB dataset in Figure 6 and for experiments on Pascal3D+ in Figure 7 (all 12 classes), Figure 8 (4 automotive classes) and Figure 9 (aeroplane, car).

5.1. Failure cases

In Figure 10, we report some failure cases of our method trained on 4 automotive classes of Pascal3D+. First of all, we identified some rare cases in which the predicted meanshape is incorrect. For instance, bicycles with large wheels are sometimes mistaken for motorbikes while cars with roofboxes are confused with buses (Fig. 10, rows 1-3). Moreover, we detected that the method sometimes outputs wrong deformations, causing the objects to be skewed, when the viewpoint is very close to the object (Fig. 10, rows 4-5). Finally, in some cases the method can not predict correct deformations of articulated parts (Fig. 10, rows 6).

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 1, 2, 3

[2] Shubham Goel, Angjoo Kanazawa, and Jitendra Malik. Shape and viewpoint without keypoints. In *European Conference on Computer Vision*, 2020. 1

[3] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *IEEE International Conference on Computer Vision*, pages 2961–2969, 2017. 2, 3

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 1

[5] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *European Conference on Computer Vision*, pages 371–386, 2018. 1

[6] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 9799–9808, 2020. 2, 3

[7] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. 1

[8] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2337–2346, 2019. 1

[9] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 1, 2, 3, 4, 6

[10] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *IEEE Winter Conference on Applications of Computer Vision*, pages 75–82, 2014. 1, 2, 3, 4, 7, 8, 9

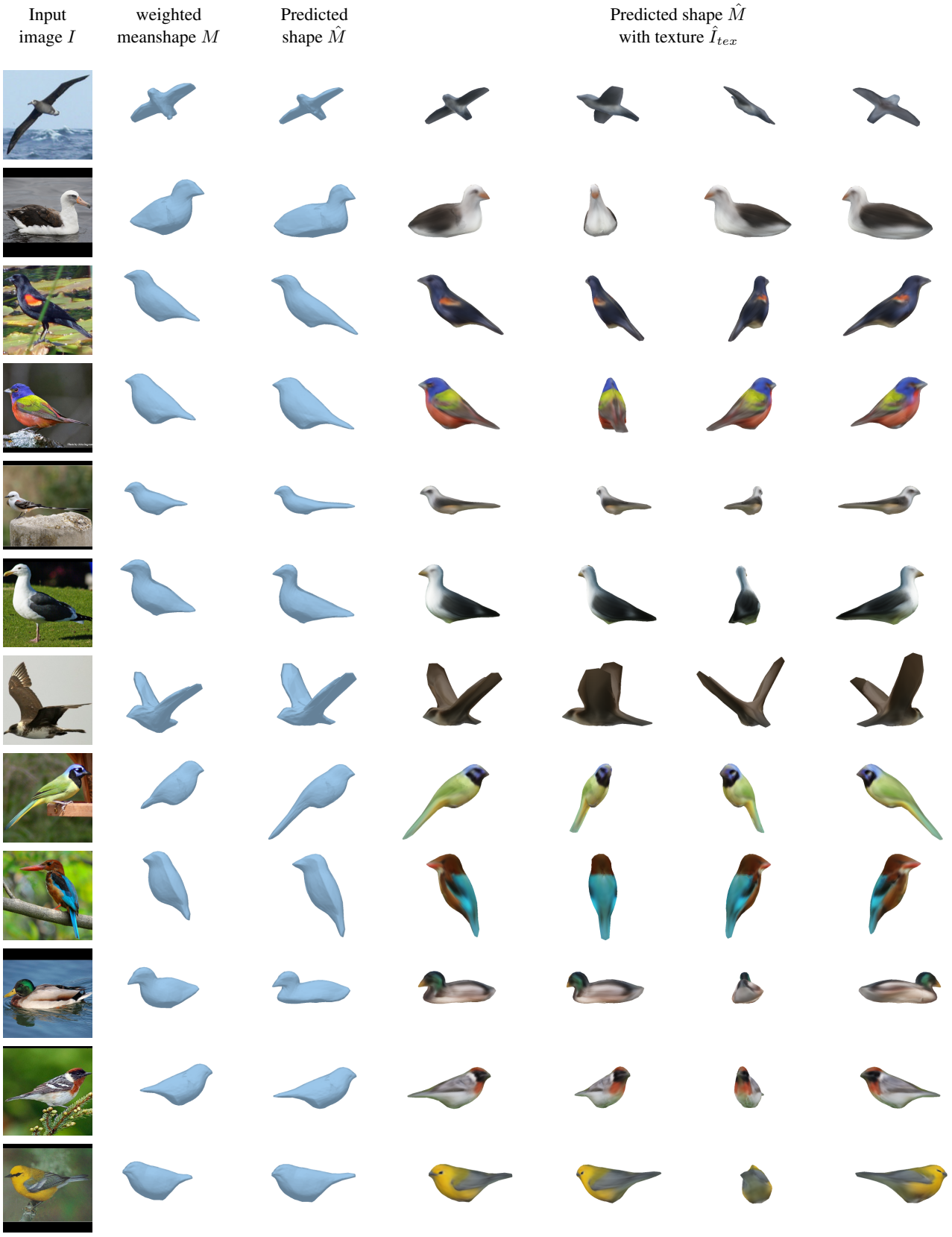

Figure 6: Additional qualitative results of our method trained on CUB [9], using 14 meanshapes.

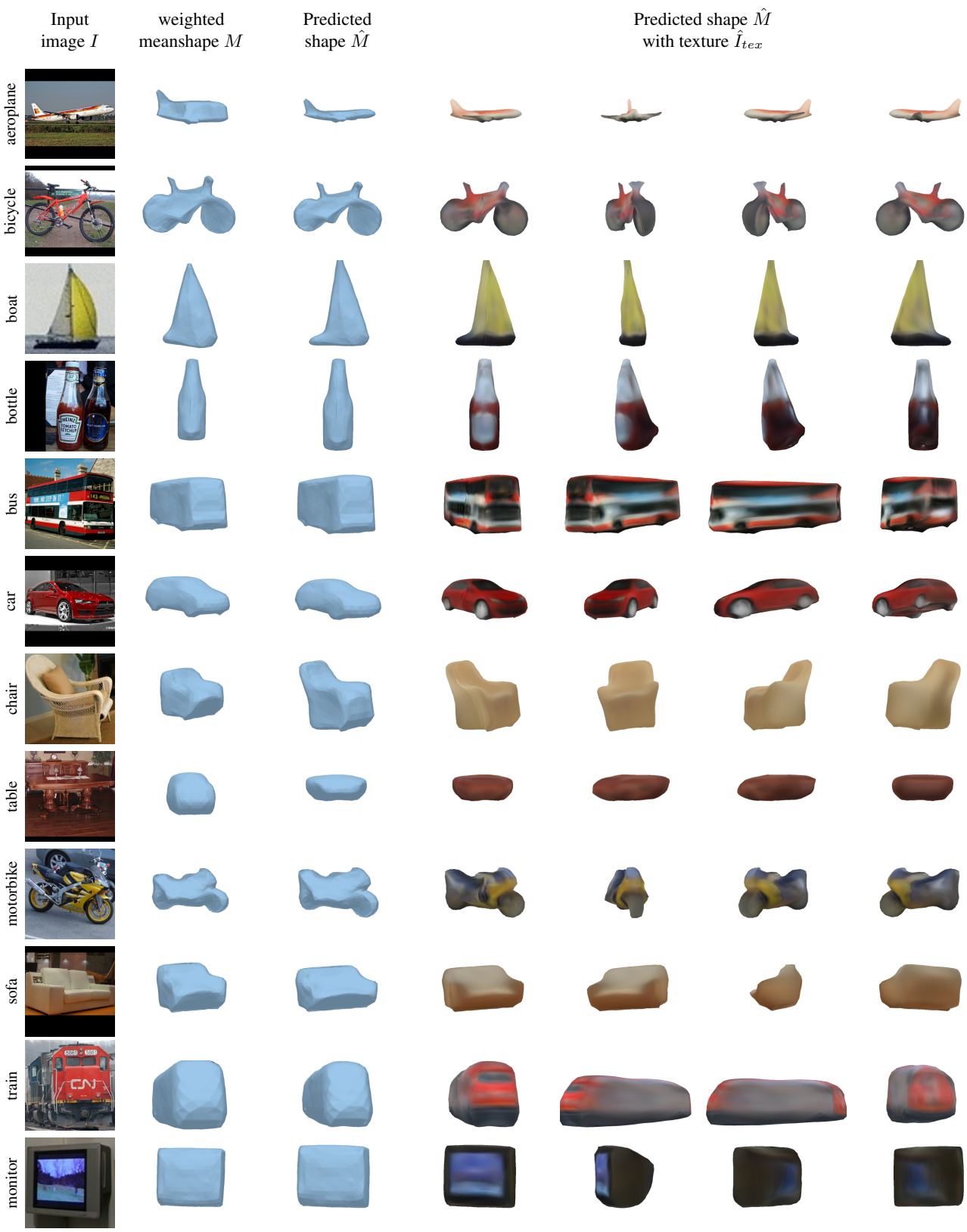

Figure 7: Additional qualitative results of our method trained jointly on all 12 classes of Pascal3D+ [10].

Figure 8: Additional qualitative results of our method trained jointly on 4 automotive classes (bicycle, bus, car, motorbike) of Pascal3D+ [10].

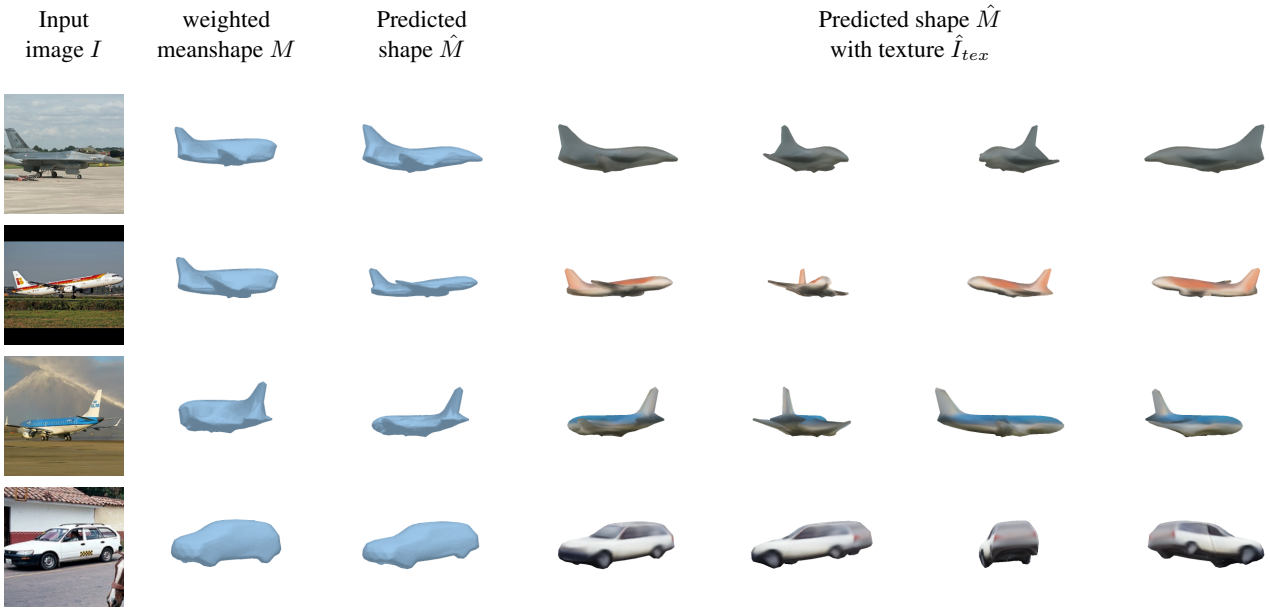

Figure 9: Additional qualitative results of our method trained jointly on aeroplanes and cars of Pascal3D+ [10].

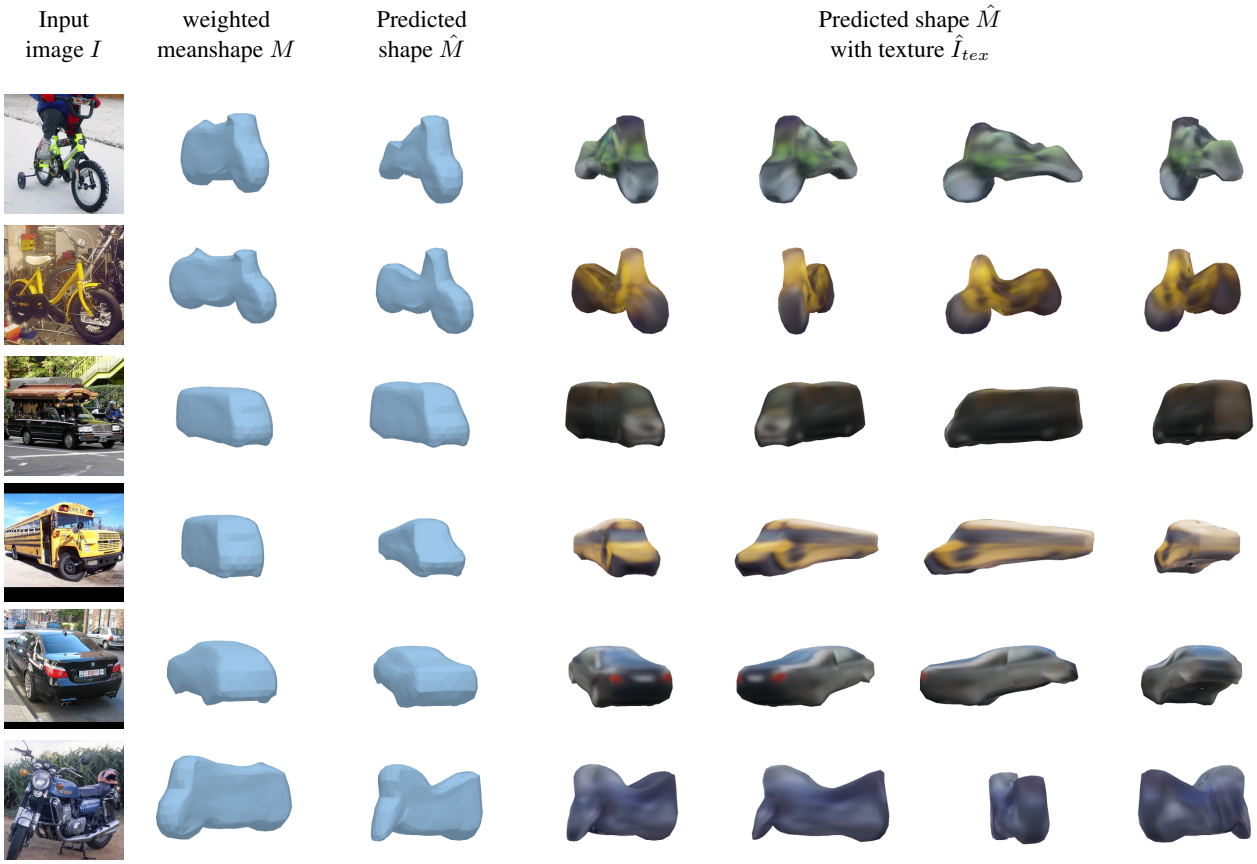

Figure 10: Some failure cases of our method trained jointly on 4 automotive classes (bicycle, bus, car, motorbike) of Pascal3D+ [10].