

Learning to Transfer Heterogeneous Translucent Materials from a 2D Image to 3D Models

Supplementary Material

Anonymous Authors

1 INTRODUCTION

This supplemental material contains five parts:

- Sec. 2 gives the details of the Efficient Viewpoints Selection algorithm.
- Sec. 3 gives more supplementary ablation experiments
- Sec. 4 provides additional results of our method.

2 EFFICIENT VIEWPOINTS SELECTION

In existing methods for generating geometric and material properties based on diffusion models, viewpoints selection typically falls into two categories: a large range of random viewpoints (e.g., DreamFusion [3], Fantasia3D [1]) and fixed viewpoints (e.g., Texture [4], Instruct-Nerf2Nerf [2]). Random viewpoints often demand a substantial number of samples to ensure thorough coverage, proving effective for geometry generation and optimization. However, for our translucent material generation, random viewpoints introduce excessive redundancy, significantly diminishing efficiency. On the other hand, fixed perspectives, though manually set for distribution control, lack adaptability for batch 3D model material generation and may lack reliability.

To effectively minimize redundant viewpoints and achieve maximum coverage of the model during the editing of translucent material on 3D models, we proposed an efficient viewpoint selection method termed Efficient Viewpoints Selection (EVS). The primary goal of EVS is to select optimal viewpoints, minimizing the required number of viewpoints while achieving comprehensive model coverage.

Viewpoints sampling. Given a sampled point p_s on the 3D model's surface, we first obtain the corresponding surface normal n . Using this sampled surface point as the origin and its normal as the ray direction, we calculate the intersection point p_i of this ray with a sphere of radius r . The camera position is located at the intersection p_i , and the camera orientation is aligned with the direction from the intersection p_i to the sampled surface point p_s . This process generates multiple viewpoints $v_i \in \{v_0, \dots, v_n\}$ for subsequent use. The approach of sampling using the normals of sampled surface points demonstrates enhanced model adaptability compared to random viewpoint sampling within the boundary sphere. This technique is particularly effective in addressing occlusion issues in various complex models.

Efficient viewpoints selection. We adopt an "efficient viewpoints selection" strategy to select the most suitable sampled viewpoints v_i . Our algorithm is detailed in Algorithm 1. The algorithm iteratively selects the best viewpoint v_i from the sampled viewpoints $V_{sampled}$, rendering the scene from each viewpoint. The algorithm

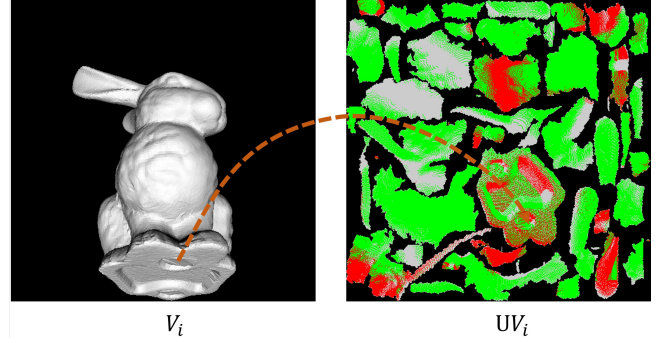


Figure 1: UV Mapping. Transforming the UV feature map of each viewpoint from UV coordinates to pixel coordinates on the texture map, we generate a UV color map corresponding to each viewpoint V_i . For a UV color map UV_i , distinct colors denote different indications.

then divides the surface into two parts: S_{new} , the surface area visible from the current viewpoint, and S_{others} , the remaining surface area. The algorithm calculates the pixel values of the two parts and evaluates the viewpoint score based on the ratio of the sum of pixel values to the total surface area of S_{new} . The algorithm selects the viewpoint with the highest score and adds it to the selected viewpoints $V_{selected}$. The algorithm repeats this process until convergence, ensuring efficient viewpoint selection for comprehensive model coverage.

To validate the effectiveness of our algorithm, we designed a UV color map through UV mapping to verify the effectiveness of the viewpoint coverage. Initially, we obtain the sampled viewpoints and render the scene for each viewpoint. In each viewpoint, we obtain rasterized UV features ranging from 0 to 1, forming a UV feature map with two channels. Subsequently, we convert the UV feature maps of each viewpoint from UV coordinates to pixel coordinates of the texture map. This transformation ensures that each pixel in the UV feature map has a corresponding mapping to the texture map's pixels. This conversion yields a UV color map UV_i of each viewpoint. For a UV color map UV_i , distinct colors denote different indications: The "black" area indicates regions without UV mapping. The "gray" area represents regions with UV mapping but is not projected by the current viewpoint. The "green" area indicates regions covered by projections from previously selected viewpoints. The "red" area shows regions not previously covered but projected by the current viewpoint.

$$color(UV_i) = \begin{cases} \text{"black"}, & \text{Non-UV mapping area} \\ \text{"gray"}, & \text{Uncovered UV map area} \\ \text{"green"}, & \text{Covered UV map area} \\ \text{"red"}, & \text{Covered area by current view} \end{cases}$$

Algorithm 1: Efficient Viewpoints Selection

```

input : 3D mesh
output:  $V_{selected}$ 

1  $mesh \leftarrow \text{Normalize}(mesh)$ ;
2  $samplePoints, normals \leftarrow \text{SampleSurfacePoints}(mesh)$ ;
3  $V_{sampled} \leftarrow \emptyset$ ;
4 foreach  $p_s, n \in samplePoints$  do
5    $p_i \leftarrow \text{BoundingSphereIntersection}(p_s, n, r = 2)$ ;
6    $v_i \leftarrow \text{SetViewOrientation}(p_i \rightarrow p_s)$ ;
7    $V_{sampled}.append(v_i)$ ;
8 end
9  $V_{selected} \leftarrow \emptyset$ ;
10 while not converged do
11   foreach  $v_i \in V_{sampled}$  do
12      $S_{new}, S_{others} \leftarrow$ 
13        $\text{DivideSurface}(mesh, v_i, V_{selected})$ ;
14      $pixel(S_{new} = 1, S_{others} = 0) \leftarrow \text{RenderSetting}$ ;
15      $image \leftarrow \text{Render}(mesh, v_i)$ ;
16      $\sum P_{value=1} \leftarrow \text{Sum}(image[value = 1])$ ;
17      $A(S_{new}) \leftarrow \text{TotalSurfaceArea}(S_{new})$ ;
18      $Score(v_i) \leftarrow \frac{\sum P_{value=1}}{A(S_{new})}$ ;
19   end
20    $v_{best} \leftarrow \text{SelectBestScore}(V_{sampled}, Score(v_i))$ ;
21    $V_{selected}.append(v_{best})$ ;
22    $V_{sampled}.remove(v_{best})$ ;
23 end
24 return  $V_{selected}$ ;

```

3 ABLATION EXPERIMENTS

Ablation of viewpoints selection. Figure 2 shows the results of transferring different materials to the 3D chair model using two different viewpoint selection methods. The results show that the selection of viewpoints greatly influences experiment outcomes. As shown in 2, a Dense viewpoint results in excessive overlap, leading to over-averaged results. On the other hand, dynamic viewpoint selection ensures efficient surface coverage, minimizing the required number of viewpoints and yielding more detailed results.

Consistency Constraint on Editing Results. We validated the effectiveness of our iterative editing-optimization strategy in ensuring consistency in the editing results. As shown in Figure 3, we compared the outcomes of only one-time editing results with those resulting from iterative editing using our iterative editing-optimization strategy. The results reveal significant color discrepancies in the one-time edit outcomes across different viewpoints, directly impacting the overall quality of the 3D model editing, as shown in the one-time update results in the figure "Comparison with other 3D scene editing methods" in the main text. In contrast, our iterative editing-optimization strategy substantially enhances the consistency of the editing results.

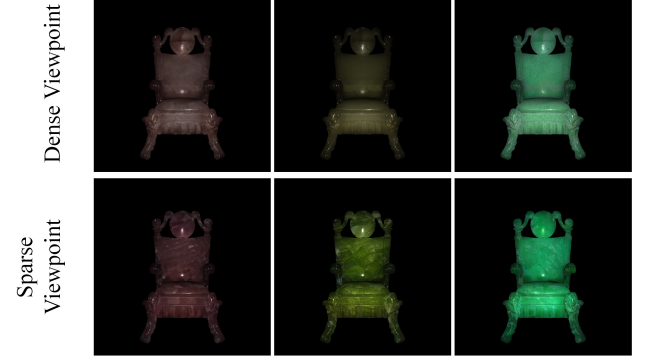


Figure 2: Ablation of viewpoint selection. We compare our results with those of editing using a dense uniform viewpoint sampling approach.

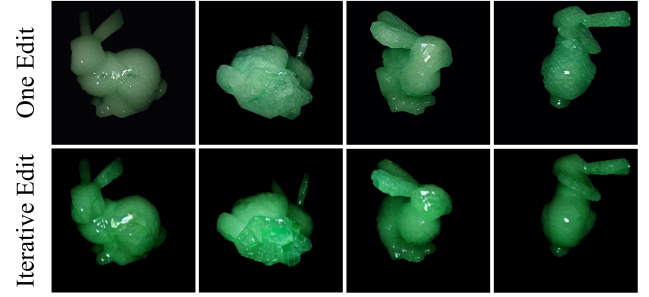


Figure 3: Ablation study for only edited results. We conducted an ablation study on the editing outcomes from multiple viewpoints, comparing the results of one-time editing with those obtained through iterative editing using our iterative editing-optimization strategy.

4 ADDITIONAL RESULTS

In this section, we show the results of editing experiments on additional models and material images, along with their re-rendered outcomes from different viewpoints.

Figure 4 shows the editing results of different 3D models including a bunny, armadillo, skull, chair, Samoyed, and girl, using the amethyst material. We validate the robustness of our method in 3D model editing by transferring the same single material across various models. Simultaneously, examining the editing outcomes from different viewpoints verifies that our iterative editing-optimization strategy effectively ensures consistency of material editing for heterogeneous translucent materials across multiple viewpoints.

Figure 5 shows the editing results of a 3D chair model using five different materials: amethyst, apophyllite, calcite, jade, and prehnite, demonstrating the effects from various viewpoints. In Figures 5, we demonstrate the editing results of a single 3D model with different materials, validating the robustness of our approach to 2D translucent material images.

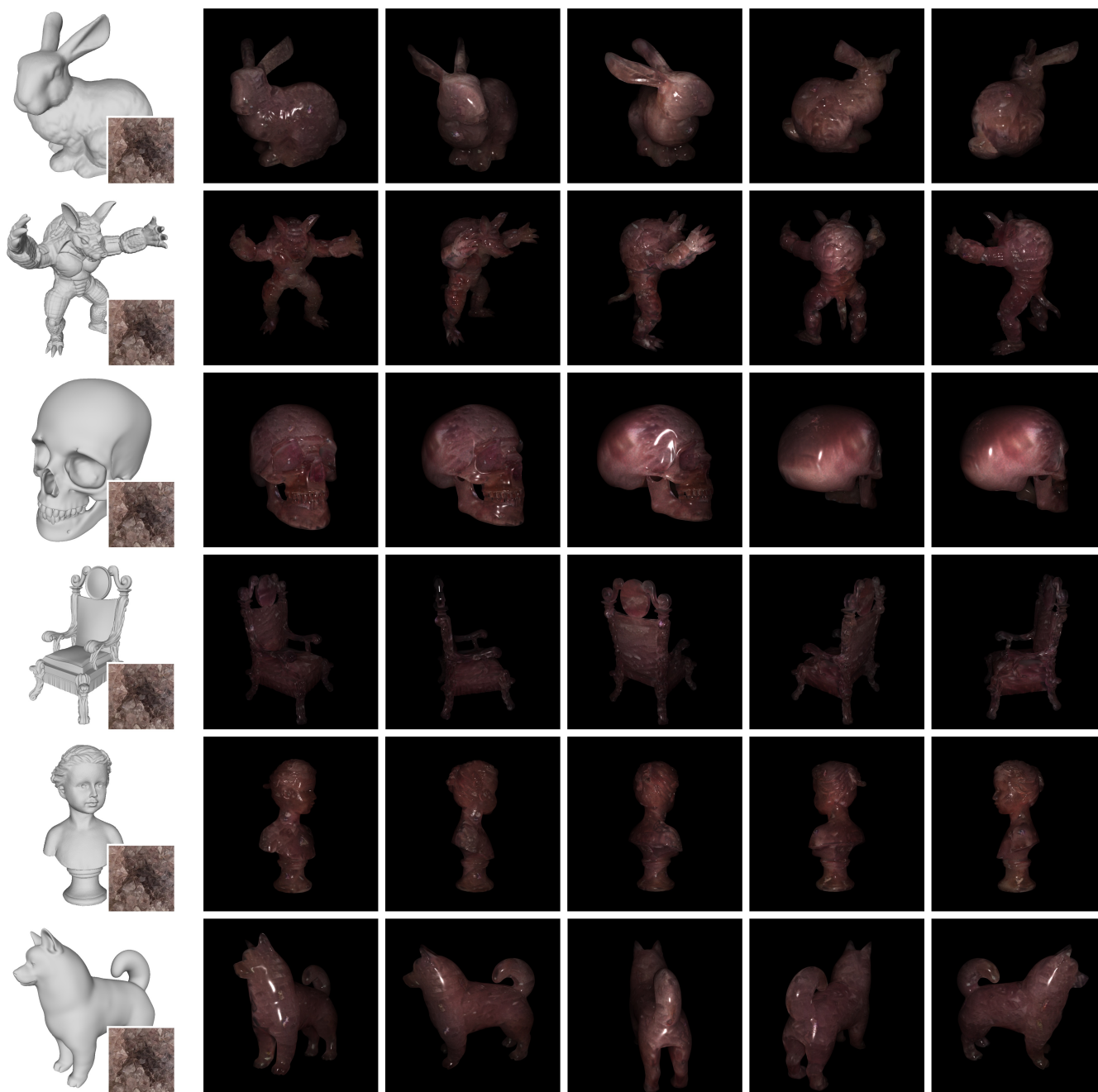


Figure 4: Different models with amethyst.

REFERENCES

- [1] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. 2023. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. *arXiv preprint arXiv:2303.13873* (2023).
- [2] Ayaan Haque, Matthew Tancik, Alexei A Efros, Aleksander Holynski, and Angjoo Kanazawa. 2023. Instruct-nerf2nerf: Editing 3d scenes with instructions. *arXiv preprint arXiv:2303.12789* (2023).
- [3] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. 2022. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988* (2022).
- [4] Elad Richardson, Gal Metzer, Yuval Alaluf, Raja Giryes, and Daniel Cohen-Or. 2023. Texture: Text-guided texturing of 3d shapes. *arXiv preprint arXiv:2302.01721* (2023).

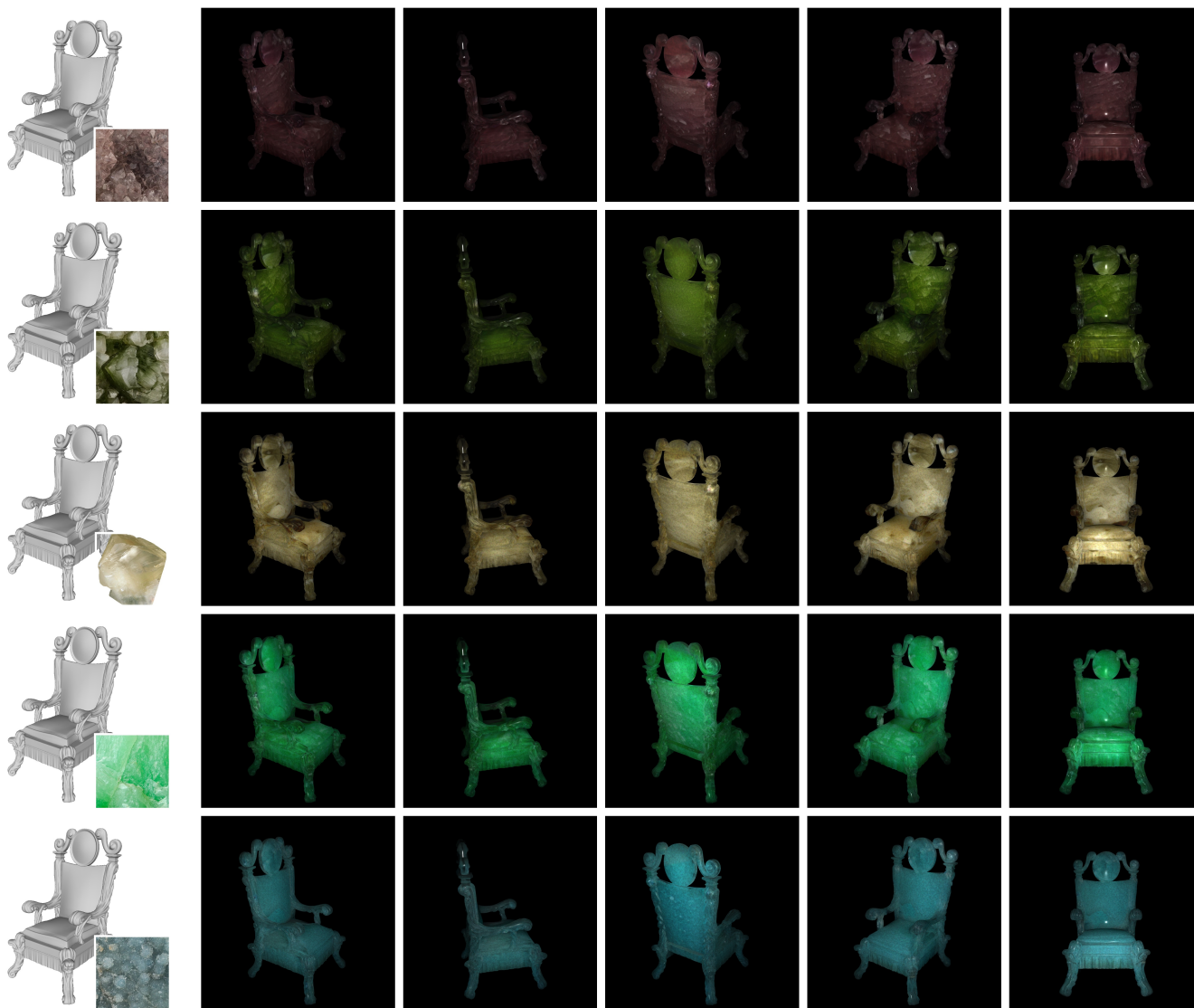


Figure 5: chair with different material images.