View Generalization for Single Image Textured 3D Models Supplementary Material

Anand Bhattad[1]* Aysegul Dundar[2,3] Guilin Liu[3] Andrew Tao[3] Bryan Catanzaro[3]

[1]University of Illinois at Urbana-Champaign [2]Bilkent University [3]NVIDIA

A. Implementation Details

Network architecture. For predicting deformation and texture maps, we use a U-Net encoder-decoder architecture. The encoder is shared between the deformation and texture maps networks. The encoder contains 7 layers of Convolution-BatchNorm-LeakyReLU. Convolution layers have a kernel size of 5, padding 2, and stride 2. At each layer, the number of filters doubles and goes as follows: $(32, 64, 128, 256, 512, 512, 512)$. The decoder for the texture network is a mirror-symmetric of the encoder and the number of filters are as follows $(1024, 1024, 1024, 512, 256, 128, 32)$. The number of filters in the decoder is two times of that encoder because the feature maps from the encoder skip to the corresponding decoder module and concatenates with the sequentially flowing feature maps. There are bilinear interpolation layers to upsample the feature maps at each layer. For the deformation decoder, we follow a similar architecture but output from an earlier layer in the decoder. For example for 16×16 decoder map, there are 4 decoder layers with the following number of filters $(512, 512, 256, 128)$, and there are no skip connections from the encoder to the decoder. For both decoders, there is a final convolution layer that decreases the number of channels to 3 for texture decoder they represent (R,G,B) channels and for the displacement decoder they represent the deformations (x,y,z) coordinates.

For predicting the camera parameters and template weights, we use a ResNet18 model pretrained on ImageNet. Same pretrained ResNet18 is also used by DIB-R for mesh prediction. We use the pretrained convolutional layers of ResNet18 and stack a fully connected layer which outputs a feature vector with 200 dimension. For camera parameters and template weights, the network branches out to two linear layers which decreases the number of feature vectors to the corresponding number of output parameters.

We train our model on 8 GPUs with batch size of 16 per GPU, for 1200 epochs with learning rate of 1^{-4}.

*work done while interning with NVIDIA

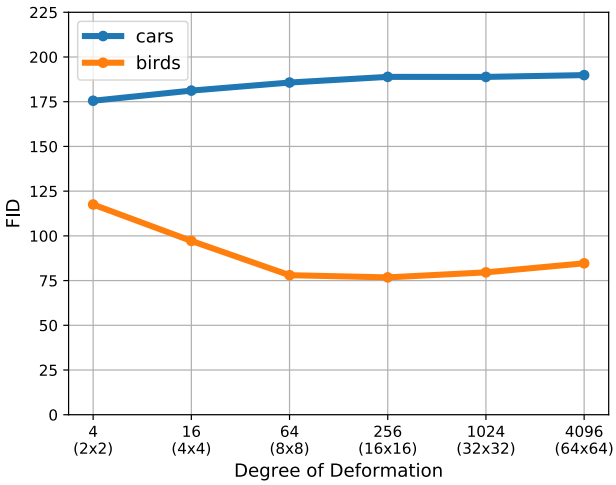

Figure 1: **DOD Ablation study.** For cars (a rigid object), increase in degree of deformation (DOD) results in poor generalization. Therefore, we use $DOD = 4$. For birds (a non-rigid object), we need a model that is flexible to undergo a reasonable deformation. We find $DOD = 256$ to generalize best across views.

B. User Study Set-up

We evaluate our algorithm via a human subjective study. We perform pair-wise A/B tests deployed on the Amazon Mechanical Turk (MTurk) platform. We give users an input image, and two GIFs at once, each of which is synthesized from a different method as shown in Figure 4. We give users unlimited time to select which GIFs look more realistic. The left-right order and the image order are randomized to ensure fair comparisons. Each test image, in total 220 of them, are compared 10 times, resulting in 2200 comparisons. Random chance results in 50% preference. In our studies, users pick our method when competed against our baseline and DIB-R for i) better texture, ii) better shape, iii) better overall synthesis. We found that the average time spent on each paired comparison was about 12 seconds.

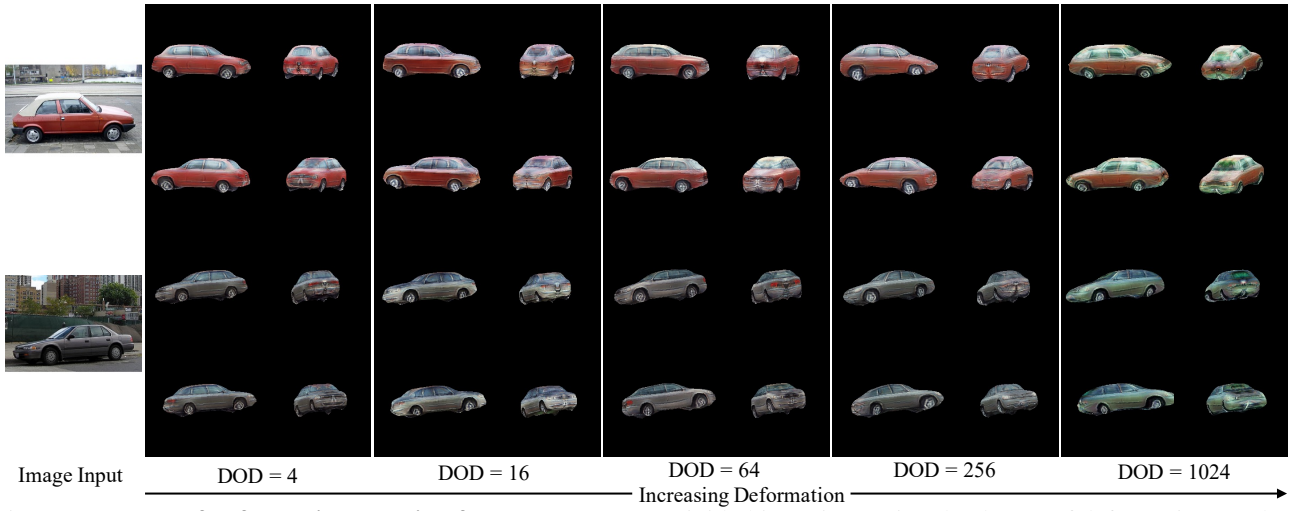

Figure 2: **Degree of deformation ablation for cars.** For cars (a rigid object), increasing the degree of deformation results in poor textured 3D model synthesis. Results degrade moving from left to right (with increasing deformation).

Figure 3: First row shows texture predictions from models trained without texture consistency and second row with texture consistency. We provide purple and blue boxes for easy visualization.

Table 1: Quantitative evaluations for evaluating reconstruction accuracy from the original view.

	LPIPS ↓	SSIM ↑	PSNR ↑	MSE ↓
DIB-R	0.33	0.86	15.84	2023.7
Ours	**0.31**	**0.93**	**16.18**	**1888.1**

C. Additional Qualitative Results

We provide more illustrative examples for birds in Figure 5 showing overall improvements in shape and texture synthesis using proposed consistency losses. Our losses completes texture with appropriate patterns for occluded regions while our baseline creates a flat colored texture without details for occluded regions.

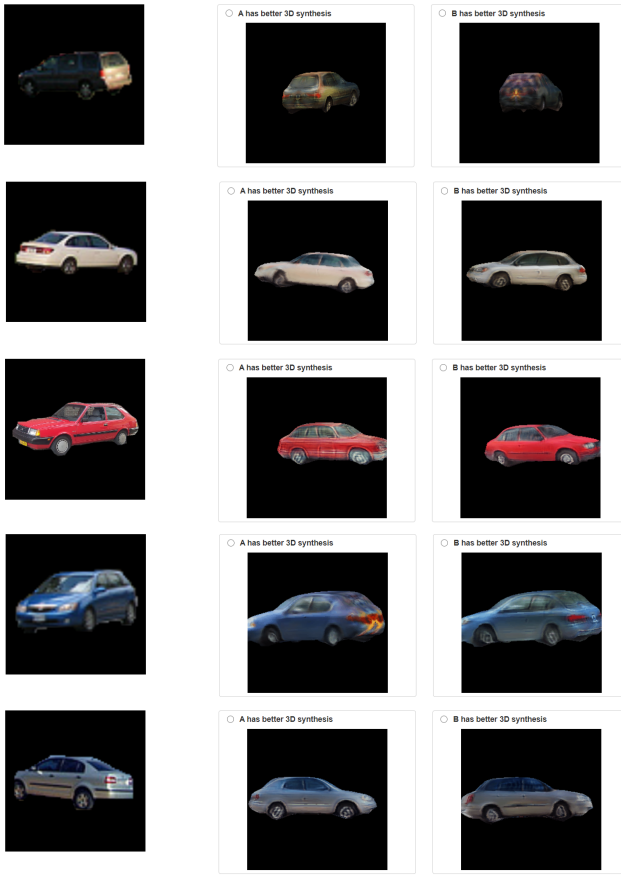

Figure 4: User study snapshots from AMT. Given input images on the left, users are asked to choose the better quality 3D inference result. 3D inference results are presented in GIF format that rotates 360°.

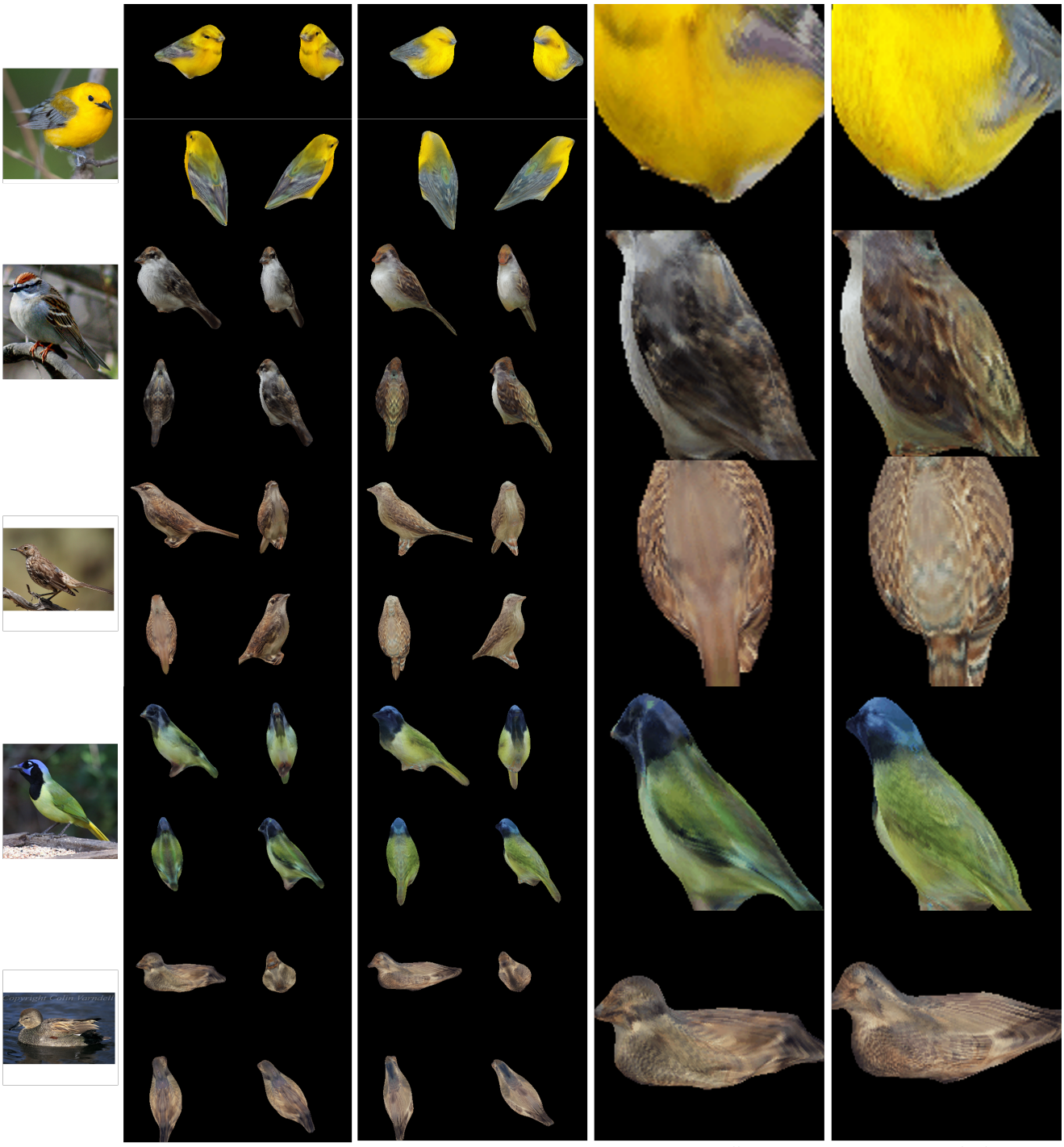

Figure 5: More example results on birds with $DOD = 256$ or 16×16 resolution UV deformation map. Given input images on the left, we show results from four different views; the original view and three novel views. Our baseline gets the overall color cues right but cannot add texture details, especially for occluded regions, and renders mostly with flat texture on the body. Our losses help to add detailed patterns in our final renderings. We show details in the last two columns to better visualize how our losses aids in improving overall texture. For the yellow bird, in the final rendering, our method adds detailed fur-like patterns on the body and our baseline falsely add's some black texture on the neck but it gets eyes better.