Robust 3D Shape Reconstruction in Zero-Shot from a Single Image in the Wild
— Supplementary Material —

Junhyeong Cho[1] Kim Youwang[2] Hunmin Yang[1,3] Tae-Hyun Oh[2,4]
[1]ADD [2]Department of Electrical Engineering, POSTECH
[3]Department of Mechanical Engineering, KAIST [4]School of Computing, KAIST
https://ZeroShape-W.github.io

In this supplementary material, we provide more implementation details (Sec. A), training and evaluation details (Sec. B), analyses (Sec. C), discussion (Sec. D), and broader impacts with ethics considerations (Sec. E), which are not included in the main paper due to its space constraints.

A. Implementation Details

A.1. Model Architecture

Pixel-level regression. We employ the backbone of DPT-Hybrid [20], which is composed of 12 transformer layers and 4 fusion layers. From the global feature map $\mathbf{X}_G$, we estimate parameters for camera intrinsics K using 4 convolutional layers, 2 ReLU activation layers, an average pooling layer, and a linear layer. From the fine-grained feature map $\mathbf{X}_F$, we regress the depth map $\mathbf{M}_D$, visible mask $\mathbf{M}_V$, and occluder mask $\mathbf{M}_O$. Specficially, the depth map $\mathbf{M}_D$ is regressed using 3 convolutional layers, 2 ReLU activation layers, and a bilinear interpolation layer. The visible mask $\mathbf{M}_V$ and occluder mask $\mathbf{M}_O$ are regressed through 2 convolutional layers, 1 ReLU activation layer, and a bilinear interpolation layer, respectively. The scaling factor γ and shift factor β used for the affine transformation are estimated via a linear layer and a SiLU activation layer.

3D point-wise regression. From the concatenated output of the visible 3D shape $\mathbf{S}_V$ and occluder mask $\mathbf{M}_O$, we extract features using the backbone of ResNet50 [10], augmented with additional 9 convolutional layers and 4 ReLU activation layers. These extracted features serve as keys and values in 2 cross-attention layers, where sine-cosine positional embeddings are incorporated to preserve spatial information. The cross-attention layers process queries constructed by the concatenation of point embeddings and text embeddings. The point embeddings are derived from 3D query points using 2 linear layers, while the text embeddings are obtained from CLIP text embeddings [19] projected by a linear layer. The outputs from the cross-attention layers are used to regress occupancy values of the 3D query points through 9 linear layers with skip connections and 8 Softplus activation layers.

A.2. Data Synthesis

The pseudocode for our data synthesis pipeline is outlined in Algorithm 1.

Rendering. We utilize the 3D shape renderings provided by ZeroShape [11]. These renderings were produced using Blender [3] with various camera configurations. Specifically, focal lengths were varied between 30mm and 70mm for a 35mm sensor size equivalent. Camera distances and LookAt points were also randomized, with elevation angles ranging from $5°$ to $65°$. These renderings, produced at a resolution of 600×600 pixels, were cropped around the center of objects and resized to 224×224 pixels.

Object appearance diversification. We utilize ControlNet [26] to simulate diverse visual variations in object appearances, excluding those with high-resolution texture maps (*e.g.*, several objects from Objaverse [5]). To be specific, variations are generated by a textual condition "a `[color] [material] [object]`", where `[color]` and `[material]` are randomly selected from pre-defined color list $\mathcal{L}_c$ and material list $\mathcal{L}_m$, respectively. The color list $\mathcal{L}_c$ includes "red", "pink", "orange", "yellow", "green", "blue", "purple", "brown", "white", "black", "gray", and an empty string. The material list $\mathcal{L}_m$ contains "metal", "wood", "plastic", "ceramic", "stone", "rubber", "leather", and an empty string. To utilize more plausible and diverse textual conditions according to each object rendering, one can leverage suggestions from LMMs [14].

Initial guidance. As described in Section 4.2 of the main paper, we leveraged initial guidance to reduce silhouette distortion of objects. To be specific, when we utilized ControlNet [26], we set 20 steps in the DDIM sampler [22] and injected the guidance at step 8. This guidance effectively forces the conditional diffusion model to precisely adhere to input spatial condition.

Filtering synthesized object images. While the initial guidance substantially aids in preserving the object silhouette as specified in the spatial condition, minor disparities may still exist between the silhouettes in the synthesized images and the original silhouettes from the renderings. To

Algorithm 1 Pseudocode of our data synthesis pipeline

Requirement: object renderer $\mathcal{R}(\cdot)$, guidance perturbator $\mathcal{P}(\cdot)$, random seed generator $\mathcal{G}(\cdot)$, diffusion model for object diversification $\texttt{DM}_{\text{obj}}(\cdot)$, diffusion model for background outpainting $\texttt{DM}_{\text{bg}}(\cdot)$

Input: 3D objects $\{\mathcal{O}_i\}_{i=1}^{K}$, number of camera views for rendering 3D objects $\{N_i\}_{i=1}^{K}$, color list $\mathcal{L}_c$, material list $\mathcal{L}_m$, scene list $\mathcal{L}_s$, IoU filtering threshold κ

Output: $\langle$3D object, camera parameters, depth map, mask, image$\rangle$−dataset

```
# C: camera parameters, D: depth map, M: mask, I: image
{C_{i,j}}_{i=1,j=1}^{K,N_i}, {D_{i,j}}_{i=1,j=1}^{K,N_i}, {M_{i,j}}_{i=1,j=1}^{K,N_i}, {I_{i,j}}_{i=1,j=1}^{K,N_i} ← R({O_i}_{i=1}^{K}, {N_i}_{i=1}^{K})   ▷ render 3D objects
dataset ← []
for i = 1, 2, ..., K do
    for j = 1, 2, ..., N_i do
        guide ← P(I_{i,j})                                   ▷ perturb an initial guidance
        while True do
            seed ← G()                                       ▷ set random seed
            [color], [material] ← sample(L_c, L_m, seed)     ▷ randomly select words
            [object] ← retrieve_category(O_i)                ▷ retrieve 3D object category
            txt ← "a [color] [material] [object]"
            fg_img ← DM_obj(D_{i,j}, guide, txt, seed)       ▷ simulate object appearance
            if not is_filtered(fg_img, M_{i,j}, κ) then      ▷ filter an image with a threshold κ
                break
            end if
        end while
        seed ← G()                                           ▷ set random seed
        [scene] ← sample(L_s, seed)                          ▷ randomly select a word
        txt ← "a [object] in the [scene]"
        fg_bg_img ← DM_bg(M_{i,j}, fg_img, txt, seed)        ▷ simulate background
        Add (O_i, C_{i,j}, D_{i,j}, M_{i,j}, fg_bg_img) to dataset
    end for
end for
return dataset
```

address this, we filter out images if the intersection-over-union (IoU) between the synthesized and original silhouettes is below 0.95. The silhouettes in the synthesized images are estimated by extracting the foreground objects in the images. We can simply extract the foreground objects using a threshold, as the initial guidance results in images that have nearly-white backgrounds. Specifically, we convert each synthesized RGB image to grayscale, and then consider pixels with values between 250 and 255 as background. This straightforward process allows us to accurately approximate the foreground silhouette in most cases.

Background diversification. We simulate diverse backgrounds using an object-aware background outpainting model [6] with a textual condition "a `[object]` in the `[scene]`", where `[scene]` is randomly selected from pre-defined scene list $\mathcal{L}_s$. As described in Section 4.3 of the main paper, we use scene categories from [23, 27] as the scene list $\mathcal{L}_s$, which contains more than 700 categories. To utilize more plausible and diverse textual conditions according to each foreground object, one can leverage suggestions from LMMs [14].

B. Training and Evaluation Details

We initialize our model with ZeroShape [11] model weights for shared components such as DPT-Hybrid backbone [20] and cross-attention layers. We first pre-train our pixel-level regression components for 10 epochs, using the Adam optimizer [12] with a learning rate of 10^{-5}, a batch size of 80, a weight decay of 0.05, and momentum parameters of $(0.9, 0.95)$. This process takes approximately 2 days on 4 RTX 3090 GPUs. Then, we train our entire model for 15 epochs, using the same optimizer with a learning rate of 10^{-5} for 3D point-wise regression components and 10^{-6} for pre-trained pixel-level regression components, a batch size of 80, a weight decay of 0.05 and momentum parameters of $(0.9, 0.95)$. This process takes approximately 4 days on 4 RTX 3090 GPUs.

Loss coefficients. Let $\mathcal{L}_c$ represent the camera intrinsics loss, $\mathcal{L}_d$ the depth loss using ground-truth depth values, $\mathcal{L}_d^{\text{aux}}$ the auxiliary depth loss using depth values estimated from Depth Anything V2 [25], $\mathcal{L}_m^{\text{vis}}$ the visible mask loss, $\mathcal{L}_m^{\text{occ}}$ the occluder mask loss, and $\mathcal{L}_o$ the occupancy loss.

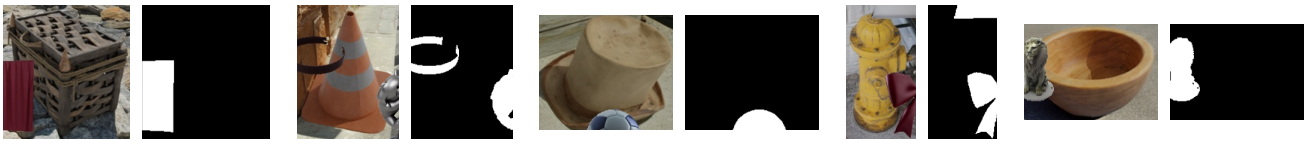

Figure B1. Examples of Copy-Paste augmentation. In this visualization, we provide augmented training samples with occluder masks.

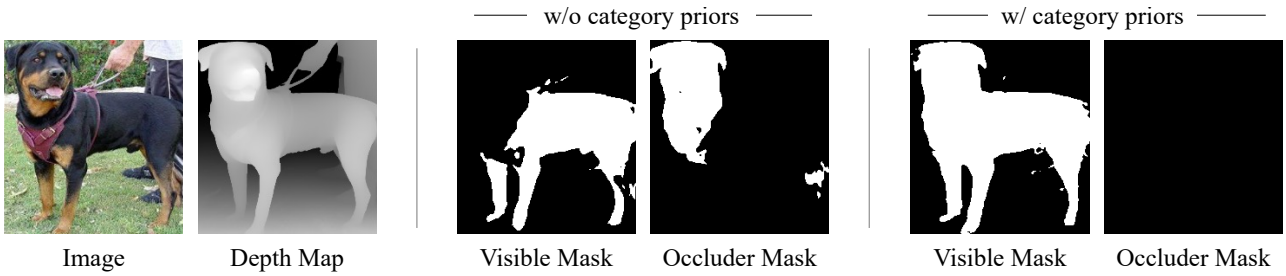

Figure C1. Effect of category priors on mask regression. Without utilizing the priors, the regression of visible and occluder masks appears to rely on depth values. By leveraging the priors, the regression is enhanced with semantic understanding.

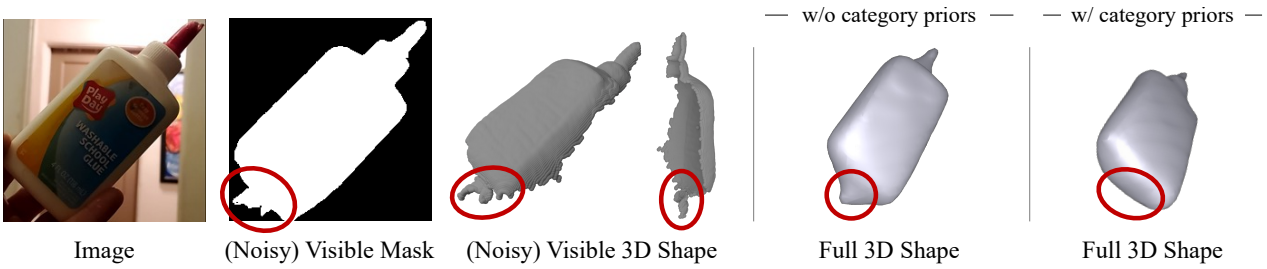

Figure C2. Effect of category priors on occupancy regression. Without utilizing the priors, it is challenging to distinguish the object from its background in the noisy visible 3D shape. By leveraging the priors, the regression is enhanced with learned 3D shape prior specific to the category, leading to more accurate results. We highlight regions with red circles.

The loss $\mathcal{L}$ used for training the pixel-level regression components is computed as follows:

$$\mathcal{L} = \lambda_c \mathcal{L}_c + \lambda_d \mathcal{L}_d + \lambda_d^{\text{aux}} \mathcal{L}_d^{\text{aux}} + \lambda_m^{\text{vis}} \mathcal{L}_m^{\text{vis}} + \lambda_m^{\text{occ}} \mathcal{L}_m^{\text{occ}}, \quad (1)$$

where $\lambda_c = 10, \lambda_d = \lambda_m^{\text{vis}} = \lambda_m^{\text{occ}} = 1$, and $\lambda_d^{\text{aux}} = 0.1$. The loss $\mathcal{L}'$ used for training our entire model is computed as follows:

$$\mathcal{L}' = \mathcal{L} + \lambda_o \mathcal{L}_o, \quad (2)$$

where $\lambda_o = 1$. To compute the occupancy loss $\mathcal{L}_o$, we randomly sample 4096 query points at every iteration.

Copy-Paste augmentation. As shown in Figure B1, we apply the augmentation by randomly selecting objects synthesized with focal lengths similar to the corresponding training sample. In each training iteration, we randomly choose from 0 to 2 occluders, resize them within a scale range of 0.4 to 0.6, and put them onto the training sample.

Category priors. We use CLIP text embeddings [19] to learn category-specific priors with ground-truth object categories from Objaverse-LVIS [5] and ShapeNetCore.v2 [2].

Evaluation. For the quantitative evaluation of our model, we compute Chamfer Distance (CD) and F-Score (FS). Regarding FS@τ, we set the distance threshold τ to 0.05. For the evaluation, we need to convert implicit 3D shapes into explicit meshes and then sample points from their surfaces. To obtain explicit meshes, we apply Marching Cubes [15] algorithm, using sampled values from a 128^3 spatial grid.

Estimation of object category. To incorporate category-specific priors, we optionally estimate the object category of an object-centric image using a VLM [14]. We use the following prompt: *What is the salient object in this image? The object should occupy the majority of the image. Please provide the category name of the salient object in the format: "[object]", where "[object]" is the specific category name (e.g., "chair", "bed", "sofa", "table").*

C. More Analyses

Category priors for regressing masks. When category priors are not utilized, our model sometimes incorrectly splits a single entity (*e.g.*, a dog) into visible region (*e.g.*, the dog's body) and occluder region (*e.g.*, the dog's head) based on depth values, as shown in Figure C1. We suspect this issue arises due to the lack of semantic understanding. To mitigate this, one may incorporate semantic priors by estimating object categories using a vision-language model.

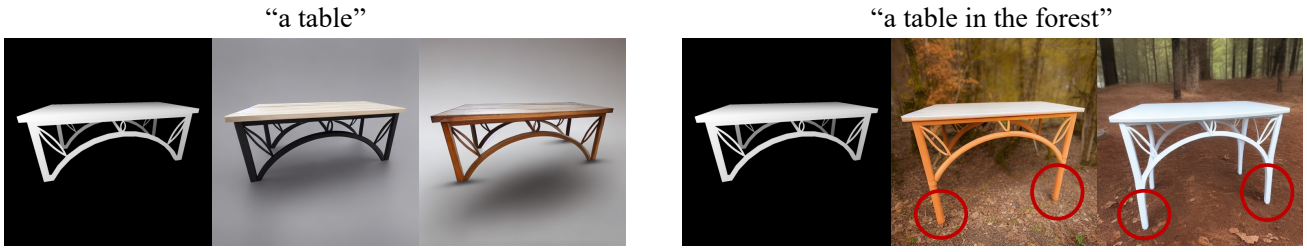

Figure D1. Observations from pre-trained conditional generative models [13, 16, 18, 26]. This phenomenon is mainly attributed to their pre-training procedure; they were trained with depth maps which contain both foreground and background information. When we use depth maps rendered from 3D objects, synthesizing backgrounds violates input spatial conditions.

Category priors for regressing occupancy values. Visible 3D shapes may include background geometries due to noisy estimations of visible masks. In such cases, it is challenging to accurately regress occupancy values for the corresponding objects. To be specific, distinguishing a salient object from its background is difficult, because a visible 3D shape only contains the xyz-coordinates of each pixel (*i.e.*, pixel-aligned point cloud) without any supplemental visual cues (*e.g.*, RGB color). To address this issue, one may leverage category priors for regressing the occupancy values, as shown in Figure C2.

D. Discussion

Why synthesizing images in two steps? Our data synthesis pipeline first generates foreground objects and then outpaints their backgrounds. A more straightforward alternative would be to synthesize the entire image at once using conditional generative models such as ControlNet [26]. However, as shown in Figure D1, when these models are forced to strictly follow the input spatial conditions, they often produce monotonous backgrounds due to the lack of background information in the conditions. On the other hand, if we encourage the models to diversify backgrounds using textual conditions, they tend to create more varied backgrounds at the cost of violating the input spatial conditions. To resolve these challenges, we first diversify object appearances while adhering to the input spatial conditions, and then use an object-aware background outpainting model [6], specifically fine-tuned to prevent distortion of object silhouettes while generating the backgrounds.

Various approaches for occlusion-aware reconstruction. Probabilistic methods (*e.g.*, PT43D [24]) are effective for handling heavily occluded objects by generating multiple plausible 3D shapes. However, there are trade-offs between (i) accuracy and sample diversity [21], and (ii) accuracy and efficiency [9]. In comparison, regression-based methods can efficiently produce competitive results for small occlusions, but they become sub-optimal and often impractical when tackling highly occluded objects.

E. Broader Impacts & Ethics Considerations

Our data synthesis pipeline utilizes 3D object collections and conditional generative models. To avoid conflicts, one should carefully follow their usage rights, licenses and permissions. Also, one should be aware that generative models might reflect biases inherent in their training data [7], and object images in the training data might also be biased [4]. Furthermore, one should keep in mind that generative models might expose their training data [1]. To avoid data privacy issues, one can use erasing methods [8, 17] capable of removing unwanted concepts from generative models.

References

[1] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting Training Data from Large Language Models. In *30th USENIX Security Symposium (USENIX Security 21)*, 2021. 4

[2] 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. *arXiv preprint arXiv:1512.03012*, 2015. 3

[3] Blender Online Community. Blender - a 3D modelling and rendering package. *Blender Foundation*, 2018. 1

[4] Terrance de Vries, Ishan Misra, Changhan Wang, and Laurens van der Maaten. Does Object Recognition Work for Everyone? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2019. 4

[5] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A Universe of Annotated 3D Objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 3

[6] Amir Erfan Eshratifar, Joao V. B. Soares, Kapil Thadani, Shaunak Mishra, Mikhail Kuznetsov, Yueh-Ning Ku, and Paloma de Juan. Salient Object-Aware Background Genera-

tion using Text-Guided Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2024. 2, 4

[7] Patrick Esser, Robin Rombach, and Björn Ommer. A Note on Data Biases in Generative Models. In *NeurIPS 2020 Workshop on Machine Learning for Creativity and Design*, 2020. 4

[8] Rohit Gandikota, Joanna Materzynska, Jaden Fiotto-Kaufman, and David Bau. Erasing Concepts from Diffusion Models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 4

[9] Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, Muhammad Shahzad, Wen Yang, Richard Bamler, and Xiao Xiang Zhu. A survey of uncertainty in deep neural networks. *Artificial Intelligence Review*, 2023. 4

[10] He, Kaiming and Zhang, Xiangyu and Ren, Shaoqing and Sun, Jian. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 1

[11] Zixuan Huang, Stefan Stojanov, Anh Thai, Varun Jampani, and James M. Rehg. ZeroShape: Regression-based Zero-shot Shape Reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 1, 2

[12] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 2

[13] Yuheng Li, Haotian Liu, Qingyang Wu, Fangzhou Mu, Jianwei Yang, Jianfeng Gao, Chunyuan Li, and Yong Jae Lee. GLIGEN: Open-Set Grounded Text-to-Image Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 4

[14] Yanwei Li, Chengyao Wang, and Jiaya Jia. LLaMA-VID: An Image is Worth 2 Tokens in Large Language Models. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 1, 2, 3

[15] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *ACM SIGGRAPH Computer Graphics*, 21(4), 1987. 3

[16] Chong Mou, Xintao Wang, Liangbin Xie, Yanze Wu, Jian Zhang, Zhongang Qi, Ying Shan, and Xiaohu Qie. T2I-Adapter: Learning Adapters to Dig out More Controllable Ability for Text-to-Image Diffusion Models. *arXiv preprint arXiv:2302.08453*, 2023. 4

[17] Zixuan Ni, Longhui Wei, Jiacheng Li, Siliang Tang, Yueting Zhuang, and Qi Tian. Degeneration-Tuning: Using Scrambled Grid shield Unwanted Concepts from Stable Diffusion. In *31st ACM International Conference on Multimedia*, 2023. 4

[18] Can Qin, Shu Zhang, Ning Yu, Yihao Feng, Xinyi Yang, Yingbo Zhou, Huan Wang, Juan Carlos Niebles, Caiming Xiong, Silvio Savarese, Stefano Ermon, Yun Fu, and Ran Xu. UniControl: A Unified Diffusion Model for Controllable Visual Generation In the Wild. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 4

[19] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In *International Conference on Machine Learning (ICML)*, 2021. 1, 3

[20] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision Transformers for Dense Prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 1, 2

[21] Akash Sengupta, Ignas Budvytis, and Roberto Cipolla. Humaniflow: Ancestor-conditioned normalising flows on so(3) manifolds for human pose and shape distribution estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 4

[22] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations (ICLR)*, 2021. 1

[23] Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. SUN database: Large-scale scene recognition from abbey to zoo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010. 2

[24] Yiheng Xiong and Angela Dai. Pt43d: A probabilistic transformer for generating 3d shapes from single highly-ambiguous rgb images. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2024. 4

[25] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth Anything V2. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 2

[26] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding Conditional Control to Text-to-Image Diffusion Models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 1, 4

[27] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 Million Image Database for Scene Recognition. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2018. 2