DOI: 10.1111/cgf.70329

EUROGRAPHICS 2026 / B. Masia and J. Thies
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 45 (2026), Number 2

PBR-Inspired Controllable Diffusion for Image Generation

Bowen Xue[1] Giuseppe Claudio Guarnera[2] Shuang Zhao[3] Zahra Montazeri[1]

[1]University of Manchester, UK [2]University of York, UK [3]University of Illinois Urbana-Champaign, USA

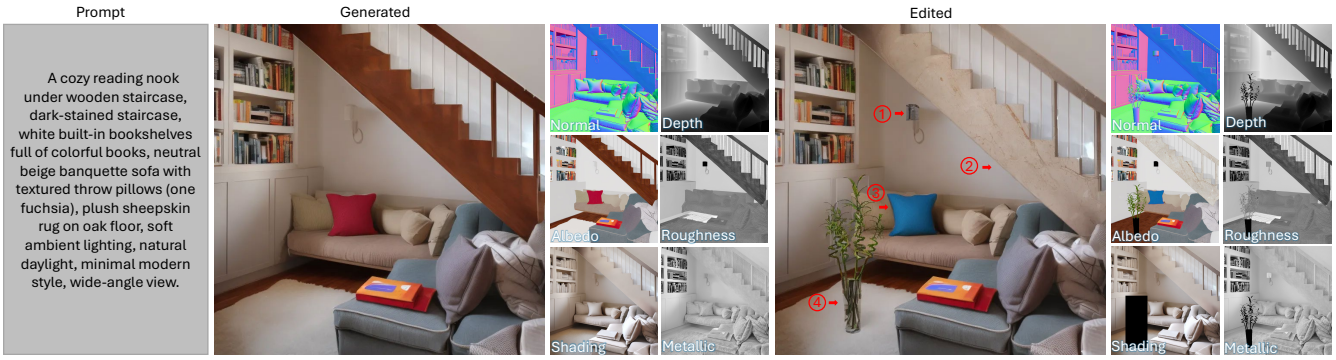

Figure 1: *We propose a controllable text-to-image pipeline that begins by generating a* G-buffer *(geometry buffer) for any given text prompt. The G-buffer encodes per-pixel geometry and material channels such as albedo, surface normals, depth, roughness, metallic, and shading. The left shows the input text prompt, the middle shows our generated result, and the right demonstrates channel-wise control to the predicted G-buffer. Users can control specific channels to achieve arbitrary edits such as surface property adjustments (enhancing the metallic quality of the wall lamp ①), material replacements (wooden staircase to marble ②), color modifications (changing the sofa cushion color from red to blue ③), and object insertion (adding a vase on the carpet ④).*

Abstract

Despite recent advances in text-to-image generation, controlling geometric layout and PBR material properties in synthesized scenes remains challenging. We present a pipeline that first produces a G-buffer (albedo, normals, depth, roughness, shading, and metallic) from a text prompt and then renders a final image through a PBR-inspired branch network. This intermediate representation enables fine-grained control: users can copy and paste within specific G-buffer channels to insert or reposition objects, or apply masks to the irradiance channel to adjust lighting locally. As a result, real objects can be seamlessly integrated into virtual scenes. By separating user-friendly scene description from image rendering, our method offers a practical balance between detailed post-generation control and efficient text-driven synthesis. We demonstrate its effectiveness through quantitative evaluations and a user study with 156 participants, showing consistent human preference over strong baselines and confirming that G-buffer control extends the flexibility of text-guided image generation.

CCS Concepts

• ***Computing methodologies*** → ***Reflectance modeling;*** *Image-based rendering;*

1. Introduction

The ability to generate and edit photorealistic scenes with precise control over PBR material properties remains a challenge in computer graphics. While recent text-to-image diffusion models [RPG*21; SCS*22; RBL*22] excel at generating visually stunning images, they operate as end-to-end mappings that obscure the underlying scene structure. This black-box nature fundamentally limits their utility in professional workflows where artists need fine-grained control over specific scene elements—adjusting material properties, modifying lighting, or editing geometry—without regenerating the entire image. For a visual preview of our controllable pipeline and edits, see Figure 1.

This limitation stems from a structural mismatch: the direct text-to-RGB mapping entangles geometry, materials, and lighting, making precise, channel-wise control impractical. Current diffusion models learn a direct mapping from text to RGB pixels, en-

tangling geometry, materials, and lighting into a single, inseparable representation. When users attempt to modify prompts for targeted edits (e.g., "make the table metallic"), the model cannot disentangle which pixels correspond to the table's material properties versus its geometry or the surrounding lighting. Recent efforts to enhance controllability have taken two divergent paths, each with fundamental limitations. Structural control methods (ControlNet [ZRA23], T2I-Adapter [MWX*24]) successfully inject spatial conditions but operate at the image level, missing the rich semantic structure of scene properties. Conversely, neural-based decomposition methods (RGB↔X [ZDG*24], neural rendering approaches [LGL*25]) attempt to recover scene structure but specialize the diffusion prior to the indoor domain via fine-tuning on relatively small indoor datasets. RGB↔X also supports editing and relighting capabilities. This division suggests a critical gap: can we design an architecture that combines the generative power of pretrained diffusion models with explicit PBR scene decomposition that allows detailed modification?

Our key insight is that this integration becomes feasible by carefully considering where and how to inject structure into the diffusion process. We identify two critical design decisions. First, we recognize that the latent space of diffusion models offers unique advantages for structured generation. Unlike image-space methods that must encode and decode through information-losing transformations, latent-space operation preserves the rich distributional structure learned during pretraining. This observation motivates our Latent ControlNet—a modified architecture that directly processes latent representations to generate G-buffers without intermediate image decoding. This enables the generation of consistent multi-channel outputs (albedo, normal, depth, roughness, metallic, shading). Second, we observe that the rendering equation naturally suggests a branch architecture for the inverse problem. Rather than learning a plain end-to-end mapping from G-buffers to images, we design a network that mirrors the PBR image formation process: separate pathways for geometry (normals, depth), materials (albedo, roughness, metallic), and lighting (shading). This factorization provides an inductive bias that improves learning efficiency—our experiments reveal a 76% reduction in reconstruction error compared to end-to-end alternatives.

These insights lead to a two-stage framework that bridges generative modeling and PBR-inspired rendering. We separate scene structuring from image rendering: a Latent ControlNet first produces a multi-channel G-buffer, and a PBR-inspired branch renderer then maps it to RGB, enabling predictable, channel-wise control. Stage one generates complete G-buffers from text prompts using our Latent ControlNet, while stage two renders these buffers into images through our branch network. Crucially, this decomposition enables intuitive post-generation editing: users can directly manipulate individual channels with predictable, localized effects on the final image.

In summary, our contributions include:
(1) Latent ControlNet Architecture: A Latent ControlNet that operates directly in diffusion latent space, enabling multi-channel G-buffer generation while preserving information fidelity. (Sec. 3)
(2) PBR-Inspired Branch Renderer: A branch network that mirrors the image formation process through separate geometry, material, and lighting pathways, achieving superior reconstruction.(Sec. 4)
(3) Practical Editing Framework: A system to enable intuitive, PBR-inspired editing of diffusion-generated images through G-buffer manipulation, validated through extensive user studies.

2. Related Work

Text-to-Image Generation. Text-to-image generation has advanced considerably through GANs and diffusion models. GAN-based methods demonstrated the feasibility of synthesizing images from textual descriptions but often suffered from low resolution and limited semantic alignment [RAY*16; ZXL*17]. Attention-based architectures improved the correlation between text embeddings and visual content [XZH*18]. In parallel, diffusion models offered better coverage of the data manifold and reduced mode collapse [DN21], leading to large-scale systems like DALL·E [RPG*21] and GLIDE [NDR*22], which introduced classifier-free guidance. Latent diffusion approaches enabled high-resolution outputs at reduced cost [RBL*22], forming the basis of Stable Diffusion, while Imagen further improved photorealistic generation through cascaded diffusion [SCS*22]. Despite these advances, post-generation editing of specific scene elements remains challenging.

Screen-Space Rendering. The G-buffer concept, originally introduced by Saito and Takahashi [ST90] for comprehensible rendering of 3D shapes, has become fundamental to deferred shading pipelines. Screen-space rendering bridges purely 2D methods and full 3D reconstructions, capturing partial spatial data via layered depth images or G-buffers [BBM*01; PZ17]. This enables moderate viewpoint changes and scene manipulation without the complexity of volumetric geometry. For instance, [HPP*18] supported free-viewpoint navigation with multi-view blending.

Neural Rendering. Neural rendering synthesizes novel views or edited scenes from volumetric or surface-based data, bridging computer graphics and machine learning. Hierarchical neural materials improve multi-scale SVBRDF fidelity [XZJM24]. NeRFs [MST*21] first demonstrated high-fidelity view synthesis later extended to dynamic scenes [PSB*21], and anti-aliased systems for unbounded environments [BMV*22]. Surface-based methods [LSS*19; TTG*20] utilize explicit geometry or point-based representations for realistic rendering but often demand substantial data and computation, complicating fine-grained edits. DiffusionRenderer [LGL*25] proposes a combined framework for both neural inverse and forward rendering using video diffusion models.

Neural Image Synthesis from G-buffer. This line of work builds on foundational intrinsic image decomposition [BT78; BBS14] and PBR-based shading models [Bur12]. Several approaches learn image synthesis from G-buffers or intermediate decompositions. Deep Shading [NAM*17] infers effects like ambient occlusion and subsurface scattering via CNNs. [ZLH*22a] propose screen-space ray tracing from intrinsic channels, and RGB↔X [ZDG*24] fine-tunes a Stable Diffusion model to render intermediate decompositions.

Editing and Relighting. Neural relighting can rely on explicit representations [GRP22; PEL*21; YME*20] or implicit

ones [RES*22; WSG*23], typically limited to simpler lighting conditions. Recent works have explored diffusion models for lighting control and intrinsic decomposition. DiLightNet [ZDP*24] provides fine-grained lighting control for diffusion-based image generation, while LightIt [KPS*24] uses light maps to control image generation. "Intrinsic Image Diffusion" [KSN24] and IntrinsicDiffusion [LCY*24] both employ diffusion models for intrinsic image decomposition. Diffusion-handles [PGG*24] applies a diffusion-based model for object manipulation. In recent years, a variety of extensions have been proposed to improve controllability of diffusion model, yet previous works either inject structural conditions (ControlNet [ZRA23], T2I-Adapter [MWX*24], GLIGEN [LLW*23]) but lack fine-grained material and lighting control, or enable local editing (Prompt-to-Prompt [HMT*22], DiffEdit [CVSC22], ZONE [LZF*24]) and concept insertion (DreamBooth [RLJ*23], AnyDoor [CHL*23]) but remain limited to object-level modifications without true channel-wise refinement capabilities. DiffusionRenderer [LGL*25] leverage diffusion models for neural rendering which offers unified inverse–forward framework for temporally consistent video decomposition and re-rendering of existing content; however our work embeds rendering knowledge into generative models for text-driven control over PBR properties to create controllable content from scratch. IntrinsicEdit [LDH*25] proposes a training-free optimization framework built on top of pre-trained RGB↔X models. While IntrinsicEdit focuses on editing existing images, our work trains a dedicated text→G-buffer→RGB backbone for generating new content. These approaches are orthogonal: IntrinsicEdit could be applied on top of our backbone as a complementary editing layer. Recent general-purpose editing models such as Flux [LBB*25] and Qwen-Edit [WLZ*25] achieve impressive results but operate directly on RGB without exposing intermediate scene representations, targeting a different use case from our channel-wise controllable generation.

3. Text-to-G-Buffer Generation via Latent-Space ControlNet

In this section, we present our two-phase, diffusion-based network integrating a latent-space ControlNet for direct G-buffer generation from text prompts. The entire pipeline is depicted in Figure 2.

Our dataset is several orders of magnitude smaller than the original Stable Diffusion training dataset and focuses predominantly on indoor scenes. Fine-tuning the diffusion model in such a narrow subset can result in overfitting and catastrophic forgetting, as shown in Figure 3.

To generate G-buffers without overwriting the pretrained generative mapping, we adopt a two-phase network design similar to [XGZM24]. In phase 1, we use a frozen diffusion model to preserve the pretrained generative capabilities. In phase 2, we introduce a latent-space ControlNet structure rather than retraining the diffusion model itself. Toward the last 5 epochs of training, we unfreeze the main diffusion model, further improving G-buffer results.

In phase 2, directly fine-tuning the stable diffusion on a small dataset D needs to relearn the entire G-buffer mapping by solving

$$\theta^* = \arg\min_{\theta} \frac{1}{|D|} \sum_{(x,y)\in D} \ell\big(F(x;\theta), y\big), \tag{1}$$

where Ïll ÏNet parameters Ïre Ïeing updated. Ïn Ïontrast, Ïe retain the pretrained UNet backbone $\mathcal{F}^*(x) = F(x;\theta^0)$ (with frozen weights θ^0) and learn only a lightweight ControlNet g_ϕ via

$$\phi^* = \arg\min_{\phi} \frac{1}{|D|} \sum_{(x,y)\in D} \ell\big(g_\phi(\mathcal{F}^*(x)), y\big). \tag{2}$$

Here ϕ denotes parameters of g_ϕ. This formulation preserves the backbone's original generative diversity, yielding better sample efficiency and reduced overfitting compared to full fine-tuning.

By preserving the pretrained parameters and introducing a relatively small ControlNet, we retain the latent function and confine new adaptations to a lower-dimensional space. This strategy mitigates catastrophic forgetting and improves generalization on small-data tasks compared to fully tuning all parameters. As shown in Figure 3. Let $\mathcal{E}$ denote the control net encoder, $\mathcal{D}$ the diffusion model VAE decoder. The standard ControlNet pipeline is:

$$z_c \xrightarrow{\mathcal{D}} x_c \xrightarrow{\mathcal{E}} z_c' \xrightarrow{g} \hat{f}_c \tag{3}$$

where z_c is the latent representation control signal, x_c is the control image, g is the standard ControlNet. Our architecture simplifies to:

$$z_c \xrightarrow{g_\phi} \hat{f}_c \tag{4}$$

By operating directly in the latent space, we avoid an additional decode–encode step that may discard information. As qualitative motivation, the data processing inequality [Cov99] for the Markov chain $z_c \to x_c \to z_c'$ gives:

$$I(z_c; Y) \geq I(z_c'; Y) \tag{5}$$

where Y represents the target G-buffer and $I(\cdot;\cdot)$ denotes mutual information. This suggests that the additional decode-encode cycle cannot increase information about the target.

Furthermore, the decoder-encoder cycle $\mathcal{D} \to \mathcal{E}$ may discard latent information that is not explicitly represented in RGB space but is crucial for G-buffer generation. For instance, the latent code z_c may contain implicit geometric or material properties that are lost when projected to RGB and cannot be fully recovered by $\mathcal{E}$.

Our Latent ControlNet, operating directly on z_c, avoids an additional decode–encode step that can discard information. Empirical validation of this design is provided in Figures 3 and 7. This design is particularly advantageous for G-buffer generation, where we need to extract structured scene information (geometry, materials, lighting) that may be implicitly encoded in the latent space but not fully visible in the RGB representation. Our method also reduces computational complexity by eliminating the control signal decoding and encoding phase and keeps representation consistency. When the control signal resides in the same latent space as the diffusion model, the conditional generation process becomes more direct.

4. G-buffer Rendering Network

Our G-buffer to image rendering pipeline leverages a fine-tuned Stable Diffusion model with a ControlNet backbone, which natively supports three-channel conditional inputs. However, our ControlNet input consists of 13 stacked channels: albedo (3), normal (3), roughness (1), shading (3), metallic (1), depth (1), and a mask (1) that indicates regions where new objects are inserted or

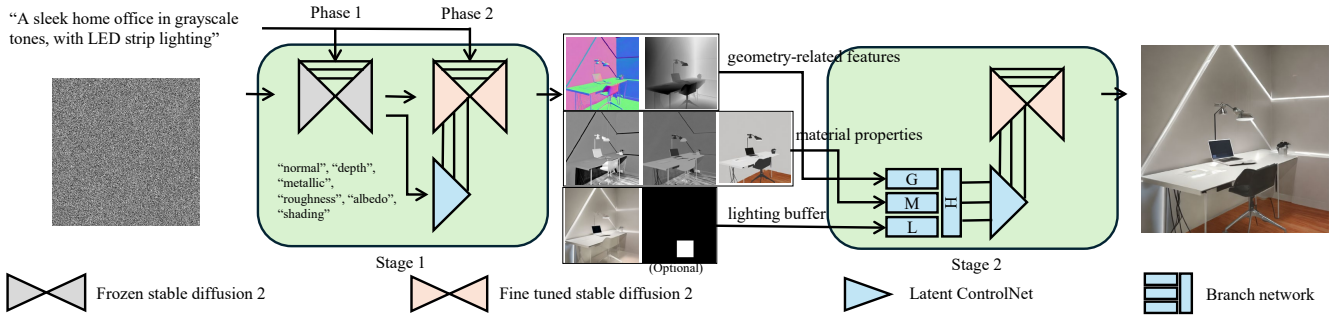

Figure 2: *Overview. Our pipeline begins with a random noise sample and a text prompt. These inputs are processed by the stage-1 network, which consists of two denoising steps: first, a frozen Stable Diffusion 2 model (in gray), second, a fine-tuned Stable Diffusion 2 model augmented with ControlNet. Stage 1 produces a G-buffer comprising albedo, normal, depth, shading, roughness, and metallic. These channels are then grouped and passed to the stage-2 network, where an optional mask is used for object movement or insertion. Each group is processed by specialized sub-networks, fused by a final grouping module, and then fed into another ControlNet-equipped, fine-tuned Stable Diffusion 2 model to generate the final RGB output.*

existing objects are moved. Simply concatenating these channels tends to cause poor performance and training instability; moreover, our extra channels form a complex, multi-component G-buffer rather than additional color images. Hence, we prepend a multi-layer CNN module to the input of ControlNet to extract low-level features from this multi-channel input. This strategy enhances compatibility with the original architecture and stabilizes training.

Single-Bounce Baseline. A simplified version of Kajiya's rendering equation [Kaj86] for a surface point p and outgoing direction ω_o (where f_r is a local BRDF, L_i is the incident radiance) can be written as:

$$L_o(p,\omega_o) = \int_\Omega f_r(p;\omega_i,\omega_o)\, L_i(p;\omega_i)\, (\mathbf{n}\cdot\omega_i)\, d\omega_i, \qquad (6)$$

For simplicity, we omit secondary bounces, subsurface scattering, and emissive effects.

Microfacet BRDF Decomposition. Under microfacet theory (Cook–Torrance [CT82]), the BRDF f_r factorizes into Fresnel F, normal distribution D, and a geometric masking term G:

$$f_r(\omega_i,\omega_o;\mathbf{x}_m,\mathbf{x}_g) \;=\; \frac{F(\mathbf{x}_m,\mathbf{h})\; D(\mathbf{x}_m,\mathbf{h})\; G(\mathbf{x}_g,\omega_i,\omega_o)}{4\,(\mathbf{n}\cdot\omega_i)\,(\mathbf{n}\cdot\omega_o)}, \qquad (7)$$

where $\mathbf{x}_m$ (material) includes roughness, metallic, and $\mathbf{x}_g$ (geometry) includes the macroscopic normal $\mathbf{n}$. Meanwhile, $L_i(\omega_i)$ may be regarded as a function of $\mathbf{x}_l$ (e.g., environment maps, shadow buffers). Substituting Eq. (7) into Eq. (6) gives:

$$L_o = \int_\Omega \frac{F(\mathbf{x}_m,\mathbf{h})D(\mathbf{x}_m,\mathbf{h})G(\mathbf{x}_g,\omega_i,\omega_o)}{4(\mathbf{n}\cdot\omega_i)(\mathbf{n}\cdot\omega_o)}\, L_i(\omega_i)\,(\mathbf{n}\cdot\omega_i)\, d\omega_i. \qquad (8)$$

Thus, geometry ($\mathbf{x}_g$), material ($\mathbf{x}_m$), and lighting ($\mathbf{x}_l$) appear in separate factors. This partial independence motivates a network design that processes these components in separate branches, often requiring fewer parameters and yielding more stable training.

G-buffers and Branch Networks. In practice, we store $\{\mathbf{n},\mathbf{d}\}$ in a *geometry buffer*, $\{\mathbf{A},r,m\}$ in a *material buffer*, and $\mathbf{x}_l$ (shading, shading mask) in a *lighting buffer*, where $\mathbf{n}$ denotes the normal, $\mathbf{d}$ the depth, $\mathbf{A}$ the albedo (base color), r the roughness, and m the metallic. The factorization in Eq. (8) provides theoretical justification for our branch network design. Instead of learning a monolithic function:

$$F : \big(\mathbf{n},\mathbf{d},\mathbf{A},r,m,\mathbf{x}_l\big) \;\mapsto\; \mathbf{L}_o,$$

we factor it into three sub-networks:

$$\mathbf{L}_o \;=\; \mathcal{H}\Big(\mathcal{G}\big(\mathbf{n},\mathbf{d}\big),\; \mathcal{M}\big(\mathbf{A},r,m\big),\; \mathcal{L}\big(\mathbf{x}_l\big)\Big), \qquad (9)$$

where $\mathcal{G}(\cdot)$ extracts geometry-related features, $\mathcal{M}(\cdot)$ operates on material properties, $\mathcal{L}(\cdot)$ processes the lighting buffer, and $\mathcal{H}(\cdot)$ fuses these intermediate embeddings to predict the final $\mathbf{L}_o$.

Branch Network Architecture. Each branch ($\mathcal{G}$, $\mathcal{M}$, $\mathcal{L}$) takes as input 512×512 resolution maps: geometry (normal + depth, 4 channels), material (albedo + roughness + metallic, 5 channels), and lighting (shading + mask, 4 channels), respectively. Each branch employs a 4-layer CNN encoder with output channels [64, 128, 256, 512] using 3×3 convolutions followed by GroupNorm(8) and SiLU activation, where layers 2 and 3 use stride-2 downsampling. At the bottleneck ($128\times128\times512$), features are patchified (patch size 2, yielding 64×64 tokens with 512 dimensions) and processed by 2 DiT-style Transformer blocks [PX22]; each block consists of a pre-norm multi-head self-attention layer (dim=512, heads=8) and an MLP (expansion ratio 4, GELU activation) with residual connections. The fusion module $\mathcal{H}$ concatenates the three branch features along the channel dimension (64×64 tokens $\times$ 1536 dim), projects to 512 dimensions, processes through one additional DiT block, reshapes from $(4096,512)$ tokens to $(512,64,64)$ spatial features, and applies a zero-initialized 1×1 convolution to 320 channels before feeding into the ControlNet backbone. The branch network totals 30.8M parameters.

This decomposition aligns with the principle that incorporating domain knowledge as structural priors improves learning efficiency [BHB*18; HMP*16]. By approximating the single-bounce microfacet model as a structural prior, our network design not only reduces the parameter space but also provides an inductive bias that empirically improves training stability.

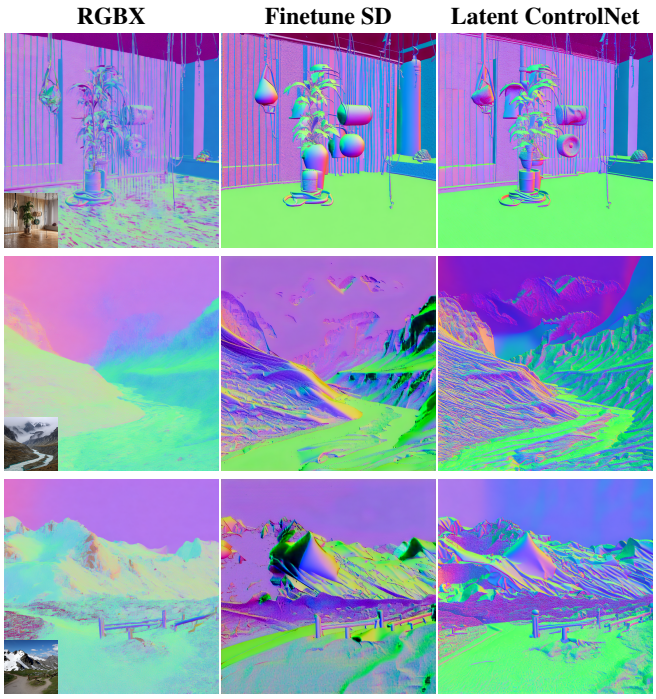

Figure 3: *Text-to-G-buffer Ablation. This figure compares the performance of three text-to-G-buffer generation approaches across three example scenes (columns), with all images depicting normal maps. The first column shows results from linking the RGB↔X network to the full Stable Diffusion pipeline, using the same noise, seed, and generator as our method. The second column presents outcomes from directly training the Stable Diffusion UNet without ControlNet. The third column showcases results from our full method, demonstrating its superior performance compared to the alternatives.*

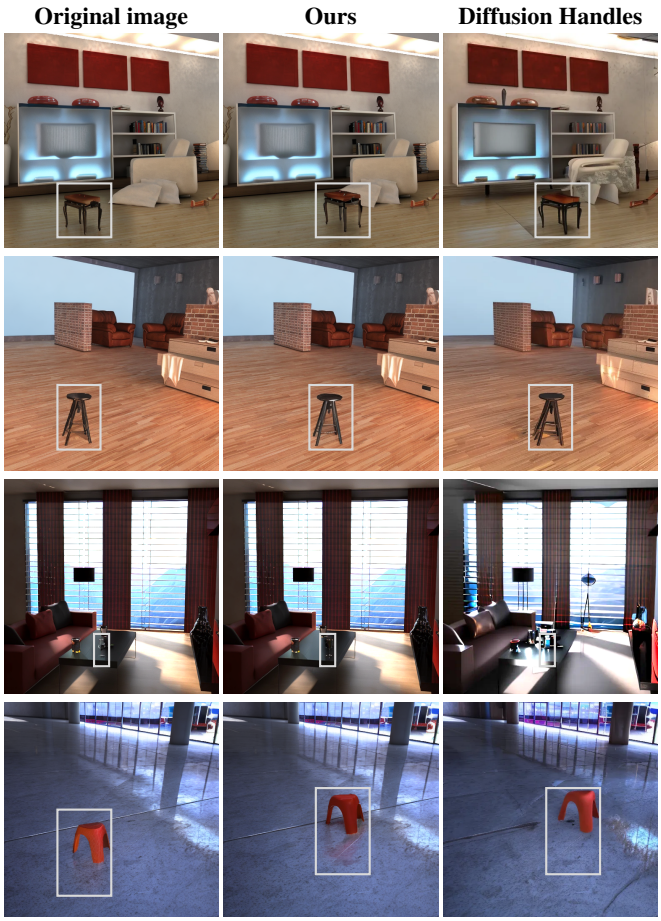

Figure 4: *Comparison with Diffusion Handle[PGG*24]. In this figure, we compare object movement results between our method and Diffusion Handle. Our approach consistently achieves higher-quality outputs with minimal background alterations, whereas Diffusion Handle exhibits more pronounced background changes and underperforms under extreme lighting conditions.*

5. Implementation Details

Editing and Inpainting. When performing inpainting or moving objects, we directly copy the target object into the albedo, normal, roughness, depth, and metallic channels. For the shading map, we fill the edited region with black (i.e., zero) to indicate that these areas need to be recalculated, and simultaneously create a mask channel. In this mask, `mask`=1 denotes unmodified regions, while `mask`=0 indicates edited regions. The same procedure applies to object movement: we update the albedo and other channels, but for the shading channel, the network is guided by the mask to re-estimate lighting where needed.

One-Click Object Insertion. For object insertion, users click on the target object region in the albedo map. We apply edge-based region selection to automatically identify the object boundaries. The selected region is then automatically propagated via copy-paste to all G-buffer channels (albedo, normal, roughness, depth, metallic). For the shading channel, we set the edited region to zero and correspondingly set mask=0 in these areas. Stage-2 then re-synthesizes the shading conditioned on the updated material channels, ensuring more consistent lighting for the inserted object. This workflow makes object insertion a mostly one-click operation.

Dataset. The original diffusion model was trained on an enormous dataset (2.47 billion samples), whereas our indoor-focused data is just 0.005% of that size. To mitigate overfitting, we combine InteriorVerse[ZLH*22b] (50k+ samples with albedo, normal, roughness, depth, and metallic) and Hypersim[RRR*21] (70k+ samples with shading/irradiance but lacking roughness and metallic), yielding over 120k samples in total.

Training Procedure. Our implementation uses a ControlNet (378M parameters) paired with a UNet (865M parameters), with ControlNet representing approximately 44% of UNet's size. We employ the mean squared error (MSE) as the loss function. All channels from the G-Buffer are normalized to the range $[0,1]$. Specifically, the shading and target images are normalized based on the 99% valid pixel range, while the depth channel undergoes

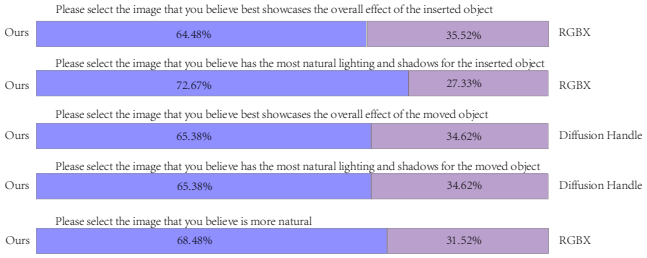

Figure 5: *User study results (156 participants). Each bar illustrates the percentage of participants who preferred either our method or the baseline methods across various evaluation criteria. Participants compared pairs of static images generated by our method and the respective baseline (RGB↔X or Diffusion Handles). Three evaluation questions were utilized for comparisons with RGB↔X, while two questions were employed for comparisons with Diffusion Handles due to technical limitations in rendering buffers with Diffusion Handles.*

logarithmic normalization. The network is trained for a total of 30 epochs. During the first 25 epochs, only the ControlNet is trained, with the main diffusion UNet kept *frozen* to preserve its large-scale generative knowledge. In the final 5 epochs, we *unfreeze* the main UNet. The initial learning rate is set to 5×10^{-6} with a linear decay schedule, and the batch size is 16. The entire training process, executed on four A100 GPUs, requires approximately 150 hours. G-buffer generation model and rendering model trained separately.

Mask-Guided Fine-Tuning. Note that our second-stage network (ControlNet + UNet) accepts albedo, normal, roughness, metallic, shading, and depth channels, plus an optional mask channel. In the first 10 epochs, the mask is set to 1 everywhere (no masking). Later, we gradually introduce regions where `mask`=0, forcing the shading there to be set to zero. This signals the model to re-estimate lighting in those areas, facilitating flexible edits in the final G-buffer. Once the network generates a complete G-buffer, each map (e.g. albedo, normal, roughness) can be freely edited, and for the shading channel specifically, we rely on the mask to indicate which parts need recomputation. Our edited result can consider the shading effect outside the masked region. While the mask indicates regions needing shading recomputation, the network considers global illumination effects extending beyond the masked region. The mask serves as a guide for the network to focus on areas requiring direct shading updates, but the network architecture captures inter-reflections and indirect lighting through learned representations. The white box in figure is for visualization only—not a network input. In Figure 4, the chair's reflection extends beyond the masked region (last image), and the cushions cast reflections outside the mask boundaries (middle two examples).

6. Results and Comparisons

In this section, we present multiple comparison results and ablation studies. Our evaluation includes both objective metrics and user studies, followed by detailed ablations of our architectural choices and comparisons with existing methods on generation and editing tasks.

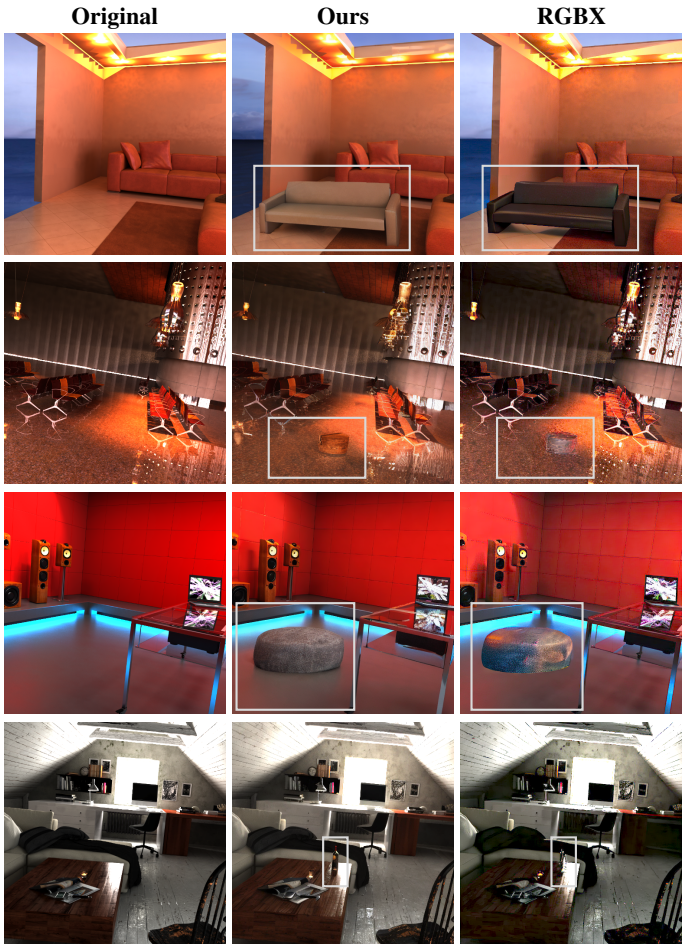

Figure 6: *Comparison with RGB↔X [ZDG*24]. This figure presents an inpainting comparison between our method and an RGB↔X-inpainting variant. The original images and inserted objects are synthetic data from the Hypersim dataset. Our approach demonstrates higher shadow quality and overall image fidelity compared to RGB↔X.*

6.1. Quantitative Evaluation

We evaluated our approach on 5,000 samples using multiple metrics: FID [HRU*17] for image quality, CLIP Text-Image Score for semantic alignment between images and prompts, and CLIP Aesthetic Score for visual quality, which quantifies visual quality by comparing an image's similarity to positive (e.g., "professional photograph") versus negative descriptors (e.g., "poor composition"). All evaluations use the CLIP ViT-B/32 model.

As shown in Table 1, our method achieves comparable FID scores to the base model, indicating that our additional G-buffer processing and rendering steps maintain image quality. Note that "RGB↔X w/ our buffer" refers to using our Stage-1 network to generate G-buffers, which are then fed into the official RGB↔X renderer for final image synthesis. For semantic alignment, we maintain the same high CLIP Text-Image Score as the base

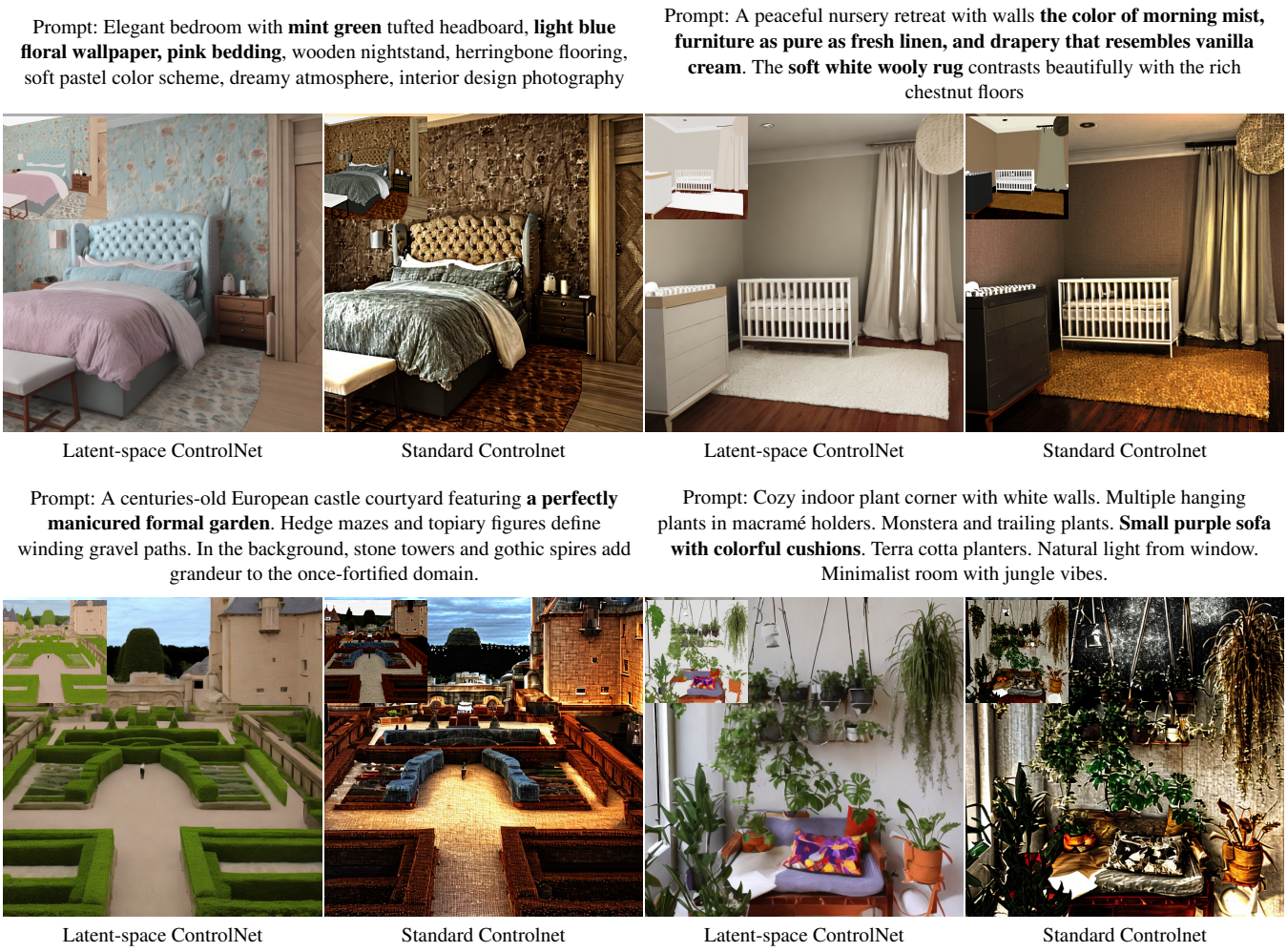

Figure 7: *Comparison of G-buffer generation methods across four scenes. Each rendered image includes its corresponding base color representation inset in the corner, providing insight into the material properties generated by each method. Our Latent-space ControlNet approach (left in each pair) consistently produces superior results with more accurate geometric consistency and finer texture details compared to standard approaches (right in each pair). The comparison spans a variety of interior styles from elegant bedrooms to minimalist Japanese-inspired spaces, demonstrating our method's versatility.*

Table 1: Comparison of different methods across objective quality metrics. ↓ indicates that lower values are better, while ↑ indicates that higher values are better.

Method	**FID↓**	**CLIP-T↑**	**CLIP-A↑**
Stable diffusion	22.96	0.29	0.0042
Ours	22.97	0.29	0.0054
RGB↔X w/ our buffer	25.86	0.28	-0.0001
RGB↔X	49.94	0.18	-0.0021

model (0.29), significantly outperforming RGB↔X. Notably, our approach achieves the highest CLIP Aesthetic Score (0.0054), demonstrating superior visual quality compared to all baselines.

User Study and Analysis. We conducted a user study to evaluate the perceived quality and realism of our method compared to competing approaches. A total of 156 participants were recruited. Each participant completed 30 forced-choice questions, with two images shown side-by-side in each question (our result vs. a baseline result). The participants were instructed to choose which image they felt looked better or more visually plausible. Please check the supplementary material for more details. Each participant received the 30 questions in a randomized order to mitigate potential ordering effects. Within each question, the left-right placement of our output vs. the baseline output was also randomized to avoid positional bias. We emphasized that participants should pay attention to both local details and global consistency. Participants spent approximately 5-10 minutes finishing all 30 questions.

As shown in Figure 5, our method consistently outperformed baselines across all evaluation criteria, with preference rates ranging from 64.48% to 72.67%. These findings demonstrate the supe-

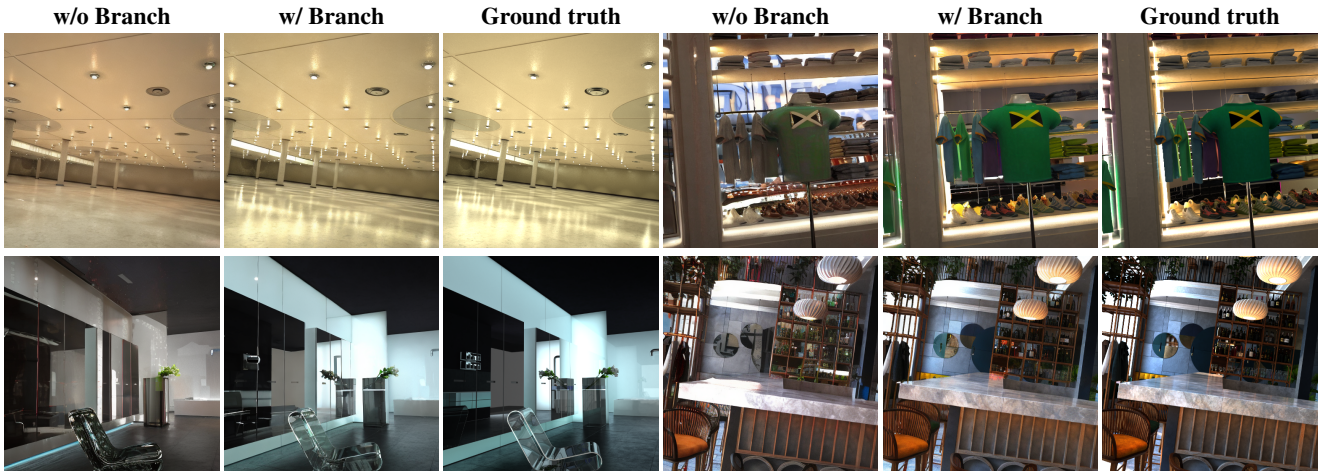

Figure 8: *Ablation of G-buffer to Final Image with or without Branch Networks. This figure illustrates the impact of Branch Networks on G-buffer rendering. Results show that including Branch Networks produces outputs more closely aligned with the ground truth. All G-buffers and ground-truth images are from the Hypersim dataset.*

Figure 9: *Outdoor scene with lake and houses. Our method maintains accurate water surface representation and natural integration with the landscape, while RGB↔X produces significant artifacts at the water boundary and fails to correctly render the lake's surface properties compared to the original image.*

riority of our approach in producing realistic and visually appealing results for both generation and editing tasks.

G-buffer Quality Validation To validate the quality of our generated G-buffers, we evaluate against ground-truth G-buffers from the InteriorVerse test set. Table 2 reports per-channel metrics. Note that for text-to-image generation, there is no unique ground-truth G-buffer for a given prompt; these metrics serve as a reference, while our primary validation comes from downstream task performance (Stage-2 reconstruction, user studies, and editing experiments).

Table 2: Quantitative evaluation of generated G-buffer quality against ground truth on InteriorVerse.

	Albedo	Normal	Depth	Roughness	Metallic
PSNR↑	17.6	20.3	16.3	14.3	14.7
LPIPS↓	0.17	0.16	0.23	0.32	0.31

6.2. Ablation Study

Text-to-G-buffer Ablation. We conduct an ablation study on different strategies for text-to-G-buffer generation, illustrated in Figure 3. To ensure fair comparisons and maintain consistent outputs, all methods use the same random noise, prompt, global seed, and generator. The first row shows an RGB↔X [ZDG*24] network connected to a complete Stable Diffusion 2 pipeline. The second row removes ControlNet and fine-tunes only the Stable Diffusion 2 UNet on our dataset, using the same training duration. The third row employs our proposed Latent ControlNet.

Without shading map, architecture like RGB↔X inpainting network, requiring first rendering result, using a mask to select insertion regions, then blending original and inpainted images. Otherwise, unintended changes occur. In Figure 6, the RGB↔X baseline is blended with a mask, whereas our results are produced without masking. Testing showed without shading map, lighting in most regions differs from original. Shading map serves dual purposes: ensuring stability and enabling lighting editing.

Latent ControlNet Ablation. To evaluate the efficacy of our

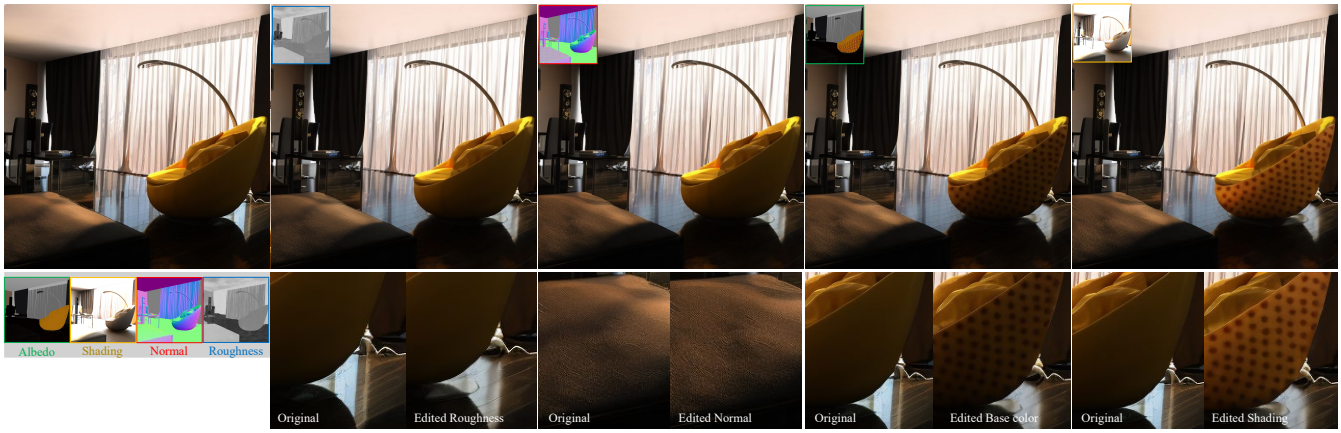

Figure 10: *Starting from a base indoor scene and original G-buffer (left), we demonstrate the effects of modifying different G-buffer channels: increasing the floor's roughness to make it more rough, modifying the normal map of the stool in the lower left corner to add multiple small bumps/protrusions, altering the base color/albedo by adding a special pattern to the sofa in the lower right corner, and adjusting shading values to change the lighting effects. This editability is enabled by our method's ability to generate consistent and PBR-inspired G-buffers with clean channel separation.*

proposed latent-space control mechanism, we conducted an ablation study comparing our method with the original ControlNet approach across diverse scenes. As illustrated in Figure 7, our Latent-space ControlNet implementation (left in each pair) consistently produces higher quality images that more accurately match the input prompts compared to standard approaches (right in each pair). This poor performance of standard ControlNet has also been documented in other related papers [LCY*24]. Figure 6 of that paper demonstrates that standard ControlNet leads to albedo color mismatches. The comparative analysis reveals a significant limitation in the original ControlNet method, where mismatches between base color representations and the intended prompt specifications frequently occur, as highlighted in the bolded sections of our caption. These discrepancies are clearly visible in the base color representations inset in each rendered image, demonstrating how standard approaches struggle with prompt-material coherence. Our Latent-space ControlNet successfully addresses this limitation, producing results that faithfully adhere to prompt specifications across a variety of environments, including both interior and exterior scenes.

G-buffer to Final Image With/Without Branch Networks. Figure 8 compares our branch-based rendering (employing sub-networks $\mathcal{G}$, $\mathcal{M}$, $\mathcal{L}$, and a merge module $\mathcal{H}$) against a single ControlNet trained end-to-end without such factorization. For clarity, we do not use the entire pipeline; instead, we take G-buffers directly from the dataset and feed them into the second-stage network to render images, allowing a direct comparison with ground truth.

Our PBR-inspired branch approach captures lighting, geometry, and material properties more consistently. In contrast, direct ControlNet training exhibits mild color mismatches (e.g., background color artifacts) and struggles with complex lighting effects (e.g., ground reflections). Transparent or highly reflective objects (glass seats, mirrors, metallic surfaces) are also rendered more accurately by our branched model. The single ControlNet baseline frequently produces metallic reflections in glass objects or overly diffuse reflections on metallic surfaces, undermining realism. Table 3 illustrates the performance improvements achieved by incorporating branch networks into our model. Specifically, the model with branch networks exhibits a significant reduction in Mean Squared Error (MSE) and Learned Perceptual Image Patch Similarity (LPIPS) scores, alongside enhancements in Structural Similarity Index Measure (SSIM) and Peak Signal-to-Noise Ratio (PSNR) metrics, compared to the model without branch networks.

Table 3: Performance Comparing Models With and Without Branch Networks. ↓ indicates that lower values are better, while ↑ indicates that higher values are better.

Metric	w/o Branch Networks	w/ Branch Networks
MSE ↓	0.0288	0.0068
LPIPS ↓	0.2686	0.0973
SSIM ↑	0.6350	0.8072
PSNR ↑	15.9983	21.9883

6.3. Comparison with Related Work

Insertion comparison with RGB↔X Figure 6 compares our method and RGB↔X [ZDG*24] on insertion tasks. To ensure fairness, we use *existing* G-buffers from the Hypersim dataset. The left column shows the original image, the middle column shows our result, and the right column is RGB↔X's output.

Our method produces more natural shadows and lighting for inserted objects. By contrast, RGB↔X often exhibits unnatural artifacts or inaccurate color for inserted objects, especially for reflective or refractive surfaces like glass (e.g., the wine bottle in the final scene). In the second example, where a wooden stump is inserted, our method preserves refraction effects, yielding a coherent scene.

RGB↔X fails to capture these details, leading to visually inconsistent results. In the third example, RGB↔X generates odd shadows, whereas ours maintains shape details and realistic shadow casting.

Comparison whole Pipeline with RGB↔X Figure 9 presents a further comparison between our approach and RGB↔X, highlighting fundamental differences in processing methodology. In this experiment, we use Stable Diffusion outputs processed through VAE networks as inputs for RGB↔X, while our method directly operates on latent representations. The results demonstrate that RGB↔X produces outputs with noticeable artifacts, particularly in outdoor scenes, where the generated images deviate significantly from the original inputs. This may occur because RGB↔X requires an additional decode–encode cycle, whereas our method operates entirely in latent space. Consequently, edited results from RGB↔X exhibit these artifacts too. In contrast, our latent-based approach produces more coherent and visually consistent results, validating the effectiveness of our direct latent manipulation strategy.

Comparison with IntrinsicEdit. Figure 11 presents a qualitative comparison with IntrinsicEdit [LDH*25] on editing tasks. IntrinsicEdit is a training-free optimization framework built on top of RGB↔X, which offers flexibility but may also inherit limitations from its base model. In the first two indoor examples, IntrinsicEdit exhibits color shifts from the original images: specifically, the shadow position under the table changes unexpectedly in the first example, and reflections of the stool appear on the floor (not present in the original) in the second. In the third example (sofa insertion), IntrinsicEdit produces bright regions beneath the sofa, similar to the effect observed with RGB↔X (see Figure 6), suggesting this issue may stem from the underlying base model. In the fourth example, the chair's reflection on the marble surface differs from the original.

Comparison Moving Object with Diffusion Handles. Figure 4 compares our method to a diffusion-based editing approach [PGG*24] that specializes in moving objects within an image. We generated multiple outputs (over five) for the competing method and chose its best result for display; even so, it often distorts the background or alters object geometry. For instance, in the first image, the sofa becomes deformed when the chair is moved. In the second image, a background scarf becomes partially transparent, and in the third example, the glass object fails to maintain realistic lighting. The fourth image consistently shows unnatural floor lighting. In contrast, our approach preserves background details, refraction, shadow consistency, and object geometry across all examples, resulting in more reliable and practical edited outputs.

6.4. Additional Editing Results

Material and Texture Editing. As shown in Figure 11 (Rows 1-2), our method demonstrates multiple material and texture editing capabilities. Starting from an original indoor scene, we expand carpet texture to cover the entire floor surface and transform a wooden cabinet to a metallic finish. These edits are cumulative—each edit includes all previous modifications. These edits highlight our approach's effectiveness in texture synthesis and material property modification while preserving unmodified scene elements.

G-buffer Channel Manipulation Figure 10 illustrates our method's control over individual G-buffer channels. From a base indoor scene, we modify different channels: increasing floor roughness, adding bumps to the stool's normal map, applying a special pattern to the sofa's base color, and adjusting shading values. This editability is enabled by our approach's generation of consistent and PBR-inspired G-buffers with clean channel separation. Figure 1 shows object insertion alongside various per-channel modifications and their combined effects.

Figure 11: *Qualitative comparison with IntrinsicEdit [LDH*25] on editing tasks. In the first two indoor examples, IntrinsicEdit exhibits noticeable color shifts from the original images. Specifically, in the first example, the shadow position (under the table) changes unexpectedly; and reflections of the stool appear on the floor (not present in the original) in the second. In the third example, bright regions appear beneath the sofa, similar to the effect observed with RGB↔X (see Figure 6). In the fourth example, the chair's reflection on the marble surface differs from the original.*

7. Discussion and Conclusion

Limitations and Future Work. While our method demonstrates strong generalization from indoor training data to outdoor scenes, it may encounter challenges with extremely complex outdoor environments that differ significantly from typical scenes (e.g., underwater environments, aerial landscapes, or abstract artistic composi-

tions). Our single-bounce approximation and training data bias can result in soft shadows and blurry specular highlights in some cases, as the model does not explicitly handle multi-bounce light transport or high-frequency specular reflections. While outdoor scene quality is improved compared to RGB↔X (see Figure 9 and supplementary materials), results remain weaker than for indoor scenes due to limited outdoor training data. Additionally, the two-stage pipeline increases inference time compared to direct text-to-image generation. Future work could explore more diverse training datasets and investigate single-stage architectures that maintain our editing capabilities while improving efficiency.

Conclusion. We presented a controllable diffusion pipeline for G-buffer generation and rendering, bridging generative diffusion with fine-grained scene control, addressing the fundamental limitation of current text-to-image systems. Our Latent ControlNet operates directly in latent space, avoiding information loss. The PBR-inspired branch network, motivated by the rendering equation, achieves 76% reduction in MSE error while enabling intuitive per-channel editing. Our pipeline design and progressive training strategy successfully learn from limited data while preserving the generative capabilities of models. Extensive evaluations, including user studies with 156 participants, confirm our method's superiority in both generation quality and editing capabilities. This integration enables controllable, PBR-inspired image synthesis with channel-wise edits to geometry, materials, and lighting.

References

[BBM*01] BUEHLER, CHRIS, BOSSE, MICHAEL, MCMILLAN, LEONARD, et al. "Unstructured lumigraph rendering". *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '01. New York, NY, USA: Association for Computing Machinery, 2001, 425–432. ISBN: 158113374X. DOI: 10.1145/383259.383309. URL: https://doi.org/10.1145/383259.383309 2.

[BBS14] BELL, SEAN, BALA, KAVITA, and SNAVELY, NOAH. "Intrinsic images in the wild". *ACM Trans. Graph.* 33.4 (July 2014). ISSN: 0730-0301. DOI: 10.1145/2601097.2601206 2.

[BHB*18] BATTAGLIA, PETER, HAMRICK, JESSICA BLAKE CHANDLER, BAPST, VICTOR, et al. "Relational inductive biases, deep learning, and graph networks". *arXiv* (2018). URL: https://arxiv.org/pdf/1806.01261.pdf 4.

[BMV*22] BARRON, JONATHAN T., MILDENHALL, BEN, VERBIN, DOR, et al. "Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields". *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, 5460–5469. DOI: 10.1109/CVPR52688.2022.00539 2.

[BT78] BARROW, HARRY G. and TENENBAUM, JAY M. "RECOVERING INTRINSIC SCENE CHARACTERISTICS FROM IMAGES". 1978. URL: https://api.semanticscholar.org/CorpusID:14892007 2.

[Bur12] BURLEY, BRENT. "Physically-Based Shading at Disney". 2012. URL: https://api.semanticscholar.org/CorpusID:7260137 2.

[CHL*23] CHEN, XI, HUANG, LIANGHUA, LIU, YU, et al. "Anydoor: Zero-shot object-level image customization". *arXiv preprint arXiv:2307.09481* (2023) 3.

[Cov99] COVER, THOMAS M. *Elements of information theory*. John Wiley & Sons, 1999 3.

[CT82] COOK, R. L. and TORRANCE, K. E. "A Reflectance Model for Computer Graphics". *ACM Trans. Graph.* 1.1 (Jan. 1982), 7–24. ISSN: 0730-0301. DOI: 10.1145/357290.357293. URL: https://doi.org/10.1145/357290.357293 4.

[CVSC22] COUAIRON, GUILLAUME, VERBEEK, JAKOB, SCHWENK, HOLGER, and CORD, MATTHIEU. *DiffEdit: Diffusion-based semantic image editing with mask guidance*. 2022. arXiv: 2210.11427 [cs.CV]. URL: https://arxiv.org/abs/2210.11427 3.

[DN21] DHARIWAL, PRAFULLA and NICHOL, ALEX. "Diffusion models beat GANs on image synthesis". *Proceedings of the 35th International Conference on Neural Information Processing Systems*. NIPS '21. Red Hook, NY, USA: Curran Associates Inc., 2021. ISBN: 9781713845393 2.

[GRP22] GRIFFITHS, DAVID, RITSCHEL, TOBIAS, and PHILIP, JULIEN. "OutCast: Single Image Relighting with Cast Shadows". *Computer Graphics Forum* 43 (2022) 2.

[HMP*16] HIGGINS, IRINA, MATTHEY, LOÏC, PAL, ARKA, et al. "beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework". *International Conference on Learning Representations*. 2016. URL: https://api.semanticscholar.org/CorpusID:46798026 4.

[HMT*22] HERTZ, AMIR, MOKADY, RON, TENENBAUM, JAY, et al. "Prompt-to-prompt image editing with cross attention control". (2022) 3.

[HPP*18] HEDMAN, PETER, PHILIP, JULIEN, PRICE, TRUE, et al. "Deep blending for free-viewpoint image-based rendering". *ACM Trans. Graph.* 37.6 (Dec. 2018). ISSN: 0730-0301. DOI: 10.1145/3272127.3275084. URL: https://doi.org/10.1145/3272127.3275084 2.

[HRU*17] HEUSEL, MARTIN, RAMSAUER, HUBERT, UNTERTHINER, THOMAS, et al. "GANs trained by a two time-scale update rule converge to a local nash equilibrium". *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, 6629–6640. ISBN: 9781510860964 6.

[Kaj86] KAJIYA, JAMES T. "The rendering equation". *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '86. New York, NY, USA: Association for Computing Machinery, 1986, 143–150. ISBN: 0897911962. DOI: 10.1145/15922.15902. URL: https://doi.org/10.1145/15922.15902 4.

[KPS*24] KOCSIS, PETER, PHILIP, JULIEN, SUNKAVALLI, KALYAN, et al. "LightIt: Illumination Modeling and Control for Diffusion Models". *CVPR*. 2024 3.

[KSN24] KOCSIS, PETER, SITZMANN, VINCENT, and NIESSNER, MATTHIAS. "Intrinsic Image Diffusion for Indoor Single-view Material Estimation". 2024 3.

[LBB*25] LABS, BLACK FOREST, BATIFOL, STEPHEN, BLATTMANN, ANDREAS, et al. *FLUX.1 Kontext: Flow Matching for In-Context Image Generation and Editing in Latent Space*. 2025. arXiv: 2506.15742 [cs.GR]. URL: https://arxiv.org/abs/2506.15742 3.

[LCY*24] LUO, JUNDAN, CEYLAN, DUYGU, YOON, JAE SHIN, et al. "IntrinsicDiffusion: Joint Intrinsic Layers from Latent Diffusion Models". *ACM SIGGRAPH 2024 Conference Papers*. SIGGRAPH '24. Denver, CO, USA: Association for Computing Machinery, 2024. ISBN: 9798400705250. DOI: 10.1145/3641519.3657472. URL: https://doi.org/10.1145/3641519.3657472 3, 9.

[LDH*25] LYU, LINJIE, DESCHAINTRE, VALENTIN, HOLD-GEOFFROY, YANNICK, et al. "IntrinsicEdit: Precise generative image manipulation in intrinsic space". *ACM Trans. Graph.* 44.4 (July 2025). ISSN: 0730-0301. DOI: 10.1145/3731173. URL: https://doi.org/10.1145/3731173 3, 10.

[LGL*25] LIANG, RUOFAN, GOJCIC, ZAN, LING, HUAN, et al. "DiffusionRenderer: Neural Inverse and Forward Rendering with Video Diffusion Models". *arXiv preprint arXiv:2501.18590* (2025) 2, 3.

[LLW*23] LI, YUHENG, LIU, HAOTIAN, WU, QINGYANG, et al. " GLIGEN: Open-Set Grounded Text-to-Image Generation ". *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, June 2023, 22511–22521. DOI: 10.1109/CVPR52729.2023.02156. URL: https://doi.ieeecomputersociety.org/10.1109/CVPR52729.2023.02156 3.

[LSS*19] LOMBARDI, STEPHEN, SIMON, TOMAS, SARAGIH, JASON, et al. "Neural volumes: learning dynamic renderable volumes from images". *ACM Trans. Graph.* 38.4 (July 2019). ISSN: 0730-0301. DOI: 10.1145/3306346.3323020. URL: https://doi.org/10.1145/3306346.3323020 2.

[LZF*24] LI, SHANGLIN, ZENG, BOHAN, FENG, YUTANG, et al. "ZONE: Zero-Shot Instruction-Guided Local Editing". *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2024, 6254–6263. DOI: 10.1109/CVPR52733.2024.00598 3.

[MST*21] MILDENHALL, BEN, SRINIVASAN, PRATUL P., TANCIK, MATTHEW, et al. "NeRF: representing scenes as neural radiance fields for view synthesis". *Commun. ACM* 65.1 (Dec. 2021), 99–106. ISSN: 0001-0782. DOI: 10.1145/3503250. URL: https://doi.org/10.1145/3503250 2.

[MWX*24] MOU, CHONG, WANG, XINTAO, XIE, LIANGBIN, et al. "T2I-Adapter: learning adapters to dig out more controllable ability for text-to-image diffusion models". *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence and Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence and Fourteenth Symposium on Educational Advances in Artificial Intelligence*. AAAI'24/IAAI'24/EAAI'24. AAAI Press, 2024. ISBN: 978-1-57735-887-9. DOI: 10.1609/aaai.v38i5.28226. URL: https://doi.org/10.1609/aaai.v38i5.28226 2, 3.

[NAM*17] NALBACH, O., ARABADZHIYSKA, E., MEHTA, D., et al. "Deep Shading: Convolutional Neural Networks for Screen Space Shading". *Computer Graphics Forum* 36.4 (2017), 65–78. DOI: https://doi.org/10.1111/cgf.13225. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.13225. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13225 2.

[NDR*22] NICHOL, ALEXANDER QUINN, DHARIWAL, PRAFULLA, RAMESH, ADITYA, et al. "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models". *Proceedings of the 39th International Conference on Machine Learning*. Ed. by CHAUDHURI, KAMALIKA, JEGELKA, STEFANIE, SONG, LE, et al. Vol. 162. Proceedings of Machine Learning Research. PMLR, 17–23 Jul 2022, 16784–16804. URL: https://proceedings.mlr.press/v162/nichol22a.html 2.

[PEL*21] PANDEY, ROHIT, ESCOLANO, SERGIO ORTS, LEGENDRE, CHLOE, et al. "Total relighting: learning to relight portraits for background replacement". *ACM Trans. Graph.* 40.4 (July 2021). ISSN: 0730-0301. DOI: 10.1145/3450626.3459872. URL: https://doi.org/10.1145/3450626.3459872 2.

[PGG*24] PANDEY, KARRAN, GUERRERO, PAUL, GADELHA, MATHEUS, et al. "Diffusion Handles Enabling 3D Edits for Diffusion Models by Lifting Activations to 3D". *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, 7695–7704 3, 5, 10.

[PSB*21] PARK, KEUNHONG, SINHA, UTKARSH, BARRON, JONATHAN T., et al. "Nerfies: Deformable Neural Radiance Fields". *ICCV* (2021) 2.

[PX22] PEEBLES, WILLIAM and XIE, SAINING. "Scalable Diffusion Models with Transformers". *arXiv preprint arXiv:2212.09748* (2022) 4.

[PZ17] PENNER, ERIC and ZHANG, LI. "Soft 3D reconstruction for view synthesis". *ACM Trans. Graph.* 36.6 (Nov. 2017). ISSN: 0730-0301. DOI: 10.1145/3130800.3130855. URL: https://doi.org/10.1145/3130800.3130855 2.

[RAY*16] REED, SCOTT, AKATA, ZEYNEP, YAN, XINCHEN, et al. "Generative adversarial text to image synthesis". *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*. ICML'16. New York, NY, USA: JMLR.org, 2016, 1060–1069 2.

[RBL*22] ROMBACH, ROBIN, BLATTMANN, ANDREAS, LORENZ, DOMINIK, et al. " High-Resolution Image Synthesis with Latent Diffusion Models ". *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, June 2022, 10674–10685. DOI: 10.1109/CVPR52688.2022.01042. URL: https://doi.ieeecomputersociety.org/10.1109/CVPR52688.2022.01042 1, 2.

[RES*22] RUDNEV, VIKTOR, ELGHARIB, MOHAMED, SMITH, WILLIAM, et al. "NeRF for Outdoor Scene Relighting". *European Conference on Computer Vision (ECCV)*. 2022 3.

[RLJ*23] RUIZ, NATANIEL, LI, YUANZHEN, JAMPANI, VARUN, et al. " DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation ". *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, June 2023, 22500–22510. DOI: 10.1109/CVPR52729.2023.02155. URL: https://doi.ieeecomputersociety.org/10.1109/CVPR52729.2023.02155 3.

[RPG*21] RAMESH, ADITYA, PAVLOV, MIKHAIL, GOH, GABRIEL, et al. "Zero-Shot Text-to-Image Generation". *Proceedings of the 38th International Conference on Machine Learning*. Ed. by MEILA, MARINA and ZHANG, TONG. Vol. 139. Proceedings of Machine Learning Research. PMLR, 18–24 Jul 2021, 8821–8831. URL: https://proceedings.mlr.press/v139/ramesh21a.html 1, 2.

[RRR*21] ROBERTS, MIKE, RAMAPURAM, JASON, RANJAN, ANURAG, et al. "Hypersim: A Photorealistic Synthetic Dataset for Holistic Indoor Scene Understanding". *International Conference on Computer Vision (ICCV) 2021*. 2021 5.

[SCS*22] SAHARIA, CHITWAN, CHAN, WILLIAM, SAXENA, SAURABH, et al. "Photorealistic text-to-image diffusion models with deep language understanding". *Proceedings of the 36th International Conference on Neural Information Processing Systems*. NIPS '22. New Orleans, LA, USA: Curran Associates Inc., 2022. ISBN: 9781713871088 1, 2.

[ST90] SAITO, TAKAFUMI and TAKAHASHI, TOKIICHIRO. "Comprehensible rendering of 3-D shapes". *Proceedings of the 17th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '90. Dallas, TX, USA: Association for Computing Machinery, 1990, 197–206. ISBN: 0897913442. DOI: 10.1145/97879.97901. URL: https://doi.org/10.1145/97879.97901 2.

[TTG*20] TRETSCHK, EDGAR, TEWARI, AYUSH, GOLYANIK, VLADISLAV, et al. "PatchNets: Patch-Based Generalizable Deep Implicit 3D Shape Representations". *Computer Vision – ECCV 2020*. Ed. by VEDALDI, ANDREA, BISCHOF, HORST, BROX, THOMAS, and FRAHM, JAN-MICHAEL. Cham: Springer International Publishing, 2020, 293–309. ISBN: 978-3-030-58517-4 2.

[WLZ*25] WU, CHENFEI, LI, JIAHAO, ZHOU, JINGREN, et al. *Qwen-Image Technical Report*. 2025. arXiv: 2508.02324 [cs.CV]. URL: https://arxiv.org/abs/2508.02324 3.

[WSG*23] WANG, ZIAN, SHEN, TIANCHANG, GAO, JUN, et al. "Neural Fields meet Explicit Geometric Representations for Inverse Rendering of Urban Scenes". *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023 3.

[XGZM24] XUE, BOWEN, GUARNERA, CLAUDIO, ZHAO, SHUANG, and MONTAZERI, ZAHRA. "ReflectanceFusion: Diffusion-based text to SVBRDF Generation". *Eurographics Symposium on Rendering*. Eurographics Association. 2024 3.

[XZH*18] XU, TAO, ZHANG, PENGCHUAN, HUANG, QIUYUAN, et al. "AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks". *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, 1316–1324. DOI: 10.1109/CVPR.2018.00143 2.

[XZJM24] XUE, BOWEN, ZHAO, SHUANG, JENSEN, HENRIK WANN, and MONTAZERI, ZAHRA. "A Hierarchical Architecture for Neural Materials". *Computer Graphics Forum* 43.6 (2024), e15116. DOI: https://doi.org/10.1111/cgf.15116. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.15116. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.15116 2.

[YME*20] YU, YE, MEKA, ABHIMETRA, ELGHARIB, MOHAMED, et al. "Self-supervised Outdoor Scene Relighting". *European Conference on Computer Vision (ECCV)*. 2020 2.

[ZDG*24] ZENG, ZHENG, DESCHAINTRE, VALENTIN, GEORGIEV, ILIYAN, et al. "RGB↔X: Image decomposition and synthesis using material- and lighting-aware diffusion models". *ACM SIGGRAPH 2024 Conference Papers*. SIGGRAPH '24. Denver, CO, USA: Association for Computing Machinery, 2024. ISBN: 9798400705250. DOI: 10.1145/3641519.3657445. URL: https://doi.org/10.1145/3641519.3657445 2, 6, 8, 9.

[ZDP*24] ZENG, CHONG, DONG, YUE, PEERS, PIETER, et al. "DiLightNet: Fine-grained Lighting Control for Diffusion-based Image Generation". *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers '24*. SIGGRAPH '24. ACM, July 2024, 1–12. DOI: 10.1145/3641519.3657396. URL: http://dx.doi.org/10.1145/3641519.3657396 3.

[ZLH*22a] ZHU, JINGSEN, LUAN, FUJUN, HUO, YUCHI, et al. "Learning-based Inverse Rendering of Complex Indoor Scenes with Differentiable Monte Carlo Raytracing". *SIGGRAPH Asia 2022 Conference Papers*. SA '22. Daegu, Republic of Korea: Association for Computing Machinery, 2022. ISBN: 9781450394703. DOI: 10.1145/3550469.3555407. URL: https://doi.org/10.1145/3550469.3555407 2.

[ZLH*22b] ZHU, JINGSEN, LUAN, FUJUN, HUO, YUCHI, et al. "Learning-based Inverse Rendering of Complex Indoor Scenes with Differentiable Monte Carlo Raytracing". *SIGGRAPH Asia 2022 Conference Papers*. SA '22. Daegu, Republic of Korea: Association for Computing Machinery, 2022. ISBN: 9781450394703. DOI: 10.1145/3550469.3555407. URL: https://doi.org/10.1145/3550469.3555407 5.

[ZRA23] ZHANG, LVMIN, RAO, ANYI, and AGRAWALA, MANEESH. *Adding Conditional Control to Text-to-Image Diffusion Models*. 2023 2, 3.

[ZXL*17] ZHANG, HAN, XU, TAO, LI, HONGSHENG, et al. "StackGAN: Text to Photo-Realistic Image Synthesis with Stacked Generative Adversarial Networks". *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, 5908–5916. DOI: 10.1109/ICCV.2017.629 2.