CLIP-GS: Unifying Vision-Language Representation with 3D Gaussian Splatting

Supplementary Material

1. Implementation details

We employ EVA02-E-14-plus from OpenCLIP [5]. We sample 10,000 points from the 3DGS. During training, we freeze the vision and text encoders in EVA-CLIP-T, only leaving CLIP-GS as the learnable component. The model is optimized with the Adam [6] optimizer with a weight-decay of 0.05. The learning rate is set to 5×10^{-4} for GS-Tokenizer, and 1×10^{-4} for other modules. The model is trained for 5 epochs on the triplets with 8 NVIDIA A6000 GPUs. *e.g.*, CLIP-GS-B with $\sim$ 9 million parameters, converges in approximately 14 hours with batch size = 32 on each A6000 GPU.

2. Additional Experiments

Training from scratch. To avoid the impact of initializing with the point cloud pre-trained weights, we train CLIP-GS and Uni3D from the 2D pretraining model EVA-CLIP, as shown in Tab. 1. CLIP-GS outperforms Uni3D by +3.1 Top1 accuracy, suggesting that the additional 3DGS attributes are effectively utilized. This leads us to reasonably infer that with **larger-scale training data**, 3DGS will demonstrate even greater advantages.

	3D repr	Top1	Top3	Top5
Uni3D	$P\&C$	30.8	51.1	59.84
CLIP-GS	3DGS	33.9	55.6	64.0

Table 1. Training from scratch. $P\&C$ denotes only P and C attributes from 3DGS is used.

Using fewer points. We use fewer gaussian points to stimulating 3D expression capabilities of 3DGS in Tab. 2. CLIP-GS outperforms Uni3D by +2.3 and +3.1 Top1 accuracy when using 5K/2.5K points. This leads us to reasonably infer that with **more complex 3D scenarios**, 3DGS will demonstrate even greater advantages.

	num points	3D repr	Top1	Top3	Top5
Uni3D	5K	$P\&C$	43.5	63.7	71.2
CLIP-GS		3DGS	45.8	65.5	73.9
Uni3D	2.5K	$P\&C$	42.4	62.8	70.2
CLIP-GS		3DGS	45.5	65.0	73.3

Table 2. Using fewer gaussian points. $P\&C$ denotes only P and C attributes from 3DGS is used.

Number of view selected in $\mathcal{L}_{\text{img}}$. We investigate using different numbers of views in $\mathcal{L}_{\text{img}}$. We set K=2, 4, 5, and 8 in $\mathcal{L}_{\text{img}}$ in Tab. 3. It is evident that the model's performance improves as the number of views increases. Using K=8 results in only a 0.1 improvement while increasing the training cost. Therefore, we choose K=5 as the default.

K	**Top1**	**Top3**	**Top5**
2	48.0	70.0	77.0
4	48.3	70.1	77.2
5 (default)	48.5	70.3	77.5
8	48.6	70.6	77.8

Table 3. Number of views (K) in $\mathcal{L}_{\text{img}}$.

Number of view selected in $\mathcal{L}_{\text{img}}$. We use the more advanced Cap3D [10] to provide better captions. In Tab. 4, we compare the captions from BLIP-2 [7] (provided by Openshape [9]) and Cap3D. We also include the results of finetuning Uni3d on the $P\&C$ attributes. Results show that different captions exhibit similar performance, indicating that captions are not the key determining factor.

	Caption	3D repr	Top1	Top3	Top5
Uni3D	Cap3D	$P\&C$	46.9	68.5	75.9
CLIP-GS		3DGS	48.5	70.3	77.5
Uni3D	OpenShape	$P\&C$	46.1	67.7	74.6
CLIP-GS		3DGS	48.0	70.2	77.6

Table 4. 3D shape captions from BLIP-2 and Cap3D. $P\&C$ denotes only P and C attributes from 3DGS is used.

Evaluated on real-world datasets. To validate the generalization ability of CLIP-GS, we collect 11 real-world scenes from the Mip-NeRF-360 dataset [2] (as shown in Fig. 1) and generate the corresponding 3DGS, following the approach outlined in Sec. 3. The zero-shot classification results are shown in Tab. 5.

	Top1	Top3	Top5
Uni3d	9.1	9.1	18.2
CLIP-GS	9.1	36.4	45.5

Table 5. Comparison with Uni3d on Mip-NeRF-360 (real-world scans).

Applying to segmentation tasks. We use Uni3D or CLIP-GS as the backbone and add a segmentation head [8] for 3D segmentation. Both Uni3D and CLIP-GS are trained on CloSe-Di [1] for 10 epochs. The 3DGS generation process

Figure 1. 11 Real-world scans from Mip-NeRF-360. Zoom in for a better view.

follows the approach outlined in Sec. 3. Tab. 6 presents the 3D mIoU results on CloSe-Di.

	mIoU
Uni3d	87.0
CLIP-GS	89.0

Table 6. Comparison with Uni3d on CloSe-Di (segmentation).

3. Excluded and Retained 3D shapes

We filter the 3D shapes in Objaverse [3, 4] and select those with a diversity of colors and textures. We use LLaVA-OneVision-7B to filter out meaningless 3D shapes and remove shapes that feature fewer than five distinct colors. The prompt used for filtering is: "$\{Img\}$ The image is rendered from a 3D model. Is this 3D model meaningless? Answer yes or no without explanation"

In this section, we provide a visual comparison of the retained and excluded 3D shapes, as shown in Fig. 2. We excluded monochromatic, meaningless 3D shapes. *e.g.*, items like the Christmas tree, car, and bucket in the first row of Fig. 2 only contain the contours of the 3D objects, with colors that are single-toned and lack texture information. The retained 3D shapes have colors with intricate texture details, which are difficult for point clouds to depict.

4. Retrieval Results

In Fig. 3, we showcase how CLIP-GS successfully retrieves 3D shapes from text or real-world images. We retrieve the most similar or the Top 2 / Top 3 similar 3D shapes according to the corresponding images or text. CLIP-GS performs well when retrieving real-world images (Fig. 3 top). CLIP-GS has learned the encoding of 3DGS and can align the features of 3DGS well with the image spaces, allowing it to retrieve the most suitable 3D shapes based on the input of one or two images. Moreover, the results indicate that CLIP-GS retrieves reasonable 3D shapes based on text in the query set (Fig. 3 bottom). This retrieval is not limited by category, and CLIP-GS can retrieve reasonable expressions (*e.g.*, smiling), textures (*e.g.*, antique), and other information that is easily lost in point cloud representations.

References

[1] Dimitrije Antić, Garvita Tiwari, Batuhan Ozcomlekci, Riccardo Marin, and Gerard Pons-Moll. Close: A 3d clothing segmentation dataset and model. In *2024 international conference on 3D vision (3DV)*, pages 591–601. IEEE, 2024. 1

[2] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5470–5479, 2022. 1

[3] 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*, pages 13142–13153, 2023. 2

[4] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[5] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, et al. Openclip, july 2021. *If you use this software, please cite it as below*, 2(4):5, 2021. 1

[6] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1

[7] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023. 1

[8] Dingkang Liang, Xin Zhou, Xinyu Wang, Xingkui Zhu, Wei Xu, Zhikang Zou, Xiaoqing Ye, and Xiang Bai. Pointmamba: A simple state space model for point cloud analysis. *arXiv preprint arXiv:2402.10739*, 2024. 1

[9] Minghua Liu, Ruoxi Shi, Kaiming Kuang, Yinhao Zhu, Xuanlin Li, Shizhong Han, Hong Cai, Fatih Porikli, and Hao Su. Openshape: Scaling up 3d shape representation towards open-world understanding. *Advances in neural information processing systems*, 36, 2024. 1

[10] Tiange Luo, Chris Rockwell, Honglak Lee, and Justin Johnson. Scalable 3d captioning with pretrained models. *Advances in Neural Information Processing Systems*, 36, 2024. 1

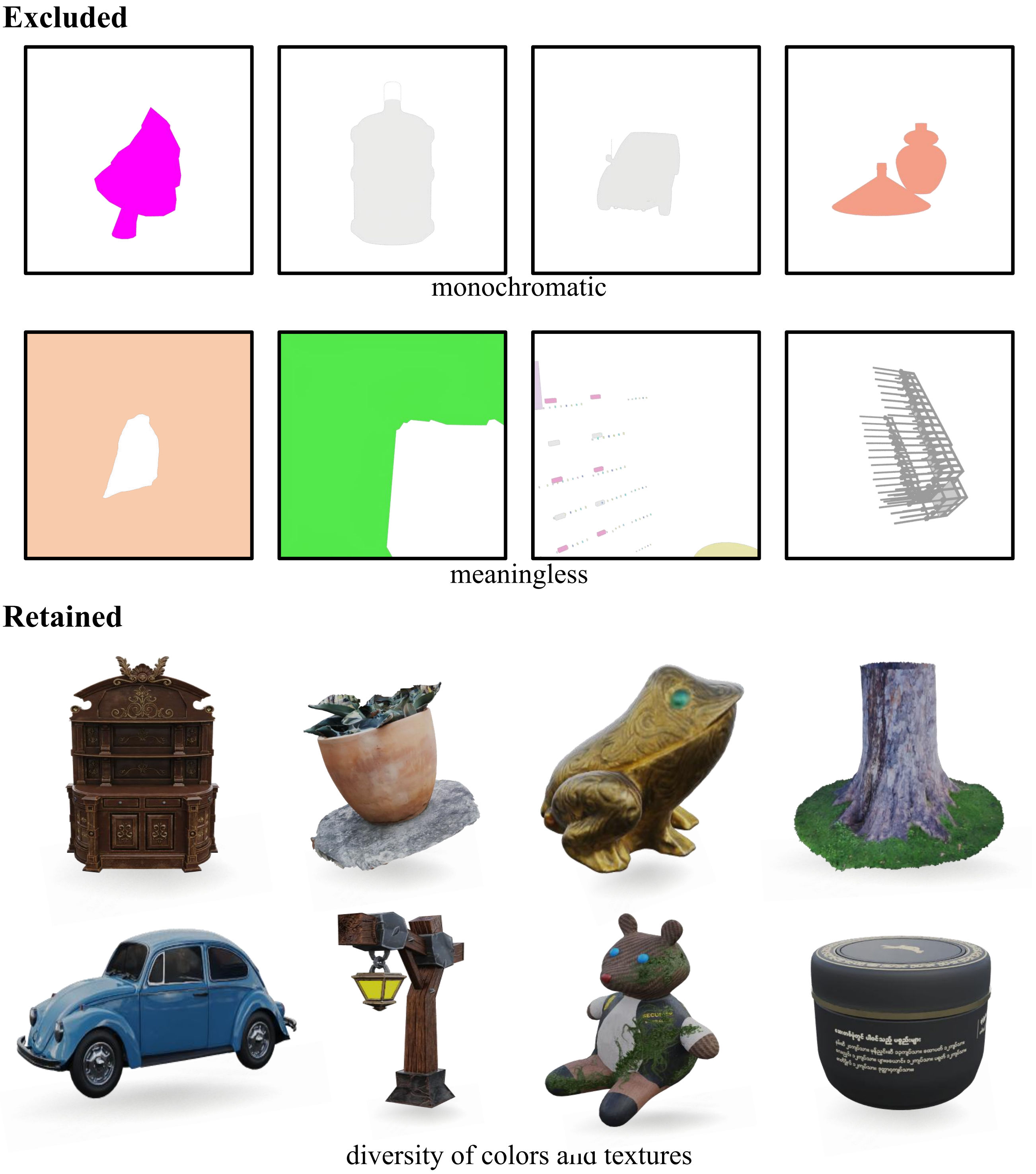

Figure 2. Visualization of the sampled 3D shapes.

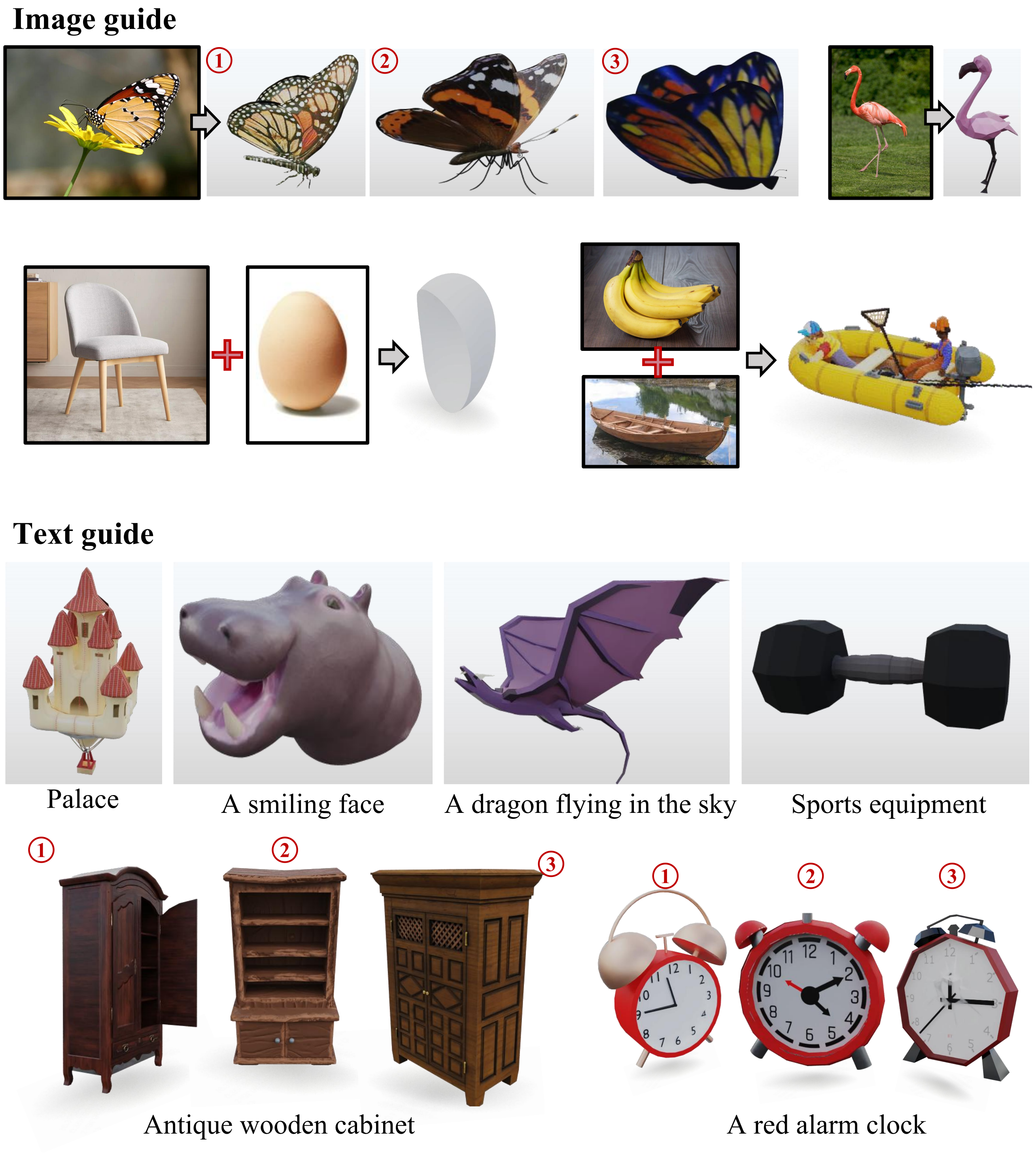

Figure 3. Image / text → 3D shape retrieval results. Top: we query the most similar or top 2 similar 3D shapes for each text. Bottom: we take one or two images as inputs and retrieve the most similar 3D shape.