Uni-Renderer: Unifying Rendering and Inverse Rendering Via Dual Stream Diffusion

Supplementary Material

1. Appendix / supplemental material

In this supplementary, we will first discuss the detailed network architecture and the detailed algorithm for calculating different timesteps for reducing the tasks spaces. Then we will provide a description of the configurations used for baseline comparison. We also include more qualitative cases to demonstrate the capacity of our framework to perform smooth rendering and inverse rendering.

1.1. Parallel Stream Diffusion

The design of our framework involves two parallel stable diffusions. The upper branch takes in channel-concatenated attributes. Its UNet input "$Conv\_in$" and output "$Conv\_out$" layers are modified and extended to 24 channels for the corresponding input and output latents. The lower branch remains unchanged. The communication between the upper and lower branch are implemented through a cross-connected manner. We first take the intermediate feature "$mid\_block\_res\_samples$" from the upper encoder and add it to the lower decoder through a zero convolution layer. We do the same for the lower encoder. Such design effectively enables the communication between two stable diffusions in a cross-conditioned manner. The introduction of the zero convolution layer maintains the pertaining weight not get disrupted during training.

1.2. Model Training

During training, we adopted the x0 prediction into our loss calculation. It effectively helps to solve the channel bandwidth overhead problem. The training of diffusion models is performed on eight A800 GPUs, with a batch size of 4, a learning rate of $1e{-}5$, and a total training iteration number of 150,000. We utilize the Adam optimizer with $adam\_beta1$ and $adam\_beta2$ equal to 0.8 and 0.999, respectively.

1.3. Modeling conditional distributions with Two Timesteps

To achieve rendering and inverse rendering within a single model, we introduced a reduced timestep strategy to eliminate redundant tasks and thereby speed up convergence time and generation quality. In algorithm 1, we showed the pseudo-code for generating different timesteps.

1.4. Configuration for rendering baseline comparison

We show comparison against GAN-based material editing [3], Null-text inversion with prompt-to-prompt [2], InstructPix2Pix [1], and InstructPix2Pix prompt-only version trained on our data. For rendering baseline comparison, we first performed inverse rendering to acquire the necessary intrinsic attributes and used those to re-render with swapping attributes. By doing this, we ensured our setting was the same as other material editing pipelines. Next, we will go over each of the baselines, and briefly discuss the testing configuration for each of the methods.

Algorithm 1 Compute time steps matrix

Require: len_t Length of timesteps matrix, defaults to 2 in our case.
Require: $num\_timesteps$ Number of timesteps, ranging from 0 to T.
Require: bs Batch size
$timesteps \leftarrow$ intialize a zero matrix of size $len\_t \times bs$
$idx \leftarrow$ random integer from 0 to $len\_t - 1$
$all\_t[idx] \leftarrow$ random integers from 0 to $num\_timesteps - 1$ for each column
for $i \leftarrow 0$ to $len\_t - 1$ **do**
  if $i \neq idx$ **then**
    for $j \leftarrow 0$ to $bs - 1$ **do**
      $all\_t[i][j] \leftarrow$ random choice of $\{0, num\_timesteps - 1\}$
    end for
  end if
end for
return $timesteps$

InstructPix2Pix with our data [1]. This method takes an input image and a text prompt for material editing. We compared our method with two versions of InstructPix2Pix: the finetuned version and the original version. For training, we finetuned the model using a training set of 300 objects and trained for 1000 epochs To create pairs for the image editing framework, we varied two attributes—roughness and metallicity—from 0 to 1 while keeping the other attribute at 0. Prompts are built as "make it more/less rough/metallic." For example, for the prompt "make it rougher," the roughness of the input and ground truth would be 0 and 1, respectively. For testing, we evaluated both the finetuned model and the original model on our validation set using the default settings: inference steps set to 100, text CFG set to 7.5, and image CFG set to 1.0.

Null text inversion [2]. This method, also a prompt-only version, first optimizes the null text embedding to recover

Figure 1. **More Qualitative on Inverse Rendering** We included more qualitative cases to demonstrate the ability of Uni-renderer to perform inverse. **Best viewed in color.**

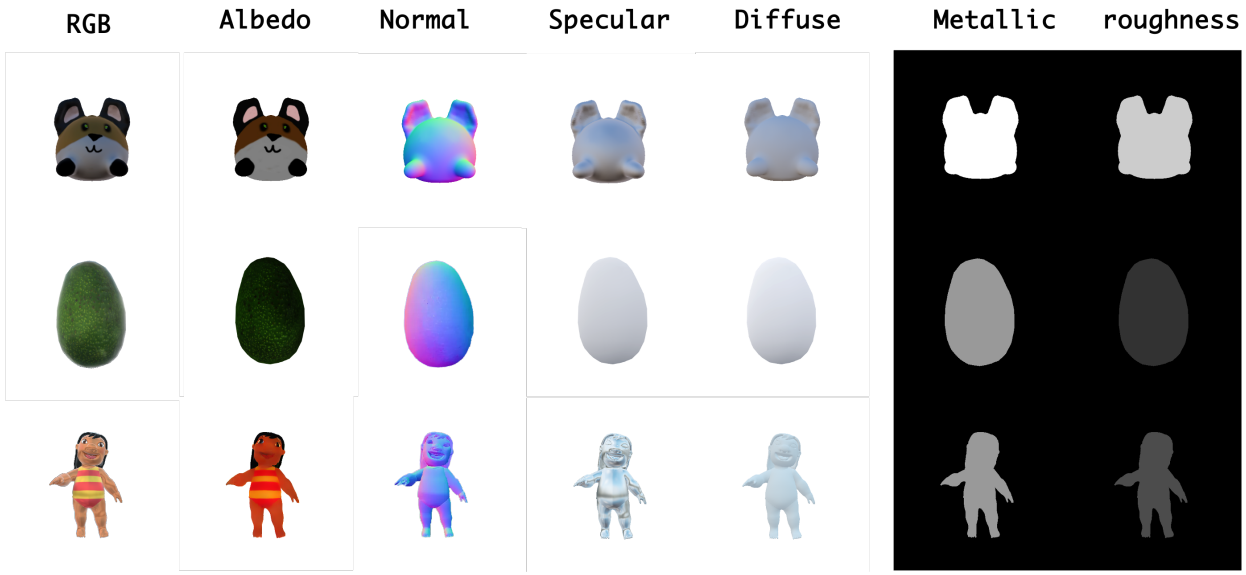

Figure 2. **More Qualitative on Inverse Rendering** We included more qualitative cases to demonstrate the ability of Uni-renderer to perform inverse. **Best viewed in color.**

the original DDIM latent sequences at inference with a high CFG value. It then performs prompt-to-prompt for image editing. For optimization, the steps are set to 300, and the prompt pairs used are in the format "a object name" to "a metallic/rough object name." For inference, the steps are set to 50, and the CFG value is set to 7.5.

Subias et. al [3] The method takes an image as input along with a scalar as input to perform relative material editing for glossiness and metallic. The method requires an input mask for localizing the object in the image. We generated the mask using the provided format input scripts. We input the image with the required transformation and tested it with a scalar of 1.0.

1.5. More Visual Samples

In this section, we will include more qualitative samples to further support the effectiveness of our method. We also include some cases to demonstrate the capacity of our frame-

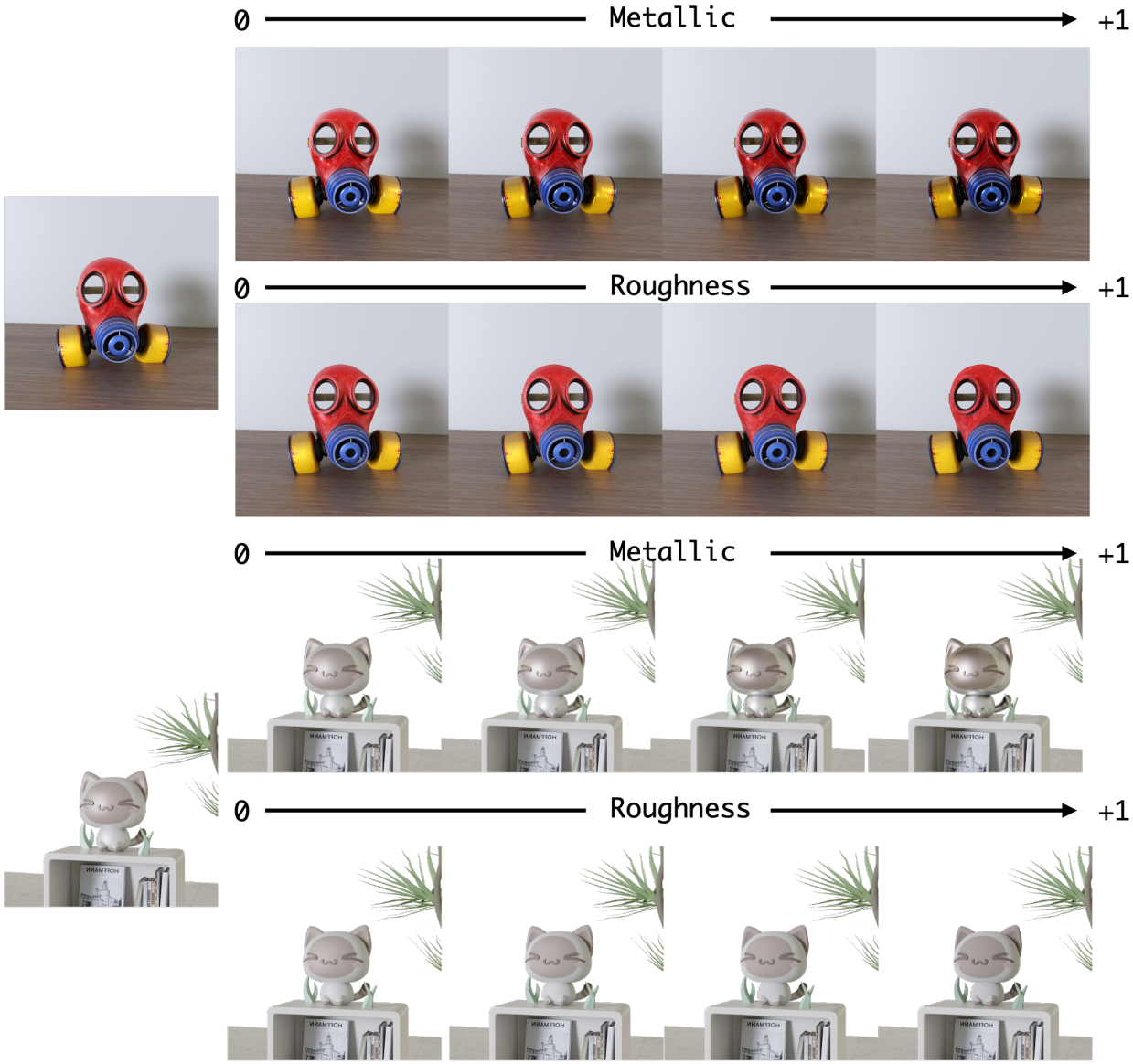

Figure 3. **More Qualitative on Accurate attributes editing** We included more qualitative cases to demonstrate the ability of Uni-renderer to perform rendering. The leftmost are reference images, and we have provided both increasing metallic and roughness. **Best viewed in color.**

work to perform smooth rendering.

References

[1] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions, 2023. 1

[2] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models, 2022. 1

[3] J. Daniel Subias and Manuel Lagunas. In-the-wild material appearance editing using perceptual attributes, 2023. 1, 2