Supplementary Document for CLIP-Sculptor

Aditya Sanghi[†] Rao Fu[‡] Vivian Liu[§] Karl D.D. Willis[†] Hooman Shayani[†]
Amir H. Khasahmadi[†] Srinath Sridhar[‡] Daniel Ritchie[‡]
Autodesk Research[†] Brown University[‡] Columbia University[§]

Contents

1. More Qualitative Results

Figure 1 shows more qualitative results on ShapeNet13 [1]. In Figure 2 we extend our method, CLIP-Sculptor, to ShapeNet55. It can be seen that our method can generate diverse 3D shapes from different categories, subcategories, common names, and objects with semantic attributes.

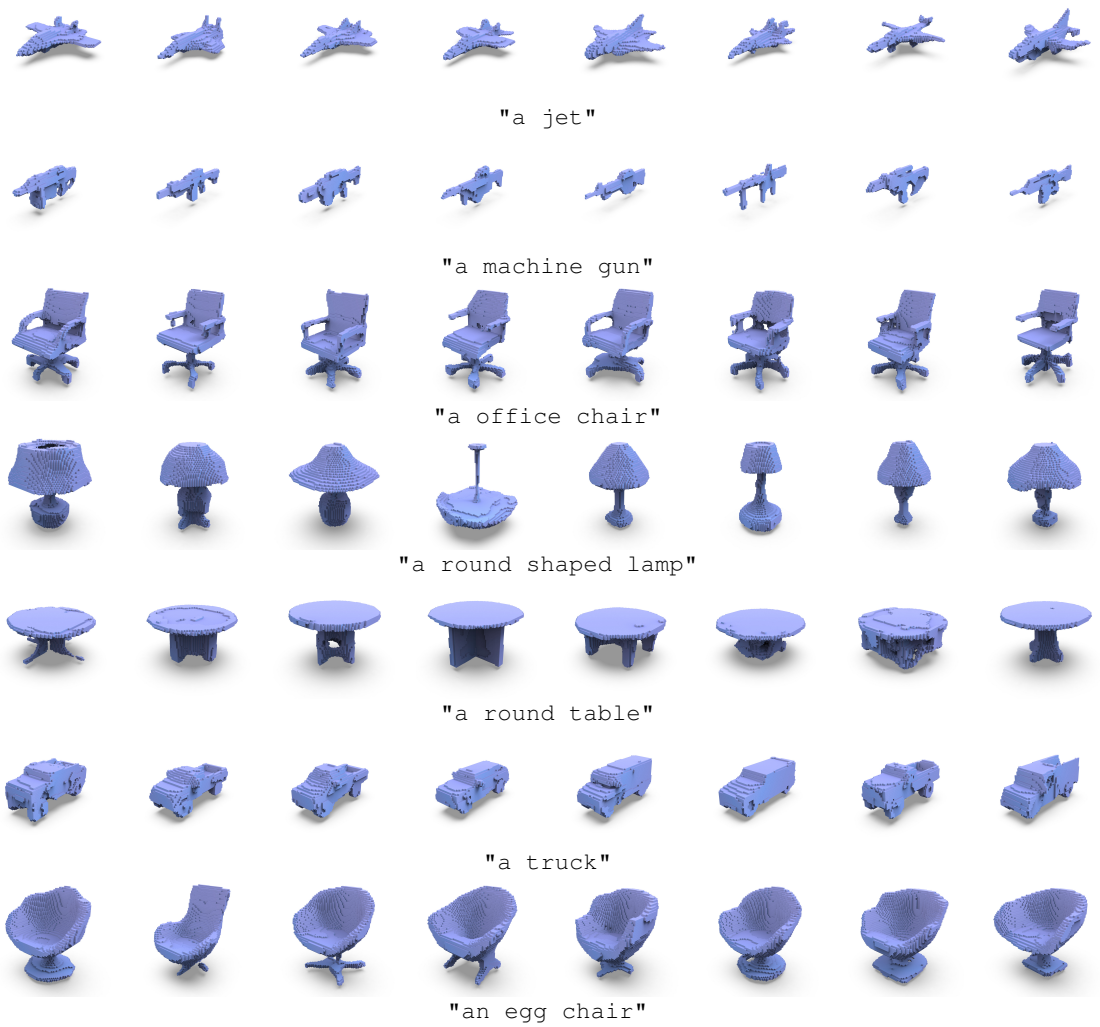

Figure 1. Multiple generated 3D shapes by CLIP-Sculptor using different text inputs. The text inputs are (sub-)category names of ShapeNet13 and phrases with semantic attributes.

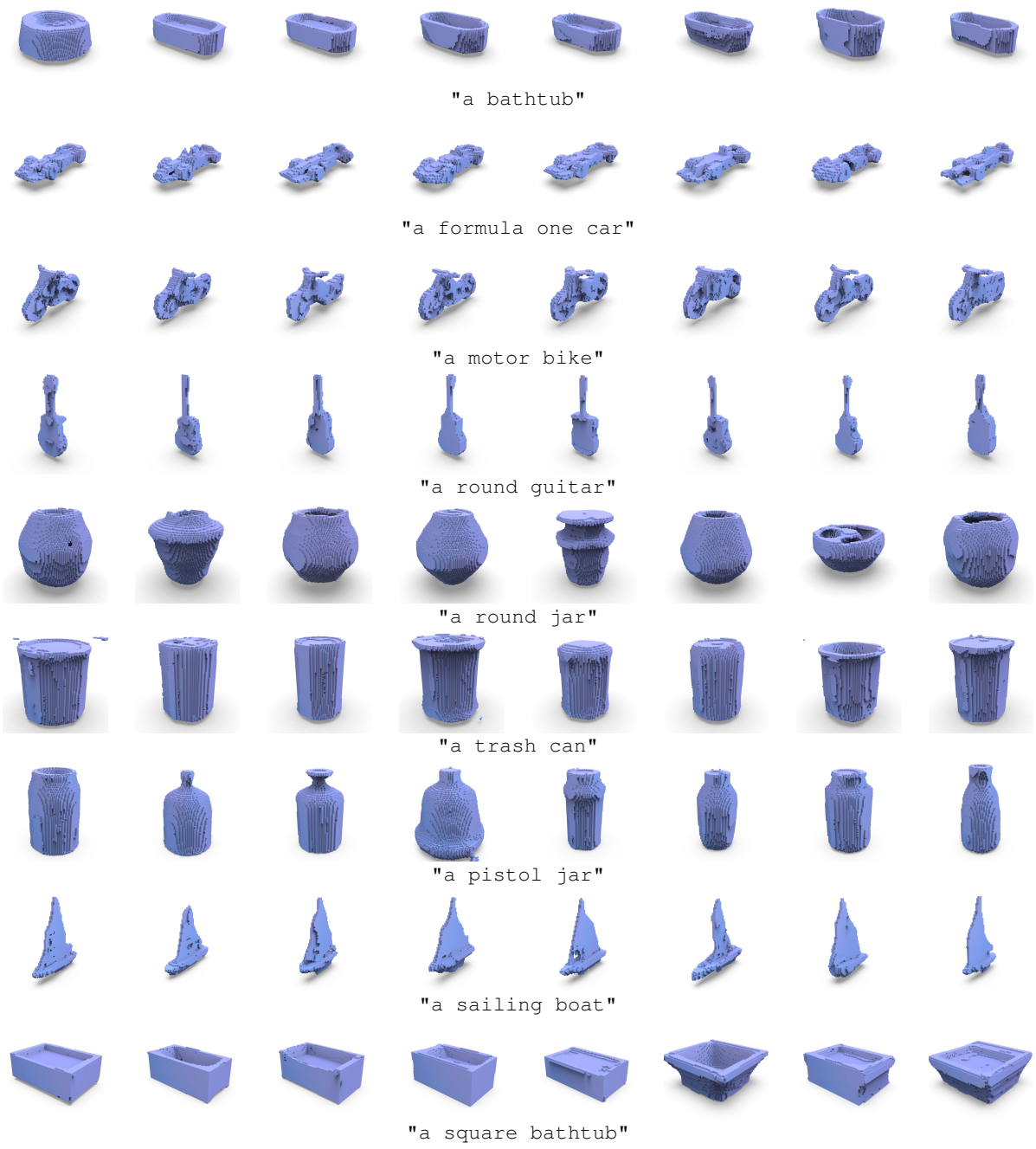

Figure 2. Multiple generated 3D shapes by CLIP-Sculptor using different text inputs. The text inputs are (sub-)category names of ShapeNet55 and phrases with semantic attributes.

2. Visual Results for More Descriptive Texts

In this section, we provide qualitative results for shapes generated by more descriptive texts in Figure 3. We investigate text which is longer and more descriptive in nature (row 1 and 2). We also add queries which have texture or material information while also describing the semantic nature of the object (row 3). Finally, we also investigate verbose text with shape attribute information (row 4). Our method manages to generate plausible shapes even when given more descriptive texts. However, we believe there is room for improvement and that combining our zero-shot method with a few supervised descriptive texts would improve the results significantly. Moreover, too have texture and material for these shapes, we would need to integrate a texture network. We leave these extensions to future work.

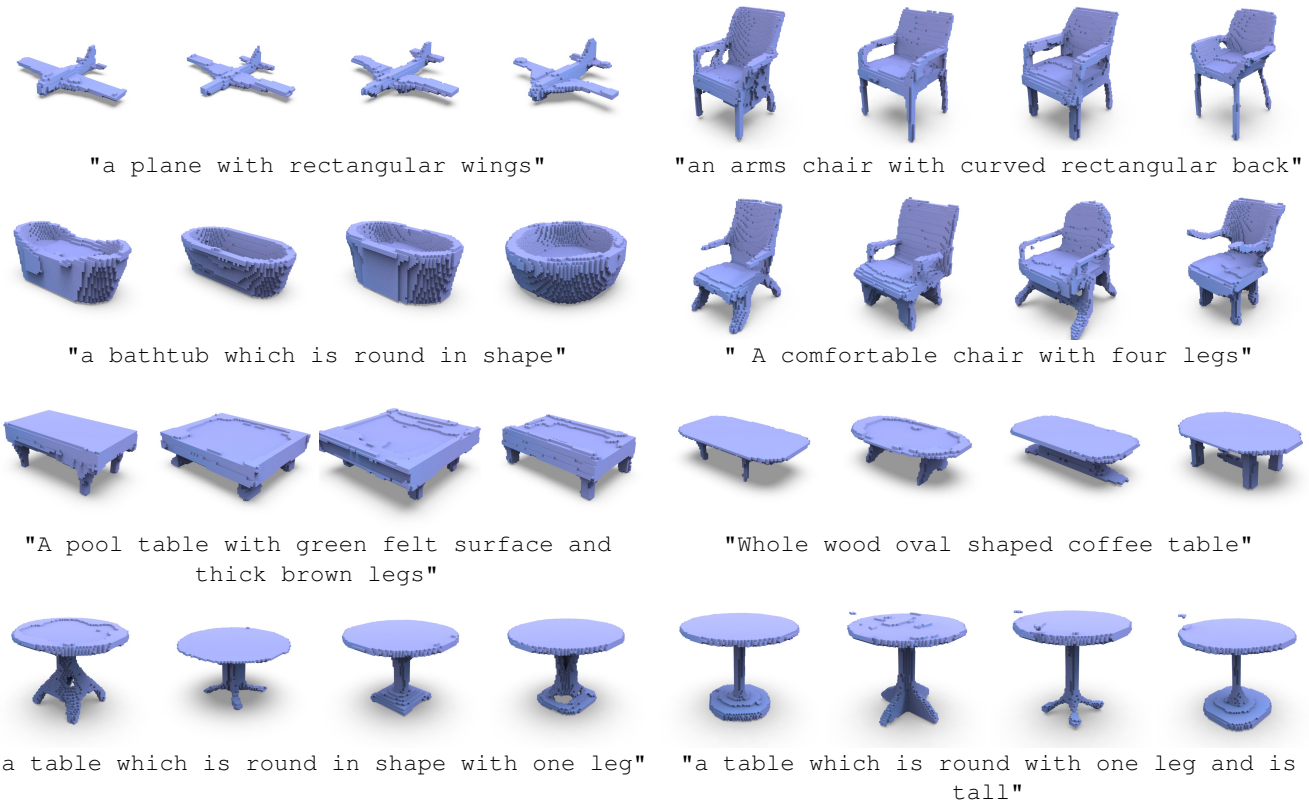

Figure 3. Shapes generated from more descriptive text prompts. CLIP-Sculptor is able to generate diverse and high-fidelity shapes corresponding to these descriptive text prompts.

3. Preliminary Investigation on Extending to 128^3 Voxel Grid and Implicit Representation

We also present initial results where we replaced the 64^3 VQ-VAE with a 128^3 VQ-VAE and an implicit VQ-VAE. We use the same settings for the fine transformer as for the 64^3 VQ-VAE. Figure 4 provides qualitative results on shapes represented with 128^3 voxel grid and implicit fields.

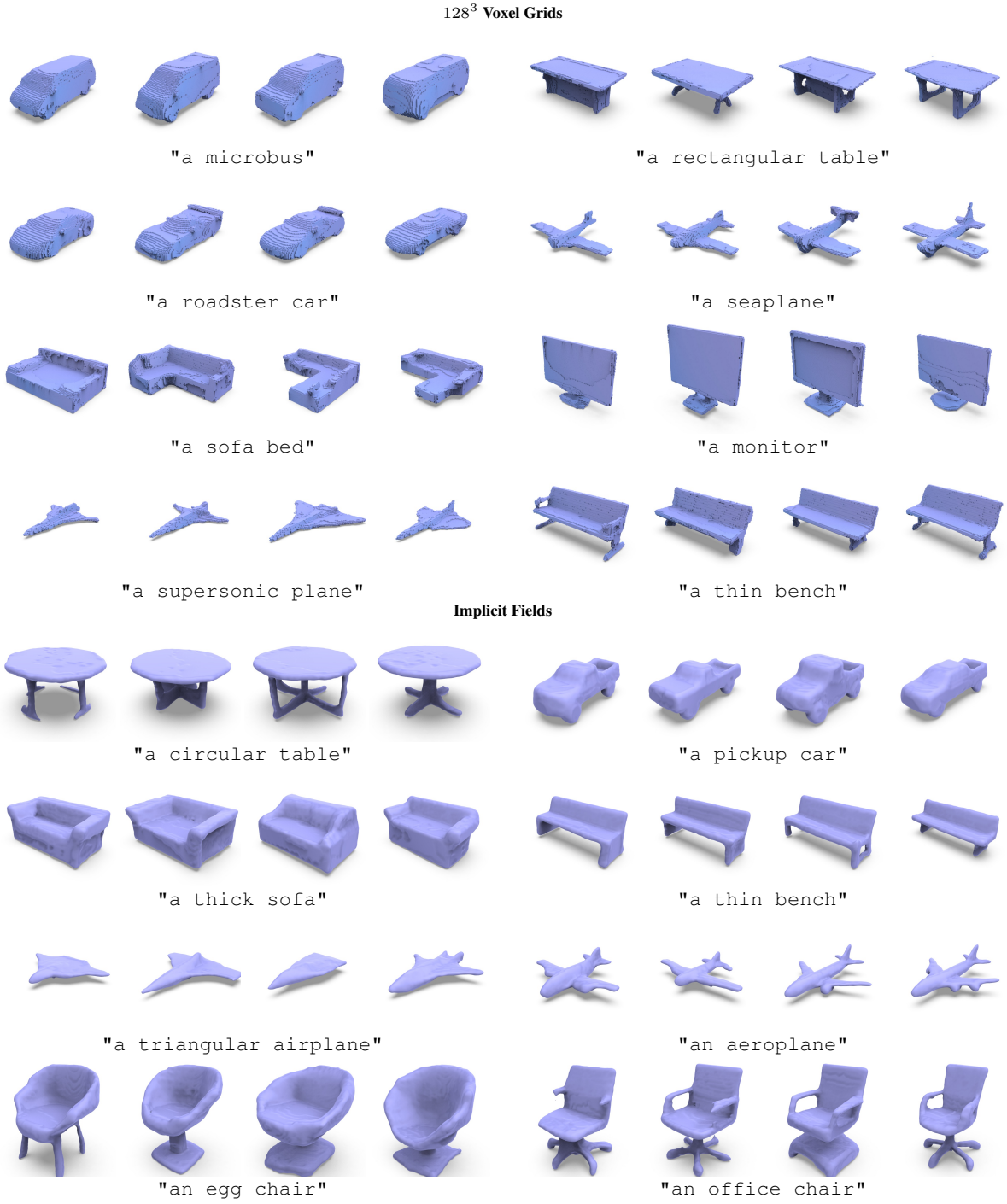

Figure 4. Shapes represented by 128^3 voxel grids and implicit fields. CLIP-Sculptor is able to generate diverse and high-fidelity shapes corresponding to these descriptive text prompts.

4. Ablations on Voxel VQ-VAE

We compare different design decisions for the VQ-VAE at 32^3 resolution using Mean-Square Error (MSE) and Intersection Over Union (IOU) metrics. The results are shown in Table 1. In the first 4 rows, we investigate if the VQ-Autoencoder is sensitive to the codebook loss hyperparameter β and find that the quality of reconstruction is not affected to great degree when we vary it 0.1 to 1. Next, we evaluate in the next 2 rows if the size of codebook embedding has an effect on reconstruction quality and also find minimal change. Finally, we add residual connections to the decoder and encoder and find that this significantly improves the reconstruction quality. For all the experiments with VQ-VAE, we use the last row of Table 1 as hyperparameters. Moreover, we use the exact settings for 64^3 VQ-VAE, with the addition of another ResNet block to the encoder and the decoder.

beta	*emb dims*	*encoder*	*decoder*	**IOU↑**	**MSE↓**
0.1	64	VoxEnc	VoxEnc	0.8876	0.005740
0.25	64	VoxEnc	VoxEnc	0.8872	0.005808
0.50	64	VoxEnc	VoxEnc	0.8856	0.005918
1.0	64	VoxEnc	VoxEnc	0.8831	0.006081
0.25	32	VoxEnc	VoxEnc	0.8856	0.005898
0.25	128	VoxEnc	VoxEnc	0.8820	0.006092
0.1	64	Res-VoxEnc	Res-VoxEnc	0.9148	0.004333

Table 1. Different hyperparameters for stage 1 VQ-VAE at 32^3 resolution

5. Category-wise Accuracy Results

In this section, we report the category-wise accuracy results in Table 2. We compare our method with all multi-shape generation methods as in Table 2 (main paper). It can be seen that our method generates accurate shapes across most categories in ShapeNet. Our method especially performs well on categories with less data such as Phone, Speaker, and Boat. In the categories Chair and Sofa, our method underperforms slightly which we attribute to classifier errors (CLIP-Forge [9] classifier gets 93% accuracy) as some classes are semantically close and there is mislabelling of data within the ShapeNet dataset (some Sofa shapes are present in Chair category).

Method	Airplane	Bench	Cabinet	Car	Chair	Monitor	Lamp	Speaker	Gun	Sofa	Table	Phone	Boat
CF-G	67.03	48.21	57.03	71.48	90.75	67.63	79.36	43.47	34.60	83.06	75.16	20.21	35.42
CF-TG	80.78	54.69	63.87	78.42	**92.55**	75.67	84.17	46.02	44.20	85.69	77.03	27.92	40.28
CF-CG	86.25	56.70	66.21	81.74	**92.55**	76.79	86.04	42.61	51.34	**87.83**	78.13	32.5	42.71
ZS-ASDF	21.72	25.89	7.42	25.59	84.75	12.5	59.79	39.49	5.36	71.38	52.81	47.08	28.99
CS-constant	**100.0**	56.92	58.98	96.48	91.11	81.70	**98.33**	**81.53**	**97.99**	73.20	**92.34**	78.13	98.26
CS-sqrt	**100.0**	56.03	**66.99**	96.68	92.31	85.05	97.08	70.17	97.77	75.66	**92.34**	79.58	97.92
CS-linear	**100.0**	**57.59**	64.65	**97.07**	91.71	**85.50**	98.13	76.14	97.54	76.81	91.88	**79.79**	98.78
CS-cosine	**100.0**	57.14	66.40	96.77	91.83	84.60	96.67	78.13	97.32	74.86	92.03	78.13	**99.13**

Table 2. Category-wise accuracy results

.

CLIP-Model	**FID↓**	**Acc↑**
ViT-B/32	1821.78	86.59
ViT-B/16	2034.60	87.17
ViT-L/14	1771.85	86.57
ViT-L/14 + DALLE-2 prior	1882.12	81.82

Table 3. Effect of different CLIP architecture on FID and Acc metrics

6. Results with Other CLIP Models

In this section, we investigate other CLIP models which are primarily larger in model size. The quantitative results are shown in Table 3 whereas qualitative results are shown in Figure 5. We use the same hyperparameters as mentioned earlier. We use the constant annealing scheme to compare these models. We find that for all the models the results are quite similar, this may be due to all CLIP models being trained on the same amount of data. Moreover, we do not optimize the hyperparameters for these models which could be a contributing factor.

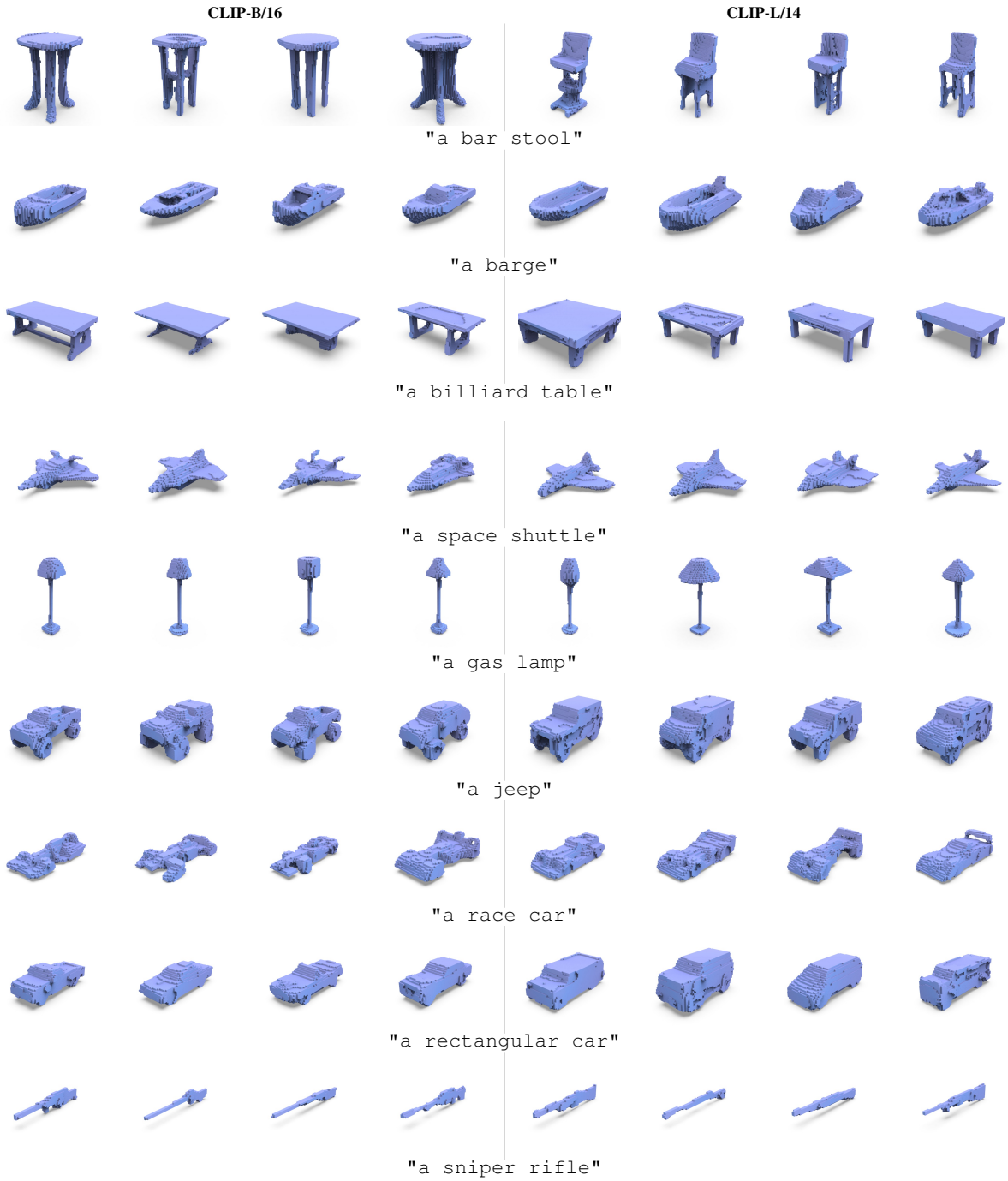

Figure 5. Shapes generated by CLIP-Sculptor with other CLIP models. The results are diverse and of high-fidelity, which shows CLIP-Sculptor is compatible with larger CLIP models.

7. Preliminary Experiments with DALLE-2 Prior

Our method can readily be integrated with the DALL-E2 prior [8]. We use the implementation in https://github.com/lucidrains/DALLE2-pytorch and use the trained models by LAION. The DALL-E2 prior network takes text embeddings from the CLIP model ViT-L/14 as input and outputs image embeddings. As we train our network on image embeddings, intuitively, using the DALL-E2 prior during test time should help. However, our initial investigation shows that the DALL-E2 prior underperforms as shown in Table 3. There could be many reasons for such an unexpected result. One reason could be that the DALL-E2 prior is trained on natural images whereas our method uses renderings. Another factor could be that the addition of noise in our training may have affected the DALL-E2 inference time results. Alternatively, hyperparameter selection could also be a major factor for such an unexpected result.

8. Comparison with CLIP-Forge Supervised Setting

In this experiment, we compare our method to supervised methods as described in CLIP-Forge [9]. We do not use any supervised data to train our method and use the text queries provided in [9] as the comparison metric. We use the same baselines as provided in CLIP-Forge and report the results in Table 4. We use the constant annealing scheme for this experiment. It can be seen that supervised methods struggle to perform well across all categories of ShapeNet as they have labeled data on only few categories.

method	**FID**↓	**Acc.**↑
text2shape-CMA [2]	16078.05	4.27
text2shape-supervised [2]	14881.96	6.84
CLIP-Forge	2425.25	83.33
CLIP-Sculptor (ours)	1821.78	86.59

Table 4. Comparing CLIP-Sculptor with supervised models using the text2shape dataset.

9. Implementation Details

We use the Adam Optimizer [6] with a learning rate of 1e-4 for all stages of training. For both the 32^3 and 64^3 VQ-VAE, we apply the ResNet architecture on a convolutional encoder and decoder. We also use a codebook size of 512 where each embedding dimension is of size 64. We choose a grid size of 4^3 for the 32^3 VQ-VAE and a grid size of 8^3 for the 64^3 VQ-VAE . For the Stage 2 and Stage 3 transformers, we use a bidirectional transformer with 8 attention blocks, 8 attention heads, and a token size of 256. For Stage 2, we train the network for 250 epochs with a batch size of 32, whereas for Stage 3 we train for 300 epochs. We do not use any dropout in the transformers. For all the experiments, we use 24 renderings [3] of ShapeNet13. We run both the coarse and fine transformer for 13 steps during inference. For all results in main paper, we use the CLIP ViT-B/32 model, which uses a transformer based encoder.

For results in Table 2 (main paper), all annealing schemes use the starting scale parameters as 4.05 with 13 sampling steps. We use $\gamma = 1.2$ and 3 layers of mapping network. We use a dropout rate, ρ, of CLIP image features at 5% and use Step-Unrolled Training. In Figure 3, we use the same parameters except we generate all figures using the sqrt annealing scheme. Note for these experiments we use the best seed similar to the protocol used in CLIP-Forge. In the rest of the paper, we average over 3 seeds. We also calculate all the results on the same classifier and resolution as CLIP-Forge. We refer the readers to CLIP-Forge appendix [9] for the text queries used in comparisons (Table 2 main paper).

The hyperparameters used for Table 2 (main paper) are based on the ablation study section 4.2 (main paper). We start with no classifier free guidance, noise parameter $\gamma = 0$, and a mapping network set to 3 layers in Table 3(a) (main paper). We find that using $\gamma = 1.2$ as the noise parameter gives the best result and we use that for rest of the experiments. In Table 3(b) (main paper), we vary the number of layers, while keeping the above hyperparameters constant. We find 3 layers to be optimal and we use that for the rest of the experiments. In Table 4 (main paper), we investigate using different dropout rates (ρ) of the conditional embedding. For the Step-Unrolled Training (SUT) experiment, we use dropout ρ at 5%.

Our first baseline is CLIP-Forge [9], for which we use the trained model provided in https://github.com/AutodeskAILab/Clip-Forge. In the case of the clipped Gaussian (CF-CG) distribution, we clip the samples between -1 to 1. For the truncated Gaussian (CF-TG) distribution, we sample from $\mathcal{N}(0, 0.5)$. In Figure 3, we compare with our method using the Gaussian distribution.

For Zero-Shot AutoSDF(ZS-ASDF) [7], we use the pre-trained P-VQ-VAE and random transformer provided in https://github.com/yccyenchicheng/AutoSDF/. For the language-guided generation part, we replace the BERT model with the CLIP model, and train the conditional model with CLIP image features until the model converges. At inference time, we use the CLIP text features. For Dreamfield [4], we use the high-quality setting. For CLIP-Mesh [5], we use the default setting.

References

[1] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing 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, 2015. cite arxiv:1512.03012. 2

[2] Kevin Chen, Christopher B Choy, Manolis Savva, Angel X Chang, Thomas Funkhouser, and Silvio Savarese. Text2shape: Generating shapes from natural language by learning joint embeddings. In *Asian Conference on Computer Vision*, pages 100–116. Springer, 2018. 8

[3] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016. 8

[4] Ajay Jain, Ben Mildenhall, Jonathan T Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 867–876, 2022. 9

[5] Nasir Khalid, Tianhao Xie, Eugene Belilovsky, and Tiberiu Popa. Clip-mesh: Generating textured meshes from text using pretrained image-text models. *ACM Transactions on Graphics (TOG), Proc. SIGGRAPH Asia*, 2022. 9

[6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 8

[7] Paritosh Mittal, Yen-Chi Cheng, Maneesh Singh, and Shubham Tulsiani. Autosdf: Shape priors for 3d completion, reconstruction and generation. *arXiv preprint arXiv:2203.09516*, 2022. 9

[8] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022. 8

[9] Aditya Sanghi, Hang Chu, Joseph G Lambourne, Ye Wang, Chin-Yi Cheng, Marco Fumero, and Kamal Rahimi Malekshan. Clip-forge: Towards zero-shot text-to-shape generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18603–18613, 2022. 6, 8