Eurographics Symposium on Rendering (2025)
B. Wang and A. Wilkie (Editors)

Supplemental Material: A Controllable Appearance Representation for Flexible Transfer and Editing

Santiago Jimenez-Navarro , Julia Guerrero-Viu & Belen Masia

Universidad de Zaragoza - I3A, Spain

The supplemental material of this paper contains additional results and details not included in the main manuscript for conciseness. It is distributed as follows:

- (S1) Implementation Details of the Trained Models
- (S2) Training Dataset: Additional Details
- (S3) Additional Results of our Appearance Encoder
- (S4) Additional Ablation Studies of our Appearance Encoder
- (S5) Additional Details of our Diffusion Pipeline Evaluation
- (S6) Additional Results and Ablations of our Diffusion Pipeline

S1. Implementation Details of the Trained Models

In this section we provide further details on the configurations used to train the models discussed in the main text.

S1.1. Appearance Encoder

The network implementation builds on the work by Dubois et al. [DKLM19], which contains the definition of different VAE-based models. The system is developed using the Pytorch library [PGM*19], and trained on a NVIDIA GeForce RTX 3090. We use the Adam optimizer to optimize both the autoencoder and the discriminator in charge of computing the TC term, with learning rates of $1e^{-3}$ and $7e^{-5}$, respectively. We use a batch size of 150, and exclude the normal map information from the first two deconvolution layers. The model is trained during 2,400 epochs, with a total training time of 106 hours. The γ parameter is fixed to a constant value of 6, and we perform an annealing of the β parameter, which grows linearly from 0 to 2 during the first 1,000 epochs. For the regularization term, we use n=3.

S1.2. Diffusion-based Pipeline

We use *RealisticVisionXL4.0* (https://huggingface.co/SG161222/RealVisXL_V4.0) as the base model for our diffusion pipeline. It is a fine-tuning of Stable Diffusion XL, especially aimed at photorealism. Following community recommendations, we train the default architecture of IP-Adapter [YZL*23] for reconstruction on our training dataset. To ensure accurate appearance embeddings during training, we use the same inputs used to train our appearance encoder, at 512x512 resolution. Using object-centered images with the background masked out, we encourage the learned weights of the adapter to store information aligned with the FactorVAE's latent space. Additionally, we effectively remove the influence of the text embeddings by using uninformative prompts, such as *"image"*. Training is done in a NVIDIA A100 GPU with 10^{-4} learning rate with batch size of 10. During inference, we use the Diffusers [vPPL*22] implementation of the ControlNet inpainting pipeline for Stable Diffusion XL.

S2. Training Dataset: Additional Details

We carefully create a training dataset suitable for the disentanglement of appearance in image space. It consists of 98,550 synthetic images rendered with Mitsuba [Jak10]. These images are the result of the combination of the following factors:

- 30 geometries. We have selected geometries of different levels of complexity and realism. The 3D models come from publicly available sources: the Morgan McGuire's Computer Graphics Archive [McG17], Pixar's Renderman (https://renderman.pixar.com/community_resources), 3D Assets One (https://3dassets.one), TurboSquid (https://www.turbosquid.com/), and Polyhaven (https://polyhaven.com/models).
- 365 measured BRDFs obtained from real materials. Among these materials, 266 come from the MERL dataset [MPBM03] (173 of which are edits from the original ones [SCW*21]), 36 come from the RGL dataset [DJ18], and the remaining 63 come from the UTIA dataset [FV14]. During the selection of materials, we aimed for a dataset balanced in terms of appearance attributes.
- 9 lighting conditions. We create the different illuminations by systematically rotating the Green Point Park environment map (https://polyhaven.com/a/green_point_park) at angles -50º, -25º, 0º, 25º, and 50º in the X-axis, and 15º, 0º, -15º, -50º, and -70º in the Y-axis.

Fig. 8 shows a representative set of the aforementioned training dataset: the first three rows contain samples of the 30 geometries used, rendered with the same material and illumination. Rows four, five, and six include samples featuring the *bunny* geometry, rendered with a representative set of the materials used. Finally, the last row contains the nine illuminations.

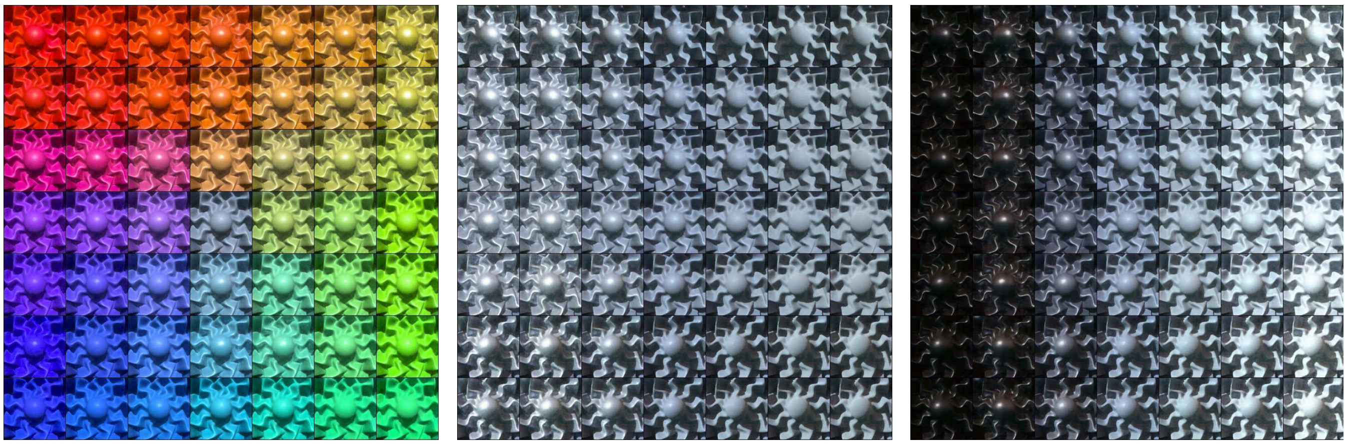

Figure 1: ***Grid plots****, showing the appearance reconstructed by the decoder when lineally combining pairs of dimensions of the latent space. We show: (left) hue #1 and hue #2, (center) light dir. #2 and gloss, and (right) light dir. #2 and lightness. Interestingly, the model has automatically learned to represent the hue in two dimensions that are perpendicular in the chromatic circle (left), without any explicit supervision.*

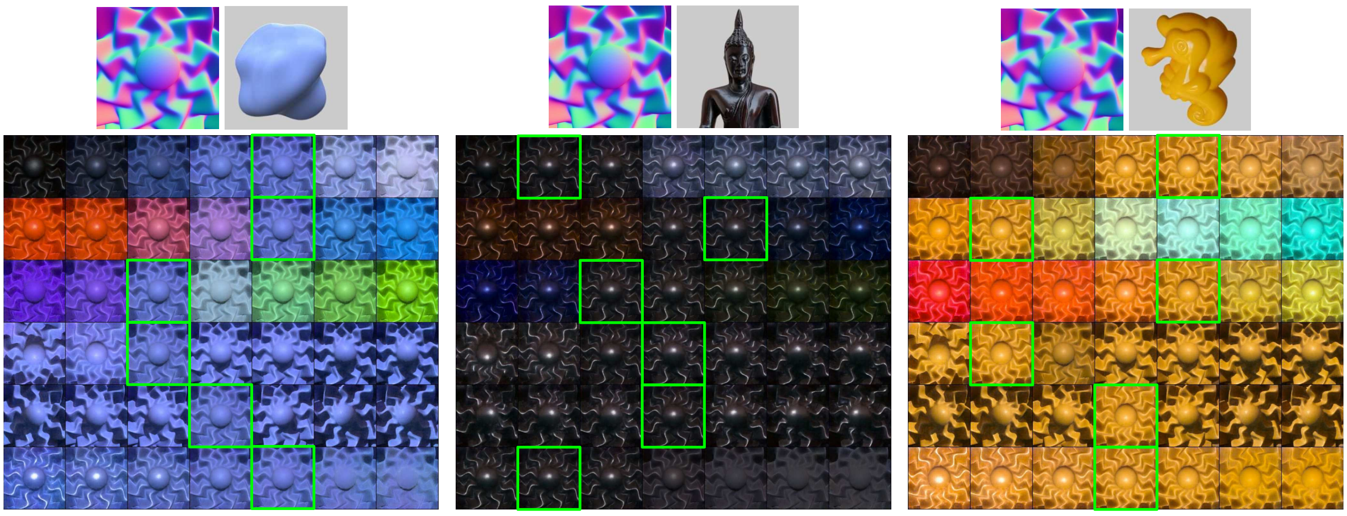

Figure 2: *Additional results of the* ***posterior traversal*** *plot. We use as reference three unseen samples, reconstructing the Havran geometry.*

S3. Additional Results of our Appearance Encoder

This section contains additional results of our model designed to encode appearance in image space. For details on how this model is built, please refer to the main text.

S3.1. Traversability and Continuity of the Latent Space

In the main text (Fig. 3), we include the *prior traversals* plot of our 6D latent space, highlighting its disentanglement and interpretability properties. Further, and thanks to these properties, we study the result of combining, two by two, these disentangled dimensions. Fig. 1 contains the result of combining three different pairs of dimensions. Thanks to the inherent continuity of VAE-based methods, we see how combining the identified dimensions results in progressive changes in the appearance of the *Havran* geometry.

S3.2. Additional Posterior Traversals

We include in Fig. 2 additional results of the *posterior traversals* plot (see main text, Fig. 4, right). Here, we compute posteriors with three additional test samples: a rendered blue rough blob, and two real photographs of a glossy black statue, and a yellow plastic seahorse, respectively. We use the *Havran* geometry as guidance in the decoder, and highlight in green the samples corresponding to reconstructions of the reference material, akin to what is done in the main text.

S3.3. Reconstruction of Unknown Geometries

As discussed in the main text (Sec. 4.2), our VAE-based appearance encoder suffers from a limited capacity to reconstruct geometries

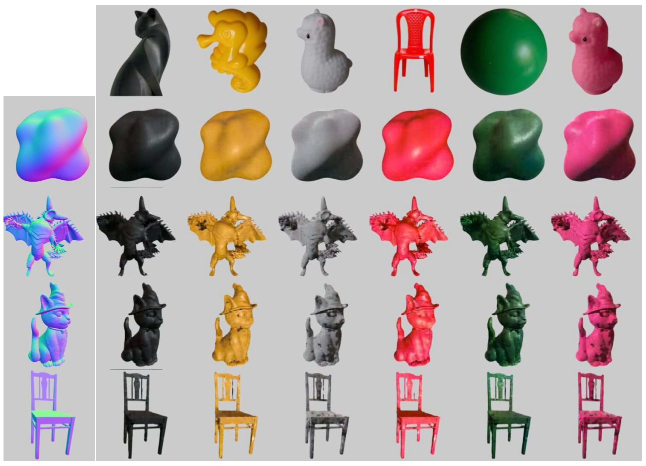

Figure 3: *Examples of reconstructions made with different goal geometries. First row depicts the six in-the-wild images used as appearance input. Second row acts as reference, performing a reconstruction using the* blob *geometry seen during training. Rows three, four, and five contain the reconstructions using the test geometries* monster, cat, *and* chair *as reference geometry.*

very different from those seen during training. This generalizability issue is not an unexpected behavior, but it hinders the applicability of such encoder *alone* to reconstruct appearance in unknown scenarios. In Fig. 3 we show the result of reconstructing the real images used in the main text (Fig. 4) with different objective geometries: as reference, we use the *blob* geometry (row two), which is in the training dataset, and then test the reconstructions for three unknown geometries, namely the *monster*, the *cat*, and the *chair* in rows three, four, and five, respectively. Analyzing the results, we see how the model is able to apply the reference appearance to some extent, but struggles in the estimation of illumination cues (i.e. shadows) and high frequency details of the geometries, while generating unexpected artifacts. This issue arises from the use of unknown geometries, which the decoder is not able to properly interpret.

S3.4. Robustness of Encoding

To evaluate the robustness of the appearance encoder's embeddings with respect to geometric variation, we conducted an experiment using two distinct test geometries: a *cylinder*, representing a smooth surface, and a *statuette*, characterized by high-frequency geometric detail. All samples were rendered under the *AutoService* environment map. We quantified embedding similarity using the mean cosine similarity (range: [-1, 1]) computed between pairs of embeddings. We observe that the mean similarity between samples of the same material is 0.879, whereas it decreases to 0.321 for samples with different materials. These findings suggest that the encoder consistently maps samples with identical materials to similar regions of the latent space, and vice versa.

Table 1: *Ablation study on alternative reconstruction loss functions. We include results of interpretability (MIR metric) for four different reconstruction losses.*

	BCE	Bernoulli	Huber	L1-smooth
MIR↑	0.1770	0.2440	0.2622	**0.2940**

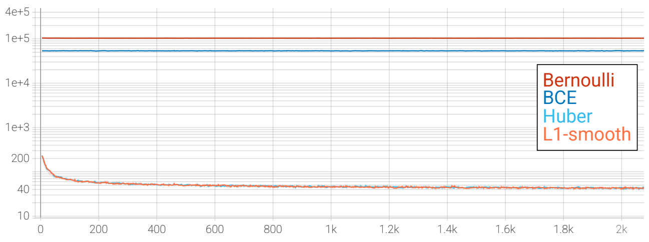

Figure 4: *Ablations. Evolution of different alternative reconstruction losses during training. Note that y-axis is log-scale.*

S4. Additional Ablation Studies of our Appearance Encoder

In this section we include additional ablation studies of our appearance encoder, highlighting the thoughtful design of our final model (see main text, Sec. 3.4 for the main ablations).

S4.1. Reconstruction Loss

During the design of the final model, we tested four commonly used reconstruction losses in autoencoder architectures, namely BCE, Bernoulli, Huber, and L1-smooth [PGM*19]. In the final implementation, we chose to use the L1-smooth function, due to its superior performance in our experiments, as shown in its improved interpretability, measured by MIR, in Table 1). Additionally, the first two reconstruction functions operate at a considerably higher magnitude than the latter two alternatives (Fig. 4), which caused training instabilities.

S4.2. Normal Map Downsampling Method

In the Sec. 3.1 of the main text, we explain that the geometry guidance in the decoder of the FactorVAE-based architecture is implemented by adequately concatenating the channels to layers of the decoder pipeline. To adequately match the shape of the maps to that of the feature maps of the layers, we need to resize the normal map image. Here we ablate the algorithm used to perform such resizing of the normal map, comparing the nearest neighbors (NN) and bilinear methods. Fig. 5 shows the performance of two models, each one trained with one of the alternatives, when reconstructing a test geometry. We can clearly see how the model that uses the bilinear interpolation achieves a better understanding of the unseen geometry, which is reflected in a reduced (but not inexistent) number of artifacts. This is probably due to a *smoother* reduction of the resolution by the bilinear algorithm, in contrast to the nearest neighbors one.

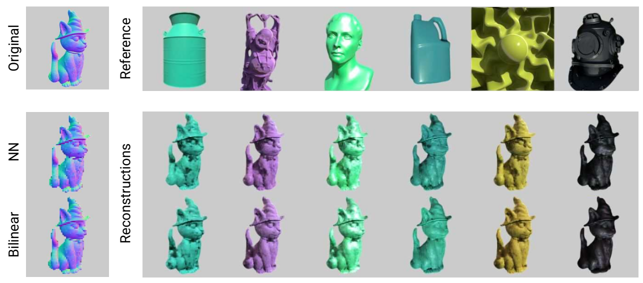

Figure 5: *Ablation study on the two alternatives for downsampling the normal map information, namely nearest neighbors (NN), and bilinear. Left: the reference geometry (*cat*), with its respective resulting appearance after applying each of the two algorithms. Right: samples used as appearance reference (first row), with their respective reconstructions using each algorithm (second and third row).*

S5. Additional Details of our Diffusion-based Pipeline Evaluation

Here we provide details regarding the evaluations performed in the Sec. 5.2 of the main paper. The different real images we used as input of the pipeline come from public datasets [SCW*21; DLC*22], have been AI-generated [Lab23] or come from the following sources:

- Copper teapot (Fig. 1, main, Fig. 6, supp): rawpixel.com.
- Napoleon (Figs. 1, 8, 11, main): Daniel Robert, unsplash.com.
- Green chair 1 (Fig. 1, main): alicdn.com.
- Red chair (Fig. 4, main): imimg.com.
- David (Fig. 7, main): Roberto Serra, news.artnet.com.
- Green chair 2 (Fig. 8, main, Fig. 6, supp): grangecoop.com.
- Luffy statue (Fig. 10, main): kumamoto.guru.
- Statue of Liberty (Fig. 10, main): Riis2602 Wikimedia.

In the Fig. 8 of the main text, we compare our results with two existing baselines: InstructPix2Pix[BHE23], and Subias and Lagunas [SL23]. For emulating the traversal of our latent space in the InstructPix2Pix pipeline, we used the following configurations:

- For increasing the gloss of the statue, we used the prompt *make the statue shiny*, with a text CFG of 5.0, 6.0, and 7.5.
- For decreasing the hue #2 of the resulting image of the previous stage, we used the prompt *make the statue orange*, with a text CFG of 2.0, 3.0, and 4.0.
- For increasing the hue #2 of the chess piece, we used the prompt *make the chess piece green*, with a text CFG of 0.5, 2.0, and 5.0.
- For increasing the lightness of the resulting image, we used the prompt *make the chess piece light green*, with a text CFG of 1.0, 5.0, and 8.0.

In the case of the Subias and Lagunas' pipeline, we directly used it for the attributes of the comparative handled by their model.

S6. Additional Results and Ablations of our Diffusion Pipeline

In this section we include additional results obtained with our diffusion-based pipeline, discussed in the Sec. 5 of the main manuscript, and run ablations on some of the design decisions taken in the development of such pipeline.

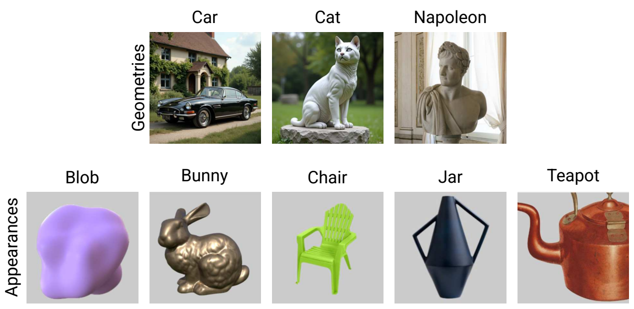

Figure 6: *Inputs used for the diffusion-based posterior traversals presented in Figs. 9- 23.*

S6.1. Posterior Traversals with our Diffusion-based Pipeline

Once trained, our diffusion pipeline is expected to resemble the good properties of our appearance encoder (e.g., disentanglement, interpretability, continuity). To evaluate this, we create a version of the *posterior traversals* plot (main text, Fig. 4, right) with the designed diffusion-based pipeline, using the four dimensions that contain appearance-related features. We use the remaining two dimensions encoding illumination features exclusively to guide the pipeline on the desired lighting conditions, thus we do not perform a traversal on such dimensions. For it, we start from the setup displayed in the teaser (main text, Fig. 1, left), where we apply a rough purple material to the painting of a car. Starting from this appearance transfer (figures marked in green), we systematically vary each of the four dimensions, obtaining as a result the visualization of Fig. 9. Rows follow the same order as the traversals in Fig. 2, namely lightness, hue #1, hue #2, and gloss. Note the overall smoothness of transitions, achieved thanks to the guidance of our custom appearance encoder. When traversing the same dimension as the appearance transfer (e.g. trying to make the paint *even* more purple), the representation may overflow the ranges seen during training, creating saturated results. To further evaluate the robustness of our method, we generate diffusion-based posterior traversals using 15 combinations of the three geometries and five appearances illustrated in Fig. 6. The resulting traversals are presented in Figs. 9- 23. These results encompass a diverse range of appearances and reference geometries, showcasing our method's applicability to real-world scenarios, as well as its limitations in some challenging examples.

S6.2. Appearance Transfer Evaluation

To further assess the improved disentanglement between geometry and appearance achieved by our custom appearance encoder, we conducted a user study involving 20 participants. Each participant was presented with results from an appearance transfer task and asked to select the output that best fulfilled the objective: "Generate an image with the given geometry and the given material appearance". As a baseline, we used the outputs of ZeST [CSM*24],

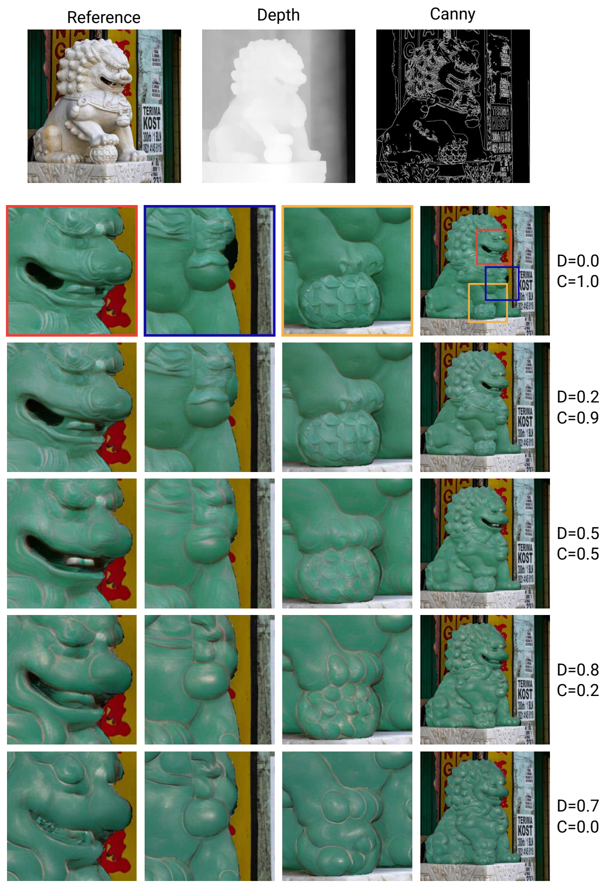

Figure 7: *Ablation on the influence of the ControlNet weights over the final result. On top are represented the geometry used as reference, with its respective estimated depth and Canny maps. Each row contains the result of running the inference pipeline with the same custom reference appearance and different weights. We include three close-up zooms for visualization purposes, and the respective weights for the Depth (D) and Canny (C) ControlNets.*

following the configurations shown in Figs. 9–23. Results showed that our proposed method was preferred in 61.5% of the selections, while ZeST was chosen in 38.5% of the cases.

S6.3. Ablation Study on the Influence of ControlNets

As described in the main paper, our inference pipeline leverages a combination of two ControlNets [ZRA23] to provide geometry guidance. Each ControlNet must be appropriately weighted, as these weights directly influence the final image. In our work, we use fixed weights of 0.2 for the Depth ControlNet and 0.9 for the Canny ControlNet. To analyze the impact of these weights, we run an ablation study, presented in Fig. 7. As observed, the Canny map may contain some ambiguous edges that require complex interpretation during inference. When relying solely on this information (D=0.0, C=1.0), the pipeline may misinterpret certain edges as being outside of the main geometry, leading to artifacts such as the black blob (first row, second column). Conversely, the depth map helps to better identify the geometry but lacks some details of the scene. If only conditioning the diffusion pipeline with the estimated depth map (Fig. 7, last row), high-frequency details are significantly diminished, which is especially noticeable in the sphere (last row, third column). Among the tested configurations, our chosen weights offer the best balance between preserving geometric structure and maintaining high-frequency fine details.

References

[BHE23] Brooks, Tim, Holynski, Aleksander, and Efros, Alexei A. "Instructpix2pix: Learning to follow image editing instructions". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, 18392–18402 4.

[CSM*24] Cheng, Ta-Ying, Sharma, Prafull, Markham, Andrew, et al. "Zest: Zero-shot material transfer from a single image". *Proceedings of the European Conference on Computer Vision*. Springer. 2024, 370–386 4.

[DJ18] Dupuy, Jonathan and Jakob, Wenzel. "An Adaptive Parameterization for Efficient Material Acquisition and Rendering". *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)* 37.6 (Nov. 2018), 274:1–274:18 1.

[DKLM19] Dubois, Yann, Kastanos, Alexandros, Lines, Dave, and Melman, Bart. *Disentangling VAE*. `http://github.com/YannDubs/disentangling-vae/`. Mar. 2019 1.

[DLC*22] Delanoy, Johanna, Lagunas, Manuel, Condor, Jorge, et al. "A Generative Framework for Image-based Editing of Material Appearance using Perceptual Attributes". *Computer Graphics Forum*. Vol. 41. 1. Wiley Online Library. 2022, 453–464 4.

[FV14] Filip, Jirí and Vávra, Radomír. "Template-based sampling of anisotropic BRDFs". *Computer Graphics Forum*. Vol. 33. 7. Wiley Online Library. 2014, 91–99 1.

[Jak10] Jakob, Wenzel. *Mitsuba renderer*. http://www.mitsuba-renderer.org. 2010 1.

[Lab23] Labs, Black Forest. *FLUX*. `https://github.com/black-forest-labs/flux`. 2023 4.

[McG17] McGuire, Morgan. *Computer Graphics Archive*. July 2017. URL: `https://casual-effects.com/data` 1.

[MPBM03] Matusik, Wojciech, Pfister, Hanspeter, Brand, Matt, and McMillan, Leonard. "A Data-Driven Reflectance Model". *ACM Transactions on Graphics (TOG)* 22.3 (July 2003), 759–769 1.

[PGM*19] Paszke, Adam, Gross, Sam, Massa, Francisco, et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, 8024–8035 1, 3.

[SCW*21] Serrano, Ana, Chen, Bin, Wang, Chao, et al. "The effect of shape and illumination on material perception: model and applications". *ACM Transactions on Graphics (TOG)* 40.4 (2021) 1, 4.

[SL23] Subias, J Daniel and Lagunas, Manuel. "In-the-wild Material Appearance Editing using Perceptual Attributes". *Computer Graphics Forum*. Vol. 42. 2. Wiley Online Library. 2023, 333–345 4.

[vPPL*22] Von Platen, Patrick, Patil, Suraj, Lozhkov, Anton, et al. *Diffusers: State-of-the-art diffusion models*. `https://github.com/huggingface/diffusers`. 2022 1.

[YZL*23] Ye, Hu, Zhang, Jun, Liu, Sibo, et al. "Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models". *arXiv preprint arXiv:2308.06721* (2023) 1.

[ZRA23] Zhang, Lvmin, Rao, Anyi, and Agrawala, Maneesh. "Adding conditional control to text-to-image diffusion models". *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, 3836–3847 5.

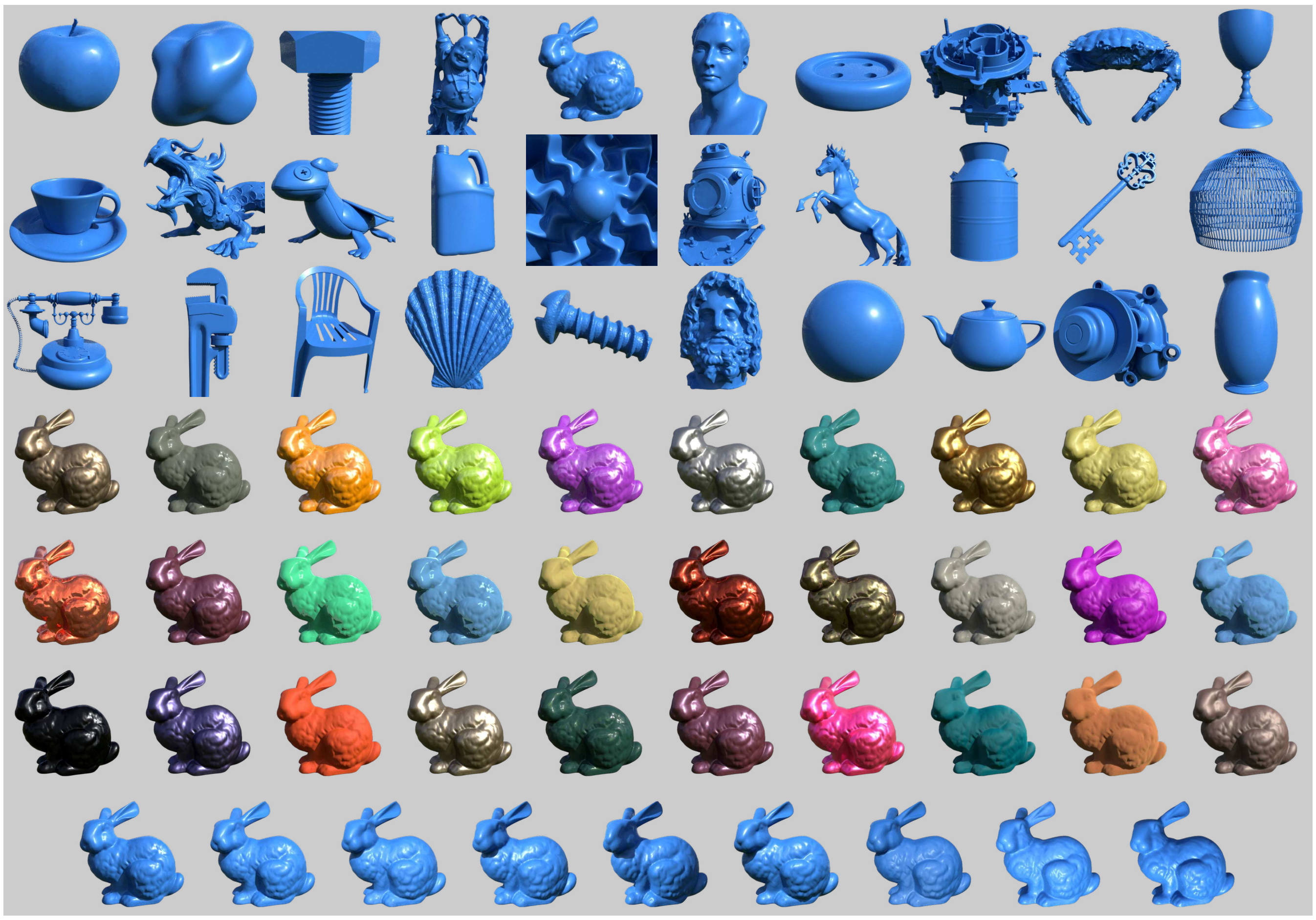

Figure 8: *Representative samples of the custom training dataset. Rows one, two, and three contain samples of the 30 geometries used. Rows four, five, and six contain a representative set of the 365 measured materials. Row seven shows the nine illuminations used in the dataset.*

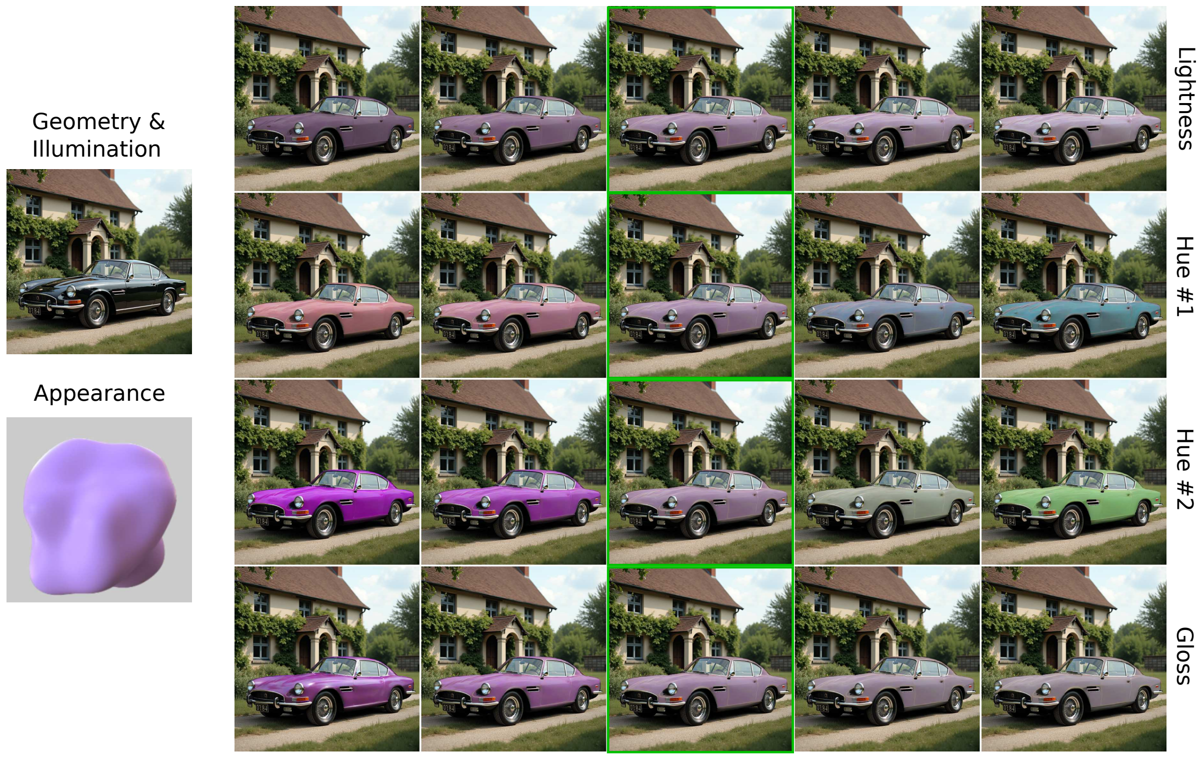

Figure 9: Posterior traversals *plot generated using **car** and **blob** as geometry and appearance references. Transfer marked in green.*

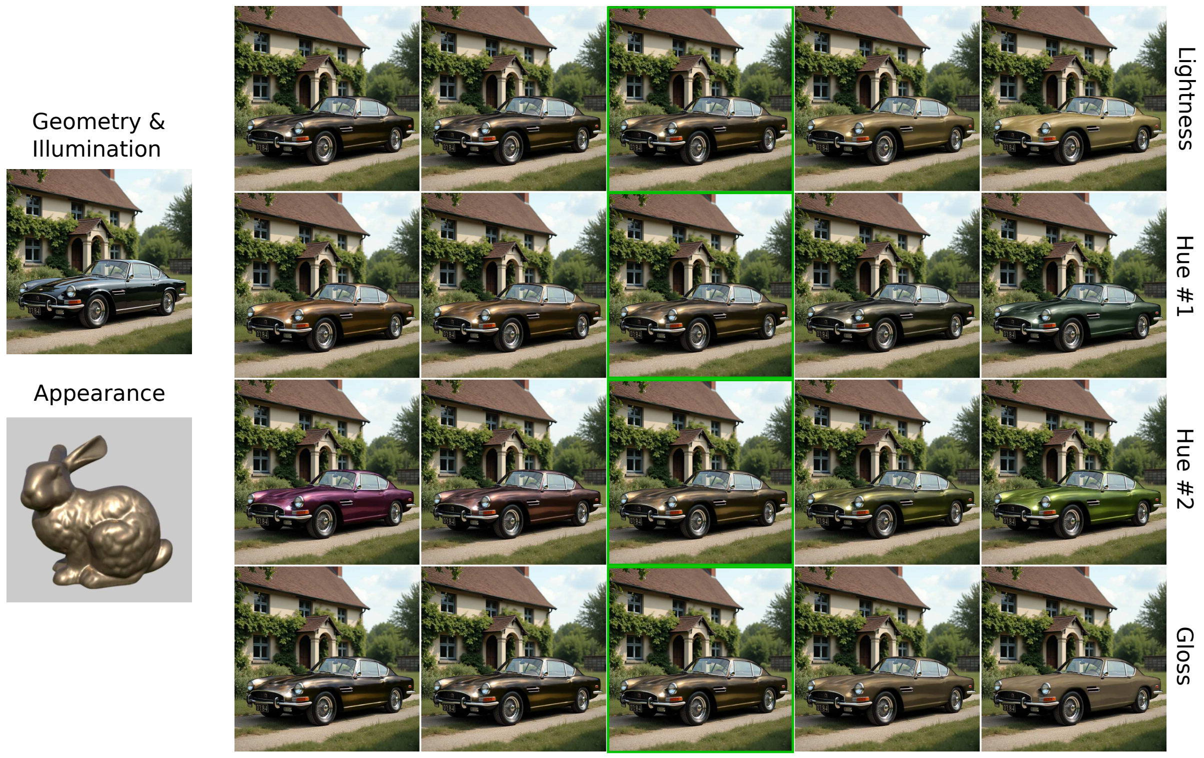

Figure 10: Posterior traversals *plot generated using **car** and **bunny** as geometry and appearance references. Transfer marked in green.*

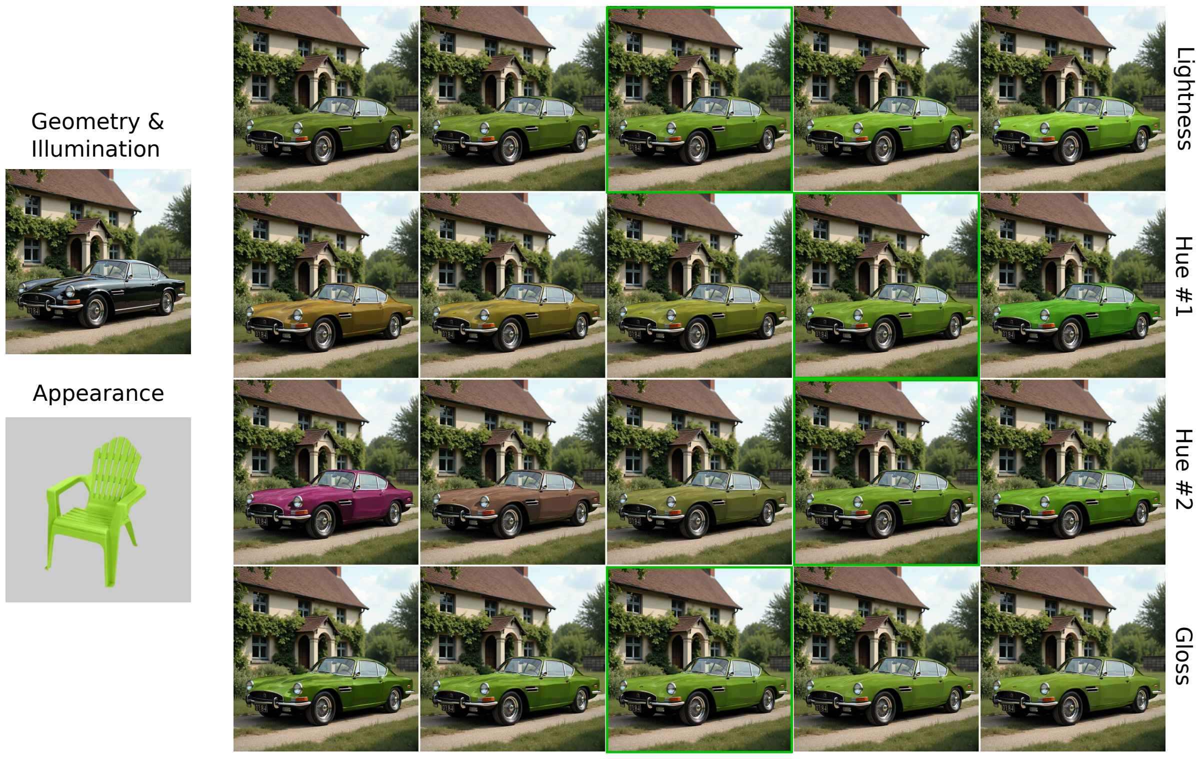

Figure 11: Posterior traversals *plot generated using **car** and **chair** as geometry and appearance references. Transfer marked in green.*

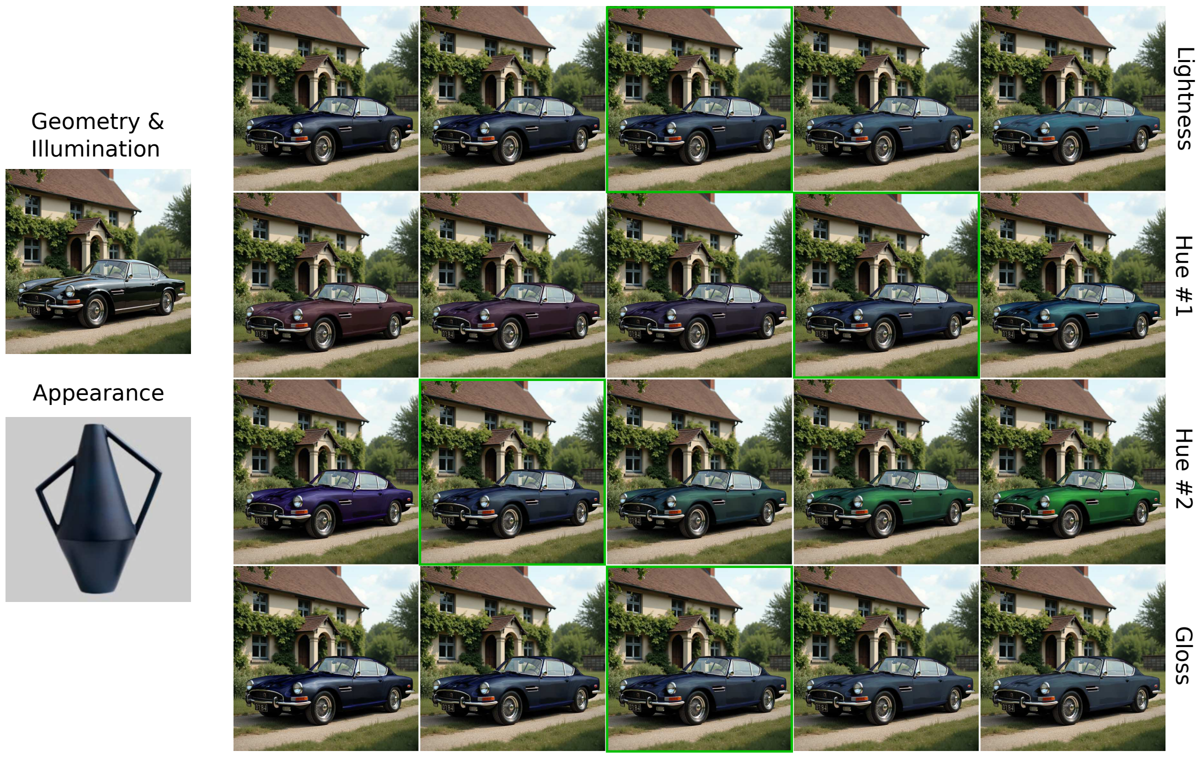

Figure 12: Posterior traversals *plot generated using **car** and **jar** as geometry and appearance references. Transfer marked in green.*

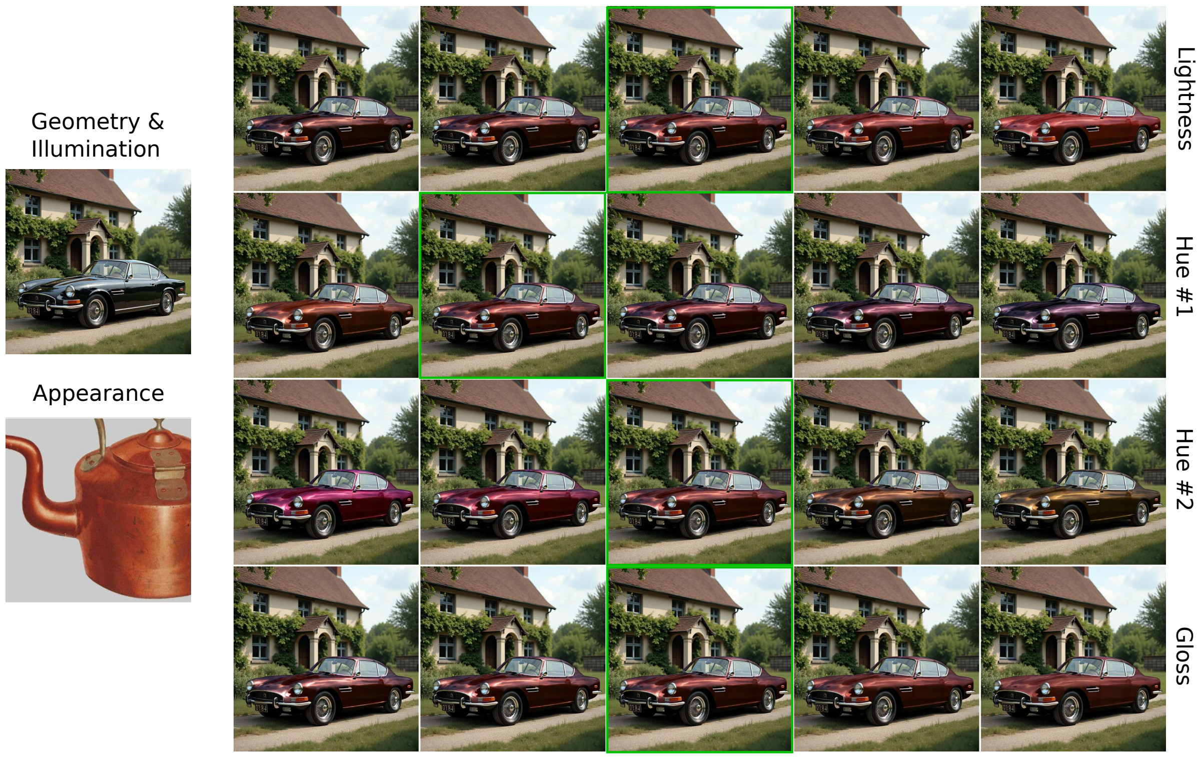

Figure 13: Posterior traversals *plot generated using **car** and **teapot** as geometry and appearance references. Transfer marked in green.*

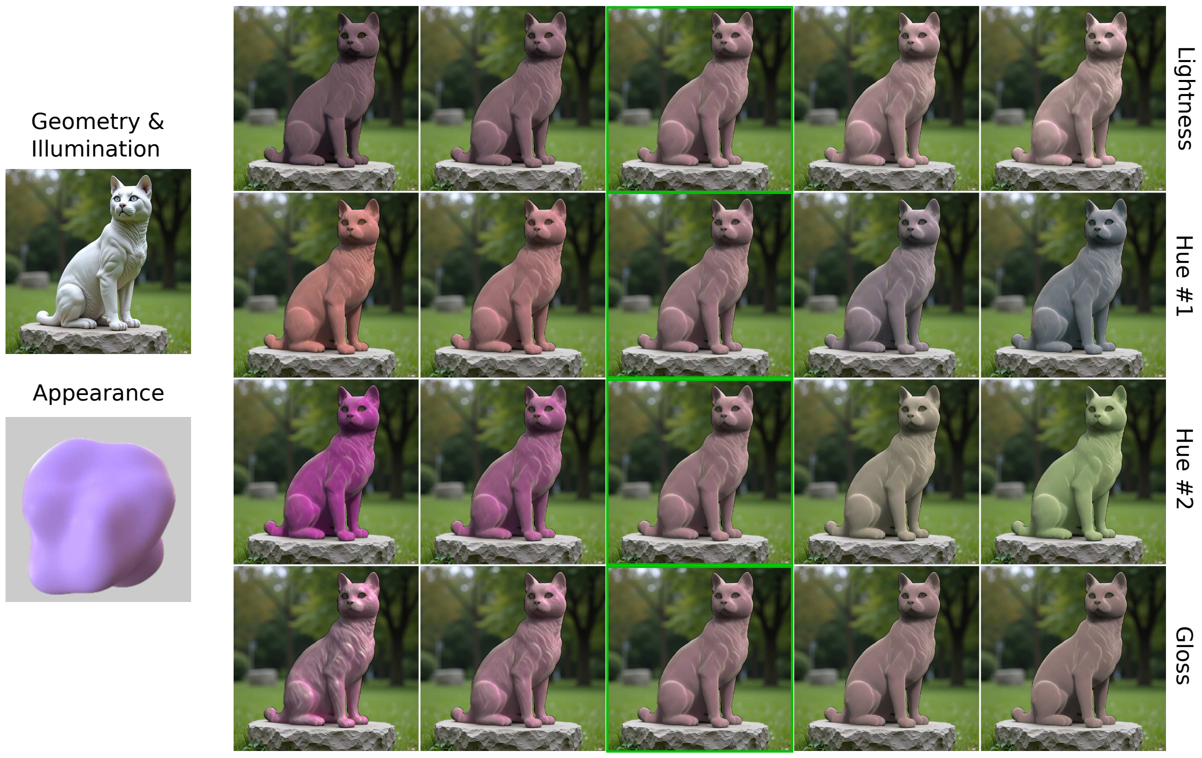

Figure 14: Posterior traversals *plot generated using **cat** and **blob** as geometry and appearance references. Transfer marked in green.*

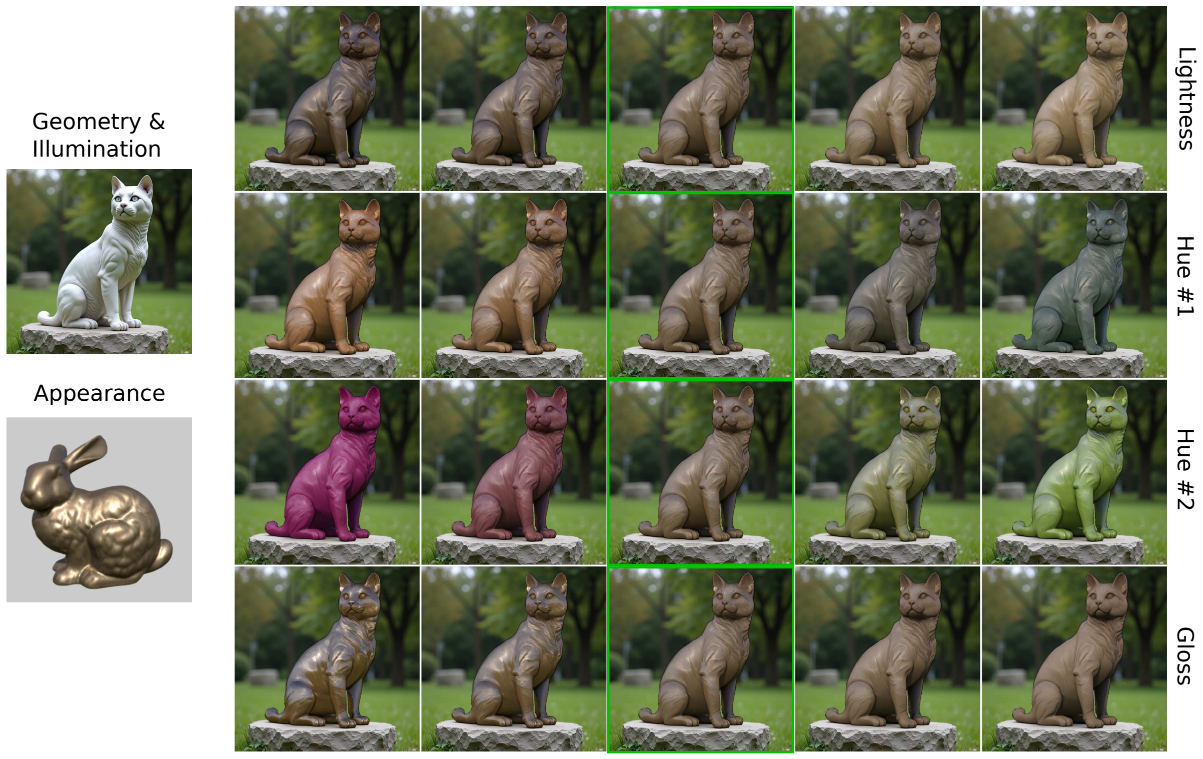

Figure 15: Posterior traversals *plot generated using **cat** and **bunny** as geometry and appearance references. Transfer marked in green.*

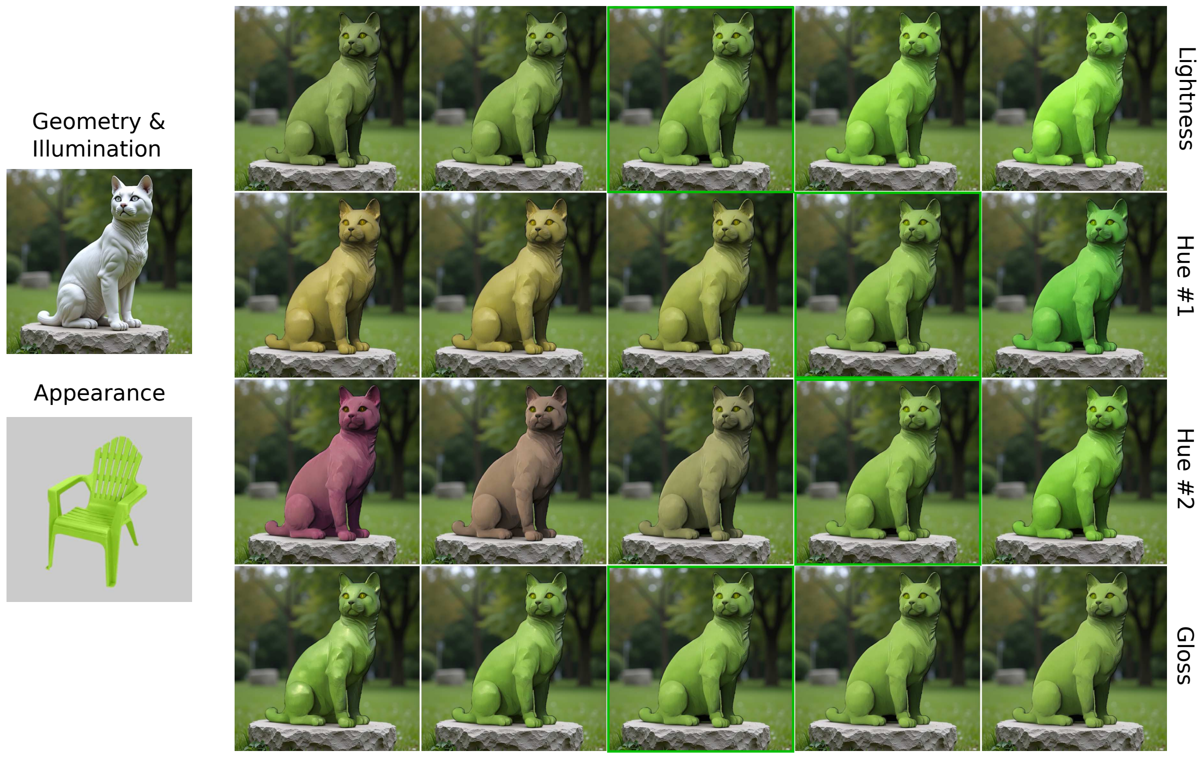

Figure 16: Posterior traversals *plot generated using **cat** and **chair** as geometry and appearance references. Transfer marked in green.*

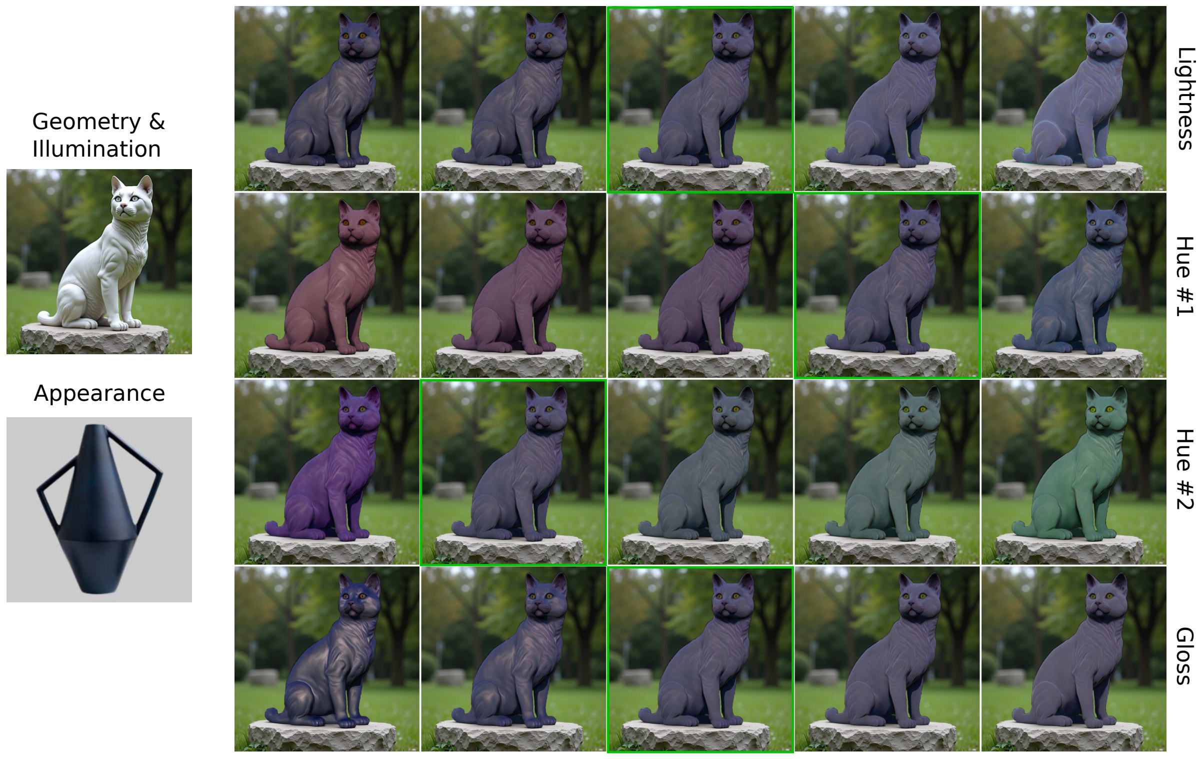

Figure 17: Posterior traversals *plot generated using **cat** and **jar** as geometry and appearance references. Transfer marked in green.*

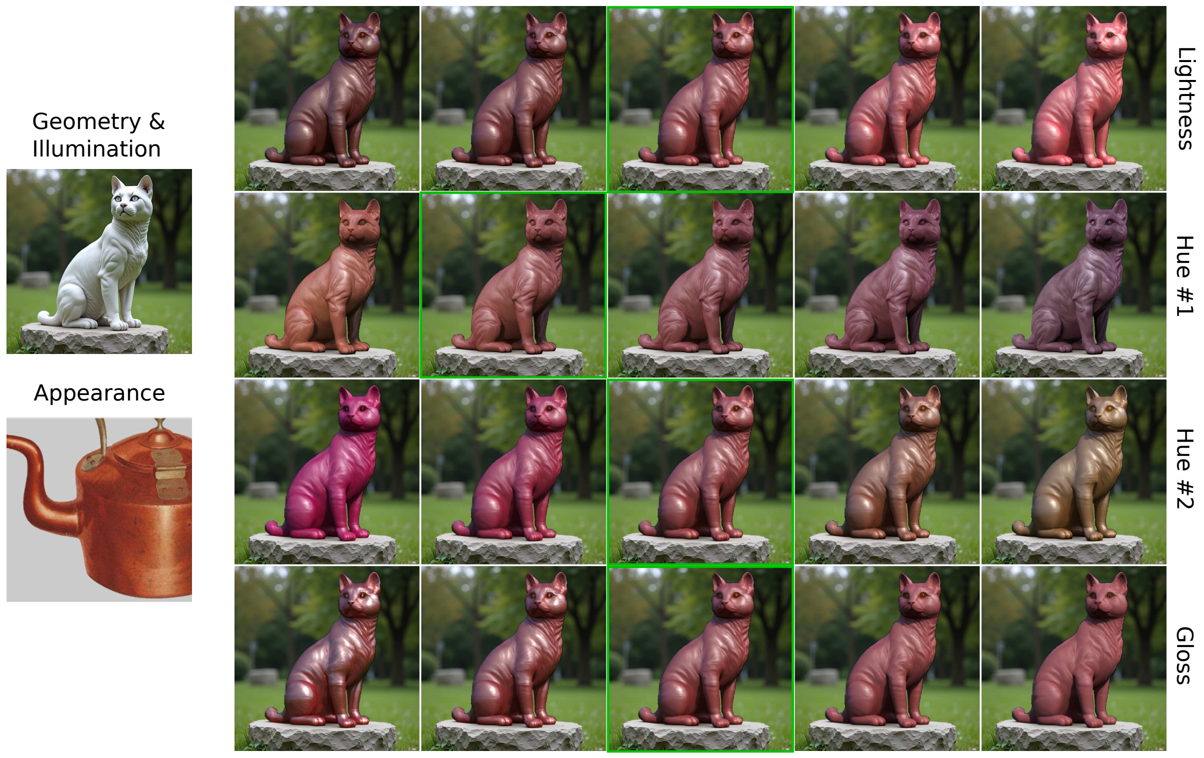

Figure 18: Posterior traversals *plot generated using **cat** and **teapot** as geometry and appearance references. Transfer marked in green.*

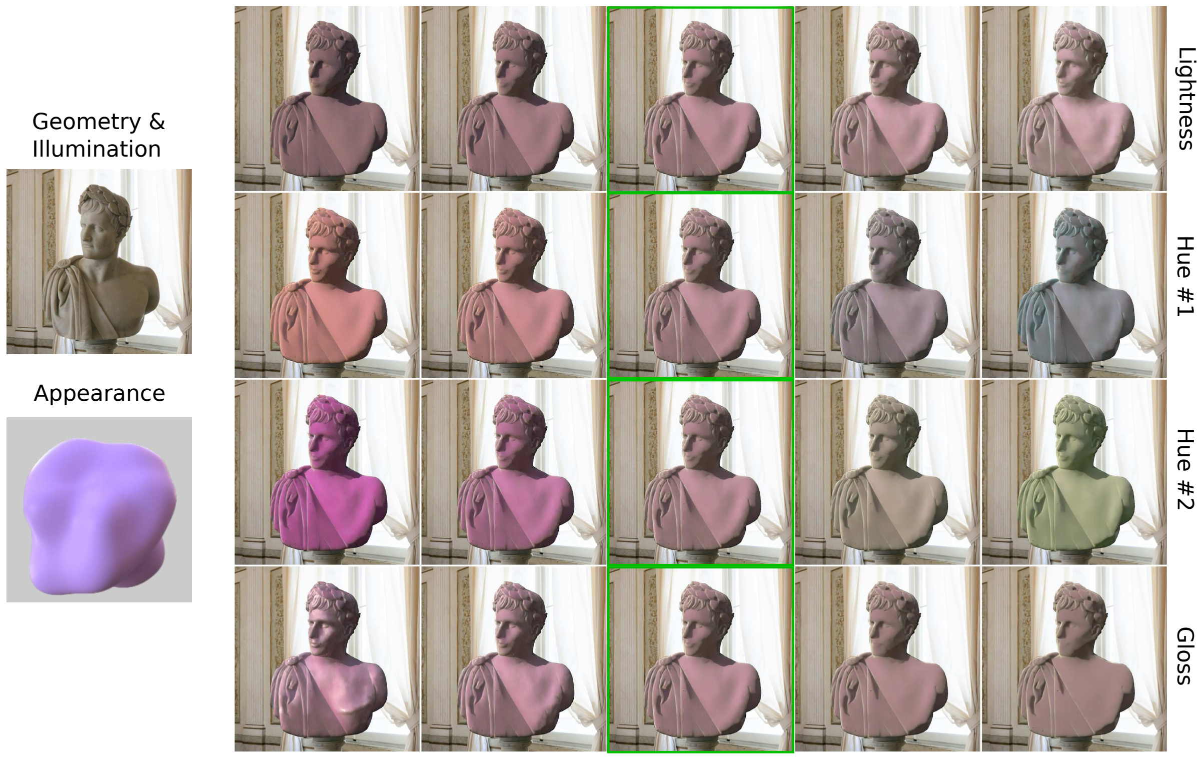

Figure 19: Posterior traversals *plot generated using **napoleon** and **blob** as geometry and appearance references. Transfer marked in green.*

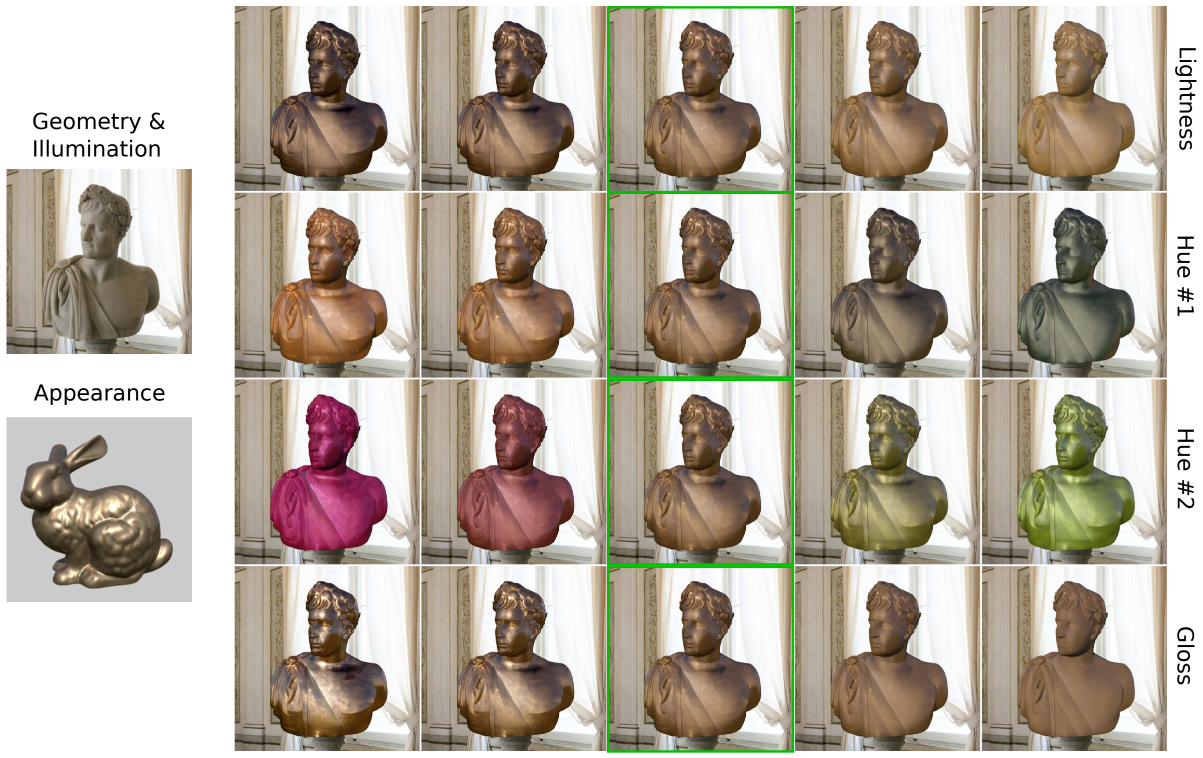

Figure 20: Posterior traversals *plot generated using **napoleon** and **bunny** as geometry and appearance references. Transfer marked in green.*

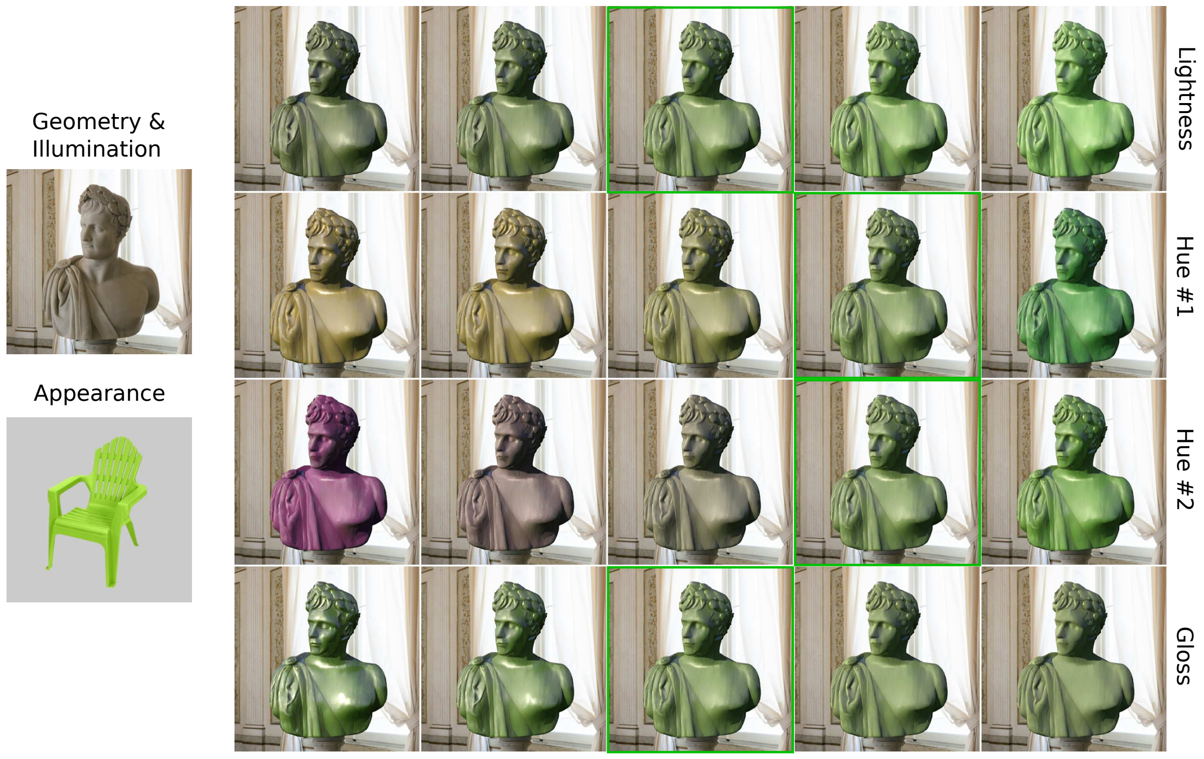

Figure 21: Posterior traversals *plot generated using* ***napoleon*** *and* ***chair*** *as geometry and appearance references. Transfer marked in green.*

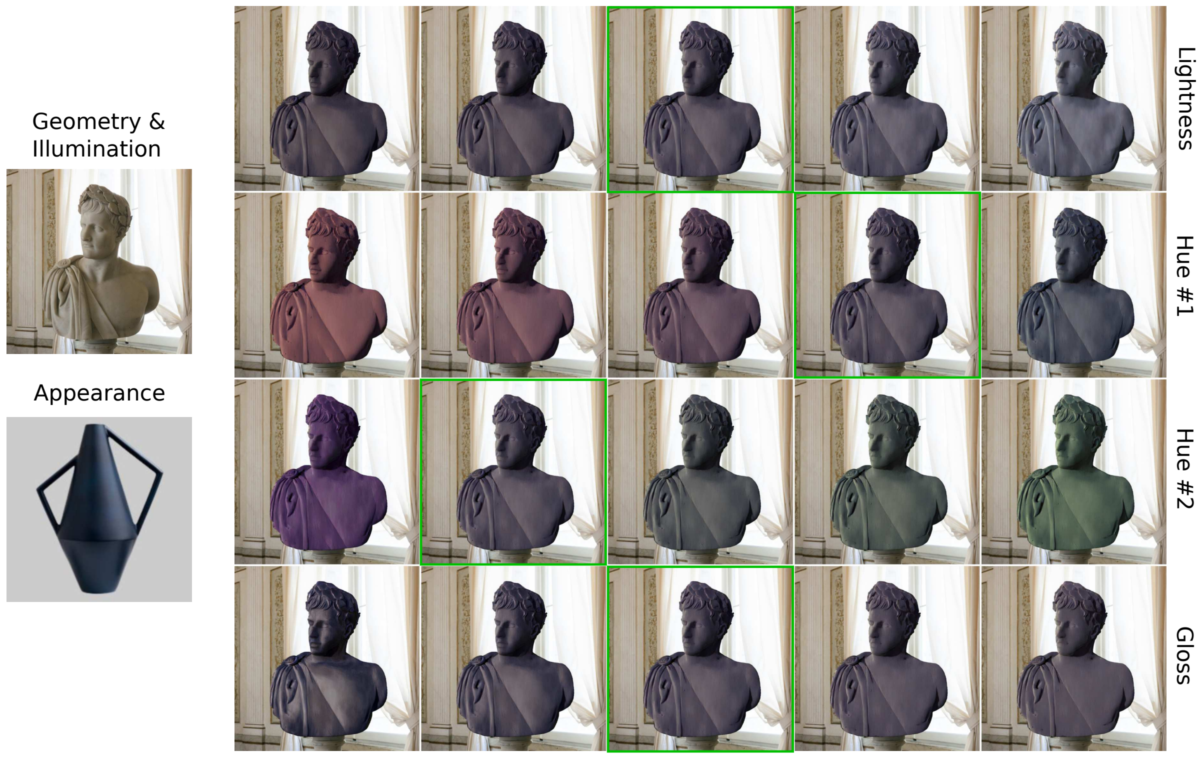

Figure 22: Posterior traversals *plot generated using* ***napoleon*** *and* ***jar*** *as geometry and appearance references. Transfer marked in green.*

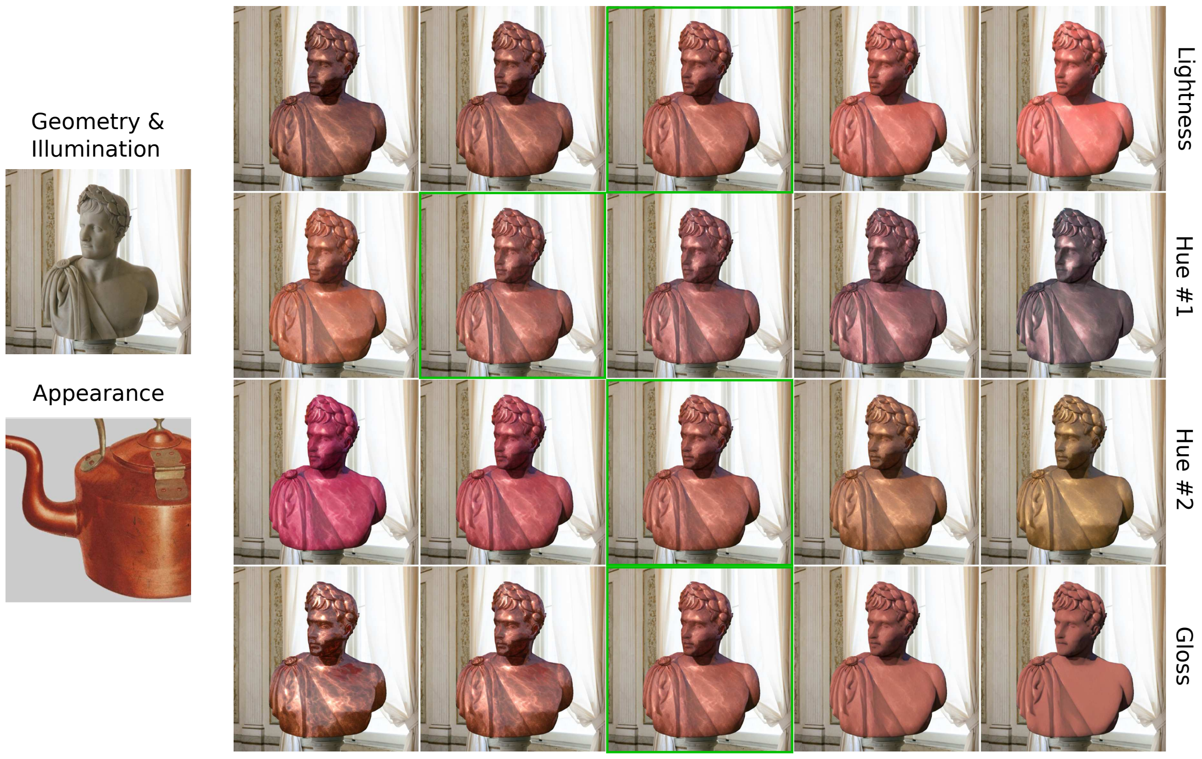

Figure 23: Posterior traversals *plot generated using* ***napoleon*** *and* ***teapot*** *as geometry and appearance references. Transfer marked in green.*