Supplementary Material

The supplementary material is structured into the following sections:

- Sec. 6: Algorithmic Implementation Details
- Sec. 7: From NeRF Distance to Disparity Maps
- Sec. 8: How to Deal with Occlusions
- Sec. 9: Quantitative Results
- Sec. 10: Rationale for Noise Range Selection
- Sec. 11: Our Approach Using ControlNet [57] and Instruct-Pix2Pix [5]
- Sec. 12: Varying N in the Denoising Process
- Sec. 13: Janus Problem
- Sec. 14: Prompts Used for each Method

6 Algorithmic Implementation Details

We have included detailed pseudocode in this supplementary material for clarification purposes. These additions aim to provide a comprehensive understanding of the algorithms discussed in our paper, facilitating reproducibility and deeper insight into the implementation. Fig. 10 shows the pseudocode representations for the two main algorithms discussed in Sections 3.3 and 3.4 of the paper.

7 From NeRF Distance to Disparity Maps

In this section, we elaborate on the process of converting distance maps to disparity maps, aiming to provide a comprehensive understanding of the preparatory steps in our approach.

Pretrained, inpainting-aware diffusion models, as described in [39,57], necessitate the use of disparity maps for conditioning. These disparity maps, which have an inverse relationship to depth maps, are derived from the distance maps generated by Neural Radiance Fields (NeRF). Within the context of a NeRF framework, one can calculate the expected distance for each ray through a weighted sum of the distances of sample points along that ray. This computation is represented by the following equation:

$$\text{dist} = \frac{\sum_i w_i \cdot t_i}{\sum_i w_i + \epsilon} \tag{4}$$

Algorithm 1 Sec 3.3 Projection Inpainting

Input: Set of sequentially organized: images $\{I_2, I_3 \ldots, I_m\}$, masks $\{M_1, M_2, \ldots, M_m\}$, depths $\{D_1, D_2 \ldots, D_m\}$, the 1st edited image $\{I^e_{1=ref}\}$
Output: Edited images $\{I^e_2, I^e_3, \ldots, I^e_m\}$

for $k = 2$ to m **do** ▷ Iterate over the images
  $I^p_k, M^{vis}_k \leftarrow \text{Reproject}_{k-1\to k}(I^e_{k-1}, D_{k-1}, I_k)$ ▷ Reproject k-1 edited img
  $M^p_k \leftarrow M_k \times (1 - M^{vis}_k)$
  $z_1 \leftarrow \text{Encode}(I^p_k) + \epsilon$ ▷ Initialize noised latent repr.
  for $n = 1$ to $N - 1$ **do**
    $M \leftarrow$ **if** n ≤ 5 **then** M^p_k **else** M_k ▷ Choose mask based on step
    $\hat{\epsilon} \leftarrow$ Blended-Diffusion(ControlNet(z_n, D_k), M) ▷ Predict noise
    $z_{n+1} \leftarrow z_n - \hat{\epsilon}$ ▷ Update latent repr.
  $I^e_k \leftarrow \text{Decode}(z_N)$

Algorithm 2 Sec 3.4 Edited NeRF Optimization

Input: Edited images $\{I^e_1, I^e_2, \ldots, I^e_m\}$, masks $\{M_1, M_2 \ldots, M_m\}$, depths $\{D_1, D_2 \ldots, D_m\}$, original NeRF model
Output: Optimized NeRF model

$c = 0$, $T = 4000$, $S = 30$ ▷ Initialize counter, max iters, img generation
for iteration $i = 1$ to 1000 **do** ▷ NeRF update with all edited images
  Update NeRF with $\{I^e_1, I^e_2, \ldots, I^e_m\}$
for iteration $i = 1001$ to T **do** ▷ NeRF update with IDU
  if i % $S = 0$ **then**
    $z \leftarrow \text{Encode}(I^e_c) + \epsilon$ ▷ Initialize noised latent repr.
    $I^e_c \leftarrow$ Decode(Blended-Diffusion(ControlNet(z, D_c), M_c))
    $c \leftarrow (c + 1)$ % m ▷ Move to the next image in a cyclic manner
  Update NeRF with $\{I^e_1, I^e_2, \ldots, I^e_m\}$

Fig. 10: Pseudocode of DATENeRF. Left. Section 3.3 and Right. Section 3.4

In this equation, w_i denotes the weight assigned to the i-th sample point on the ray, t_i represents the distance of the sample along the ray, and ϵ is a small constant added to prevent division by zero. To translate this into a depth map, one must project these distances onto the camera's Z-axis, as shown in the equation below:

$$\text{depth} = \mathbf{R_z} \cdot (\mathbf{D} \cdot \text{dist}) \tag{5}$$

Here, $\mathbf{R_z}$ symbolizes the camera's perspective along the Z-axis, while $\mathbf{D}$ is the directional vector of the ray. Subsequently, the disparity map is obtained by inverting the values of the depth map:

$$\text{disparity} = \frac{1}{\text{depth} + \delta} \tag{6}$$

In this final step, δ is a small constant added to the depth to avoid division by zero when the depth is very close to zero. This procedure ensures that the resulting disparity map is accurately formatted for the pretrained models' requirements.

For specific scenes such as *person-small* and *fangzhou-small*, we impose a clamping range on the depth maps between $[1, 5]$. This adjustment is necessary because the distance maps produced by NeRF for these scenes exhibited significant artifacts, particularly in modeling the depth of walls located behind the subjects. By applying this range limitation, we effectively mitigate these artifacts, ensuring a more accurate and reliable disparity map for further processing.

8 How to Deal with Occlusions

In this section, we aim to clarify our method for addressing occlusions during mask extraction. Initially, we employ Grounded-SAM [22, 27], which provides a reliable mask for various objects, such as clothes, persons, t-shirts, bears, tables, etc. However, these masks can occasionally present issues. For example, in some frames, SAM may not detect the object, or it might incorrectly include unwanted objects, or it may only capture parts of the object. An instance of this is when the table label is used for segmentation, and it fails to accurately segment the table's legs, as shown in Fig. 11(b).

To mitigate these challenges, we introduce a straightforward step based on the assumption that "the majority of the masks are sufficiently accurate." We utilize the depth information from each view to verify if the 3D points within the masks consistently align across a significant percentage of the images, ensuring the inclusion of only the desired object (Fig. 11(c)). The subsequent stage involves projecting this refined point cloud back onto the image plane to obtain the final masks.

At this juncture, addressing occlusions becomes crucial, as the point cloud now represents the entire object. Our resolution involves a simple yet effective strategy: leveraging the depth information from NeRF and the reprojected point cloud, which provides the z-axis distance in camera coordinates. By prioritizing

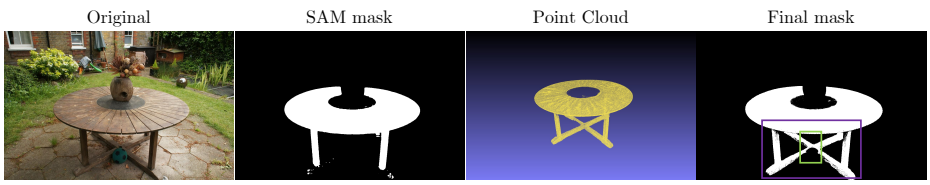

Fig. 11: Mask Extraction and Refinement Process. From left to right. (a) Original image of a scene with a table. (b) Initial SAM mask providing a coarse outline of the object. (c) Derived point cloud representing the geometric structure of the table. (d) Final mask after applying occlusion handling and refinement, with the legs of the table segmentation indicated by the purple square, and ball object occlusion tackled by the green square.

points closest to the camera and disregarding those situated behind, we efficiently manage occlusions, ensuring that our final masks accurately represent the target object, as depicted in Fig. 11(d).

9 Quantitative Results

We align our evaluation metrics with those reported in [13], focusing on two specific metrics: the CLIP Directional Score and the CLIP Direction Consistency Score. The Directional Score is designed to assess how well changes in textual descriptions correlate with corresponding changes in images. In contrast, the Consistency Score evaluates the cosine similarity of CLIP model embeddings for sequential frames. Both are computed while following a new camera path in rendering, meaning that we use the test set for each scene.

For the Directional Score, we use paired images (original and modified, viewed from the same perspective) and corresponding text prompts that describe each scene. This approach enables a precise comparison of text-image alignment, reflecting how well the modified image adheres to the new textual description.

Regarding the Consistency Score, our analysis involves examining consecutive frames along a novel trajectory (test set). We compare the original NeRF with its modified counterpart, leading to four distinct CLIP embeddings: two from the original rendering and two from the modified one. The Consistency Loss, defined as the cosine similarity between the changes in embeddings from one frame to the next, quantifies the directional consistency of edits in the CLIP-space across frames.

The formula for consistency loss, as detailed in [13], is:

$$\cos\_\text{sim} = \frac{(C(e_i) - C(o_i)) \cdot (C(e_{i+1}) - C(o_{i+1}))}{\|C(e_i) - C(o_i)\| \|C(e_{i+1}) - C(o_{i+1})\|} \tag{7}$$

In this equation, $C(e_i)$ and $C(e_{i+1})$ represent the CLIP embeddings of the edited rendering at frames i and $i+1$, respectively, while $C(o_i)$ and $C(o_{i+1})$

correspond to those of the original rendering. This measurement effectively captures the consistency of the directional changes in CLIP-space from one frame to the next.

These metrics have been applied to the *face*, *bear*, and *person* scenes, utilizing a diverse set of 24 prompts for evaluation. For our evaluation metrics, we have opted to utilize masks with Instruct-NeRF2NeRF. This approach is necessitated by the fact that Instruct-NeRF2NeRF's global editing capabilities can cause some prompts to trigger modifications beyond the intended object. Measuring the quality of these edits on a scene-wide scale could skew our metrics, leading to a misrepresentation of the precision of our object-specific edits. By employing masked images, we ensure that our metrics are specific to the edits of interest, thereby providing a more accurate assessment of our approach's performance in targeted scene editing.

10 Rationale for Noise Range Selection

The justification for our decision to employ a narrower noise range, specifically $[\mathrm{t_{min}}, \mathrm{t_{max}}] = [0.8, 0.98]$, in our approach, referred to as "*Ours without Projection*," in contrast to the broader range of $[\mathrm{t_{min}}, \mathrm{t_{max}}] = [0.02, 0.98]$ utilized in Instruct-NeRF2NeRF, is rooted in our observation that the latter struggles to align with the given text prompt during training.

Our empirical results, as depicted in Fig. 12, reveal that employing a wide spectrum of noise levels can significantly impede the convergence of the model. In contrast, the specific noise parameters we have selected ensure that NeRF training aligns effectively with the provided text prompt.

Our underlying intuition here lies in the nature of the diffusion model employed by Instruct-NeRF2NeRF (Instruct-Pix2Pix [5]), which is designed to preserve the identity of the input image. Consequently, an increase in noise within the image results in a gradual transition from the original image to a blend with the generated one.

However, in the case of ControlNet, the concept of preserving image identity is not a central concern. Therefore, varying the noise levels does not necessarily imply a gradual blending of the generated image with the original. Instead, it tends to make the generated image closely resemble the provided text prompt without the gradual transition characteristic of Instruct-Pix2Pix.

11 Our Approach Using ControlNet [57] and Instruct-Pix2Pix [5]

We further demonstrate the versatility of our approach by applying it to different editing models. In Fig. 13, we showcase two illustrative examples in which our projection inpainting technique has been employed. The first case depicts "Benjamin Franklin", while the second is "An old lady" trnaformations. In both instances, the results maintain a remarkable level of quality, evidencing the robustness of our method. This adaptability is one of the most notable strengths of our approach, enabling its application across various diffusion models without

Fig. 12: The effect of noise range. The effect of noise range on a bear scene using ControlNet for the prompt 'A husky' The Left image, with $[t_{\min}, t_{\max}] = [0.02, 0.98]$, shows a broad noise range where results do not converge as effectively, while the Right image, with $[t_{\min}, t_{\max}] = [0.8, 0.98]$, depicts a narrow noise range with better convergence of results.

sacrificing the efficiency and convergence characteristics that have been previously highlighted. Such adaptability not only broadens the potential use cases for our technique but also reinforces its practicality in a wide range of scenarios.

12 Varying N in the Denoising Process

We present results for the prompt "a corgi" within the context of the scene titled bear, comparing the outcomes when using $N = 20$ and $N = 5$ (our method) with our hybrid inpainting technique. The approach using $N = 20$ is less effective due to errors in the NeRF geometry that lead to reprojection artifacts. Furthermore, pixel propagation across significant viewpoint changes—particularly at oblique angles—results in suboptimal outcomes, primarily because of texture stretching. This issue is evident in Fig. 14 where $N = 20$ is contrasted with $N = 5$.

Fig. 13: Comparison between ControNet vs. Instruct-Pix2Pix using our approach. Both examples validate the adaptability of our projection inpainting technique across diverse diffusion models.

We have found that a more effective strategy involves using the reprojected pixels as a starting point for the diffusion-based editing process. Our novel hybrid inpainting scheme accomplishes this by retaining the reprojected pixels during the initial $N = 20$ denoising steps and then reverting to complete inpainting of the object regions in the subsequent denoising steps. These initial diffusion steps help to guide the overall appearance of the edit, while later stages allow the diffusion process to correct disoccluded areas, all the while maintaining the adaptability to amend reprojection artifacts.

13 Janus Problem

Our investigations revealed that ControlNet, like other diffusion models, is susceptible to the 'Janus problem.' This issue is characterized by the tendency of the model to generate facial features on the rear of objects, a phenomenon particularly noticeable with animals. Our approach initially faced the same challenge, as the projection process could mistake spots on an object, such as a panda, for a face in subsequent projections.

To overcome this, we devised a simple yet effective solution: querying the model with prompts such as "The back side of ...". Specifically for the bear scene, this strategy allowed us to successfully navigate the problem. This solution capitalizes on the depth conditioning employed by ControlNet, which cues the model to predominantly generate the back of an object when it is positioned accordingly, despite the provided depth. While one might assume that this would result in backside features appearing on the front, the model's inherent bias towards recognizing faces in the depth map prevents this from occurring. Our understanding of this bias has been instrumental in achieving accurate results.

For all instances labeled 'ours' within the bear scene, we utilized the prompt "the backside of ...". We considered applying this technique to "Ours without projection"; however, the only instance where it proved beneficial was with the panda, which we have highlighted in Fig. 6 Convergence Speed, in the main manuscript.

14 Prompts Used for each Method

In Table. 2, we show a detailed account of the prompts utilized, as InstructPix2Pix [5] and ControlNet [57] need distinct prompting approaches.

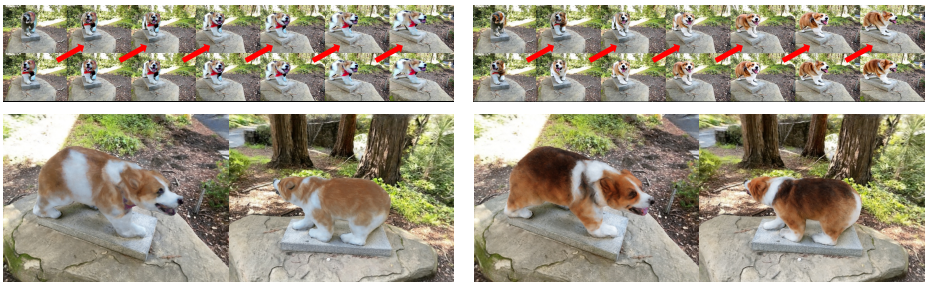

Fig. 14: Comparison between $N = 20$ and $N = 5$. We illustrate our hybrid inpainting technique applied to a scene labeled *bear*, featuring a corgi. We compare the hybrid scenario with $N = 5$ against the use of inpainting alone with $N = 20$. The top series of images display the progression of reprojected views (top: reprojected images; bottom: generated images – please zoom in for detail). The results for both approaches are depicted at the bottom. $N = 5$ showcases our refined method, where the initial denoising steps effectively guide the edit.

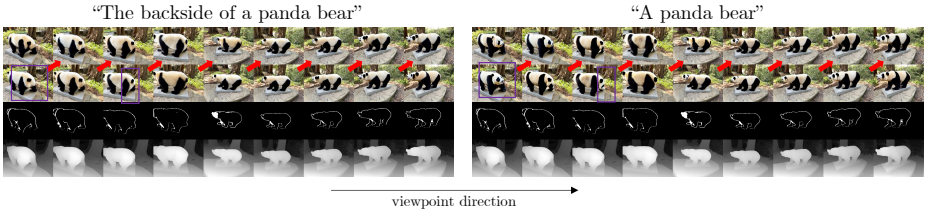

Fig. 15: Janus Problem. An example of addressing the Janus problem using different prompts. On the left, "The backside of a panda bear" successfully avoids generating a face on the rear, as highlighted within the purple square. On the right, "A panda bear" serves as a comparison prompt. From left to right, the sequence depicts the change in viewpoint. Top to bottom displays the reprojected images, generated images, then to the reprojected masks, and finally to the depth maps.

Table 2: Prompts. Comparison of prompts used for Instruct-Pix2Pix [5] (IP2P) and ControlNet [57] .

Scene	Original Prompt	Method
person-complete	Turn him into [...]" "Michael Jackson"	IP2P ControlNet
	"Turn the man into [...] "Iron Man arc reactor" / "Spiderman" / "Mario Bross clothes"	IP2P ControlNet
	"Turn the clothes of the man into [...]" "A tuxedo with a red tie bow"	IP2P ControlNet
person-small	"Turn the t-shirt of the man into [...]" "the Kentucky Fried Chicken man" / "an sleeveless shirt with the lakers word stamped on it"	IP2P ControlNet
table	"Turn the table into [...]" "a billiard table" / "sunflower-painted table" / "a starry night canvas" / "a Fauvism-style table" / "Black and White Checkered Pattern Table"	IP2P ControlNet
face	"Turn his clothes into [...]" "a tuxedo with a flower in the lapel" / "a red buffalo plaid shirt"	IP2P ControlNet
	"Make this clothes like [...]" "Superman clothes"	IP2P ControlNet
face-complete	"Turn him into [...]" "a clown" / "Hulk, the green superhero" / "an old lady" / "Matthew Mcconaughey smiling" / "Andy Warhol" / "Benjamin Franklin"	IP2P ControlNet
furry	"Turn the teddy bear into [...]" "Winnie the Pooh" / "a racoon" / "a red panda" / "a panda bear" / "a grizzly bear "	IP2P ControlNet
furry	"Turn the bear into a [...]" "tiger" / "zebra" / "sharpei" / "corgi" / "polar bear " / "panda bear" / "grizzly bear" / "wild African dog" / "husky"	IP2P ControlNet