Instruct 3D-to-3D: Text Instruction Guided 3D-to-3D conversion

Hiromichi Kamata* Yuiko Sakuma Akio Hayakawa Masato Ishii Takuya Narihira
Sony Group Corporation
Tokyo, Japan
{Hiromichi.Kamata, Yuiko.Sakuma, Akio.Hayakawa, Masato.A.Ishii, Takuya.Narihira}@sony.com

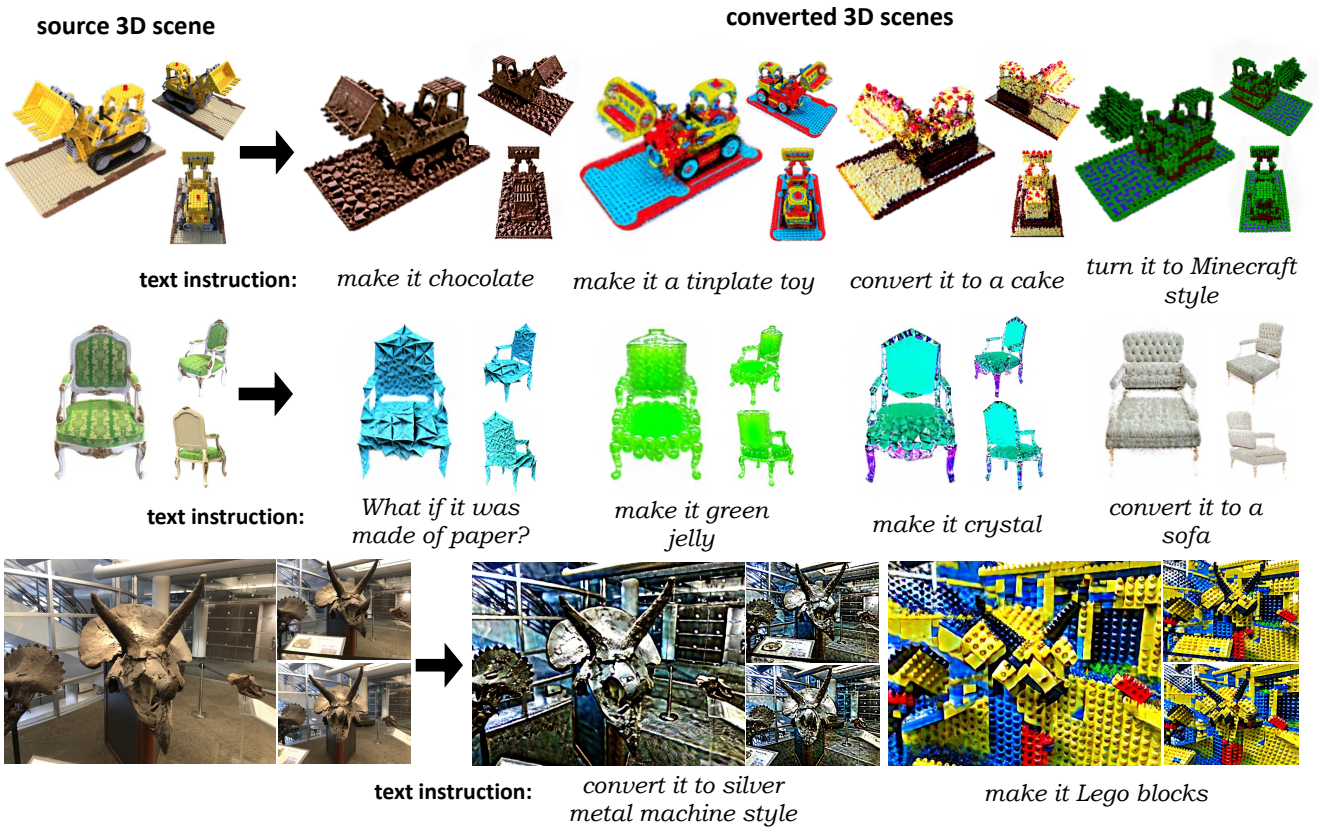

Figure 1: Instruct 3D-to-3D Results. Our Instruct 3D-to-3D successfully converts input 3D scenes according to the text instructions.

Abstract

We propose a high-quality 3D-to-3D conversion method, Instruct 3D-to-3D. Our method is designed for a novel task, which is to convert a given 3D scene to another scene according to text instructions. Instruct 3D-to-3D applies pre-trained Image-to-Image diffusion models for 3D-to-3D conversion. This enables the likelihood maximization of each viewpoint image and high-quality 3D generation. In addition, our proposed method explicitly inputs the source 3D scene as a condition, which enhances 3D consistency and controllability of how much of the source 3D scene structure is reflected. We also propose dynamic scaling, which allows the intensity of the geometry transformation to be adjusted. We performed quantitative and qualitative evaluations and showed that our proposed method achieves higher quality 3D-to-3D conversions than baseline methods.

*Corresponding author: Hiromichi.Kamata@sony.com

1. Introduction

In recent years, generative modelings based on diffusion models have been rapidly developed and applied in a wide range of domains, including image [28, 30, 29], music [12, 24], video [31, 10], and 3D [26, 16]. Especially for Text-to-Image (T2I) tasks, diffusion models have received much attention for their ability to generate high-quality images that align with input texts. Such amazing T2I models are often realized by large-scale training with a huge amount of image-text pair dataset [28, 30, 29].

The pretrained T2I models can be also utilized to solve various tasks other than T2I. In particular, the application of diffusion models to the field of 3D has been studied extensively [26, 16, 19, 37]. They used the pretrained T2I models to optimize NeRF [21] so that the content of rendered images from NeRF always matches input texts. Through this optimization, they successfully generate NeRF representation of the 3D scene described by the input texts.

Nowadays, it has become much easier to obtain NeRF representations of real-world scenes [21, 2]. If these 3D scenes can be edited with text instructions, it should also make it substantially easier to create high-quality 3D content. Prior works [35, 36] have shown that it is possible to stylize a 3D scene based on a given text that describes the target scene to be obtained after the editing. However, in their methods, we cannot directly instruct what scene to be changed, which fairly reduces the controllability of the editing. In addition, we cannot also accurately control how strongly the structure of the original scene would be preserved after the editing.

In this paper, we tackle a novel and challenging problem, which is to convert a given 3D scene to another scene according to text instructions. To solve this problem, we propose **Instruct 3D-to-3D**, which performs 3D-to-3D conversions using pretrained diffusion models.

Our Instruct 3D-to-3D achieves better 3D consistency, quality, and controllability simultaneously compared to baseline methods. For better 3D consistency, we directly use the source 3D scene as a condition to keep its semantic and structural information. Furthermore, to achieve better quality, we apply a pretrained Image-to-Image (I2I) diffusion model that allows us to maximize the likelihood of each viewpoint image. In addition, we propose dynamic scaling to enable the users to control the strength of the geometry transformation. We use voxel grid-based implicit 3D representation [33, 7] as a 3D model. Our dynamic scaling gradually decreases and increases the 3D resolution of the voxel grid to achieve controllable and smooth 3D geometry conversions.

The main contributions of this study are summarized as follows.

- We propose Instruct 3D-to-3D, a method for transforming a 3D scene based on text instructions. Our Instruct 3D-to-3D realizes the high quality and 3D-consistent conversion using a pretrained I2I model conditioned by the source 3D scene.
- We propose dynamic scaling. This allows manipulation of the strength of the geometry transformation and smooth 3D-to-3D conversion.
- We conducted qualitative and quantitative evaluations, showing that our proposed method can perform 3D-to-3D conversion with higher quality than baseline methods.

2. Related Works

2.1. Diffusion Models

Diffusion models [32, 6] are generative models inspired by non-equilibrium thermodynamics. It can generate high-fidelity images by gradually denoising data starting from pure random noise. The diffusion models comprise two processes along with timesteps: a forward process where a small amount of noise is added to the data at each timestep, and a reverse process where the data are slightly denoised at each timestep. Specifically, in the forward process, the data at timestep t can be obtained as follows:

$$\begin{aligned} \boldsymbol{x}_t &= \sqrt{\alpha_t}\boldsymbol{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} \\ &= \cdots = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon} \end{aligned} \tag{1}$$

Here, $\{\alpha_i\}_{i=1}^T$ are hyper-parameters that satisfy $0 < \alpha_i < 1$ and $\bar{\alpha}_t = \Pi_{i=1}^t \alpha_i$, $0 < \bar{\alpha}_T < \bar{\alpha}_{T-1} < \cdots < \bar{\alpha}_T < 1$. $\boldsymbol{\epsilon}$ is a random noise sampled as $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$.

In the reverse process, The noise added at each timestep needs to be predicted from noisy data using a neural network to denoise the data. We may optionally use condition information for this prediction, and here we use an input text y. The predicted noise is denoted as $\epsilon_\phi(\boldsymbol{x}_t; y, t)$, where ϕ is the parameters of the neural network. The neural network is trained to minimize the following loss function.

$$\mathcal{L} = \mathbb{E}_{t,\boldsymbol{\epsilon}}\left[w(t)\|\epsilon_\phi(\boldsymbol{x}_t; y, t) - \boldsymbol{\epsilon}\|^2\right] \tag{2}$$

where $w(t)$ is a coefficient calculated with the scheduling parameter.

In classifier free guidance [11], the strength of the condition y to the generated images can be controlled by changing the noise prediction as the following equation:

$$\begin{aligned} \tilde{\epsilon}_\phi(\boldsymbol{x}_t; y, t) = \epsilon_\phi(\boldsymbol{x}_t; \varnothing, t) \\ + s \cdot (\epsilon_\phi(\boldsymbol{x}_t; y, t) - \epsilon_\phi(\boldsymbol{x}_t; \varnothing, t)) \end{aligned} \tag{3}$$

where $\varnothing$ is a fixed null value. The value of s corresponds to the strength of y. By using larger s, we can generate images that are more faithful to the condition y.

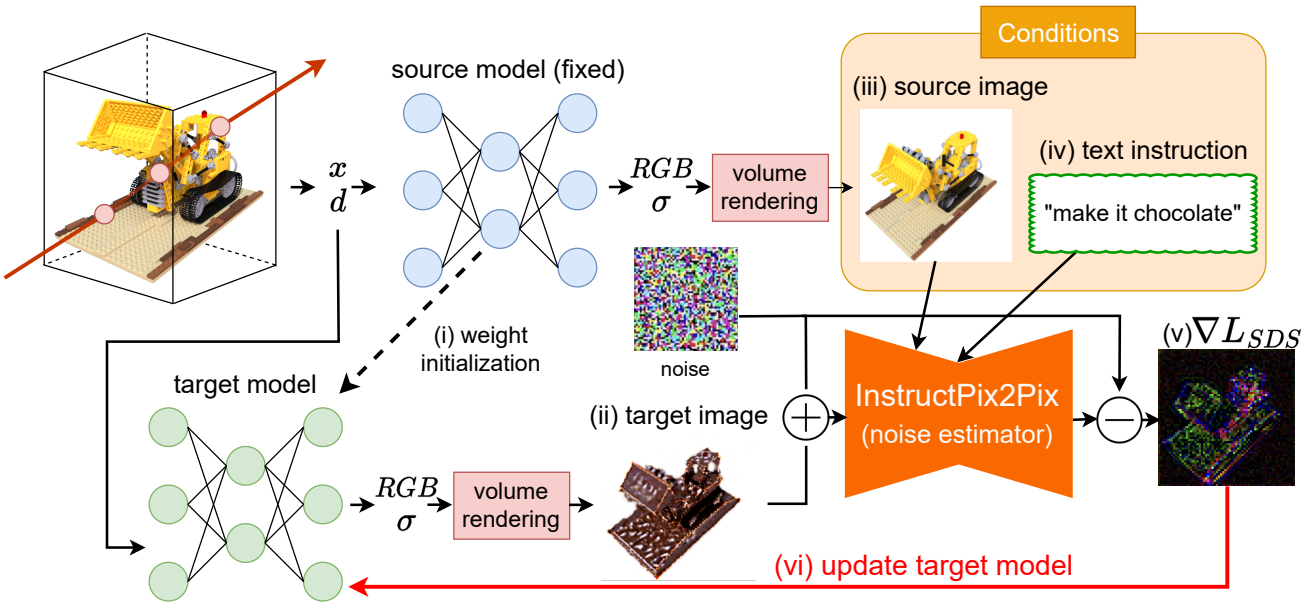

Figure 2: Overview of Instruct 3D-to-3D. First, the target model is initialized with the source model (i). Next, the target image is rendered from a random camera viewpoint (ii) and then the noise is added to input into InstructPix2Pix. The source image is rendered from the same viewpoint (iii) and input to InstructPix2Pix as conditions along with the text instruction (iv). $\nabla \mathcal{L}_{\text{SDS}}$ is calculated using them (v) and the target model is updated with it. By performing this procedure from various camera viewpoints, we can convert the target model along with the text instruction.

2.2. Text-guided Image-to-Image using Diffusion Models

Diffusion models have also been extensively studied for text-guided Image-to-Image tasks. These methods can be divided into two categories based on what the input text represents. The first category assumes that the text describes the caption of the images before and after editing such as [15, 34, 38, 9]. These methods require extra information not directly related to editing, such as descriptions of the part of the image that remains constant through editing. On the other hand, the second category assumes that the text describes the instruction on what point should be changed through editing. InstructPix2Pix [4] edits the image based on a given text instruction, which has the advantage of making image editing easier and more intuitive. They first create a dataset of editing instructions and image descriptions before and after editing with GPT-3 [5]. Then, they input it to [9] to generate edited images. After that, they fine-tune StableDiffusion to generate these edited images conditioned by the source images and text instructions. Thus, InstructPix2Pix learns to edit images according to various text instructions.

In this paper, we extend InstructPix2Pix to 3D data and realize the task of editing a 3D scene into a new 3D scene with text instructions. This enables highly intuitive editing of 3D scenes.

2.3. Implicit 3D Representation

There are various ways to represent 3D information [8, 1, 3, 14, 18, 21]. In this paper, we adopt implicit representation for its high expressive power, which has become quite popular since being used in NeRF [21]. Given images that capture a target scene from different viewpoints, NeRF builds a neural network that predicts color and density at any spatial point in the scene from its 3D coordinates. This model is trained so that each viewpoint image matches the image rendered from the corresponding viewpoint based on the model. After training, NeRF implicitly acquires a 3D representation and we can obtain images seen from any camera viewpoint.

Recently, voxel grid-based implicit 3D representations have also been studied. They retain color and density information in the form of voxel grids instead of neural networks [33, 7, 17] and achieve much faster training. In this study, we use voxel grid-based DVGO [33] to represent 3D scenes for fast 3D-to-3D conversion.

2.4. Stylization of 3D scenes

3D stylization is a task to transform a source 3D scene into a new scene that has a different style while preserving

the content of the original scene. In [39, 22], the style is specified by a reference image. On the other hand, CLIP-NeRF [35] accepts texts to specify the style, which provides substantially higher flexibility. They finetune NeRF of the original scene so that the images rendered from any viewpoint would have high similarity in the CLIP feature space with the given text. There is also a concurrent work [36] to ours for text-guided 3D stylization, which calculates a CLIP-based contrastive loss based on the source and target image to properly strengthen the 3D stylization.

These methods only match the CLIP features of each viewpoint image and the reference image or text and do not use a generative model. Hence, there is no guarantee that they can convert the input 3D scene into a high-quality one. In this paper, we use a diffusion model to maximize the likelihood of each viewpoint image. In addition, while previous studies required a description of the converted 3D scene, we use editing instructions as input to make the 3D-to-3D conversion more intuitive.

2.5. Text-to-3D models

DreamFields [13] was the first study to realize Text-to-3D. DreamFields generates a 3D scene that follows the input text from any viewpoint by optimizing the CLIP features of each NeRF viewpoint image to match the input text. However, since DreamFields only optimizes the CLIP features and does not use a generative model, it may lead to generating unrealistic scenes that just cheat the similarity at the CLIP feature space.

DreamFusion [26] is the first method to apply diffusion models to the Text-to-3D task. DreamFusion uses pre-trained Imagen [30] to generate 3D scenes by optimizing each NeRF viewpoint image $\boldsymbol{x}$ to follow the input text. Specifically, they first apply noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$ to the viewpoint image $\boldsymbol{x}$ according to the randomly sampled t, and obtain a noisy image $\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$. This noisy image is used to calculate the gradient of the loss function $\nabla_\theta \mathcal{L}_{\mathrm{SDS}}$ as the following equation:

$$\nabla_\theta \mathcal{L}_{\mathrm{SDS}} = \mathbb{E}_{t,\epsilon}\left[w(t)(\epsilon_\phi(\boldsymbol{x}_t; y, t) - \boldsymbol{\epsilon})\frac{\partial \boldsymbol{x}}{\partial \theta}\right] \quad (4)$$

where θ is the parameters of NeRF and y is the text description of the 3D scene to be generated. θ is updated using this gradient from any viewpoint. This method enables the generation of high-quality 3D scenes for a variety of text inputs. They also edit generated 3D scenes by re-training them with new texts. However, direct finetuning of a 3D scene may result in a scene that is far removed from the original 3D scene. In addition, this method requires a text description of the scene after conversion, and conversion by text instructions is not possible.

Algorithm 1 Proposed method: Instructed 3D-to-3D

Input:
y: instruction text,
θ_{src}: parameters of source model,
g_θ: volume rendering function from a model parameterized with θ
Output:
θ_{tgt}: parameters of target model

$\theta_{\mathrm{tgt}} \leftarrow \theta_{\mathrm{src}}$ # initialize θ_{tgt}
for $i = 1$ to N_{iter} **do**
  # rendering from source & target model
  $c = \mathrm{random\_camera\_pose}()$
  $I_{\mathrm{src}} = g_{\theta_{\mathrm{src}}}(c)$
  $I_{\mathrm{tgt}} = g_{\theta_{\mathrm{tgt}}}(c)$
  $L_{\mathrm{tgt}} = \mathrm{StableDiffusionEncoder}(I_{\mathrm{tgt}})$
  # add noise
  $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$
  $t \sim U[1, \ldots, T]$
  $\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}L_{\mathrm{tgt}} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$
  # calculate the gradient of the loss function
  $\nabla_{\theta_{\mathrm{tgt}}}\mathcal{L}_{\mathrm{SDS}} = \mathbb{E}_{t,\epsilon}\left[w(t)(\tilde{\epsilon}_\phi(\boldsymbol{x}_t; y, I_{\mathrm{src}}, t) - \boldsymbol{\epsilon})\frac{\partial L_{\mathrm{tgt}}}{\partial \theta_{\mathrm{tgt}}}\right]$
  $\theta_{\mathrm{tgt}} \leftarrow \mathrm{Adam}(\theta_{\mathrm{tgt}}, \nabla_{\theta_{\mathrm{tgt}}}\mathcal{L}_{\mathrm{SDS}})$
end for

3. Proposed Method

3.1. Pipeline of Instruct 3D-to-3D

Figure 2 shows the overview of Instruct 3D-to-3D. Our proposed method converts a source model, which is an implicit representation of a source 3D scene, into a new target model along with the text instruction.

The main idea of our Instruct 3D-to-3D is to learn the target model from an arbitrary viewpoint using InstructPix2Pix conditioned by the source scene and the text instruction. First, the target model is initialized with the source model. Next, using the target model, a target image I_{tgt} is rendered from a random camera viewpoint and is fed into the encoder of StableDiffusion to obtain the corresponding latent feature L_{tgt}. We add noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$ to it to make noisy latent $\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}L_{\mathrm{tgt}} + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$. A source image I_{src} is also rendered from the same viewpoint as the target image using the source model. $\boldsymbol{x}_t$ is input to InstructPix2Pix along with the source image I_{src} and the text instruction y as conditions. As InstructPix2Pix has two conditions, I_{src} and y, the noise is estimated as follows.

$$\begin{aligned}\tilde{\epsilon}_\phi(\boldsymbol{x}_t; y, I_{\mathrm{src}}, t) = \ & \epsilon_\phi(\boldsymbol{x}_t; \varnothing, \varnothing, t) \\ & + s_I \cdot (\epsilon_\phi(\boldsymbol{x}_t; \varnothing, I_{\mathrm{src}}, t) - \epsilon_\phi(\boldsymbol{x}_t; \varnothing, \varnothing, t)) \\ & + s_T \cdot (\epsilon_\phi(\boldsymbol{x}_t; y, I_{\mathrm{src}}, t) - \epsilon_\phi(\boldsymbol{x}_t; \varnothing, I_{\mathrm{src}}, t))\end{aligned} \quad (5)$$

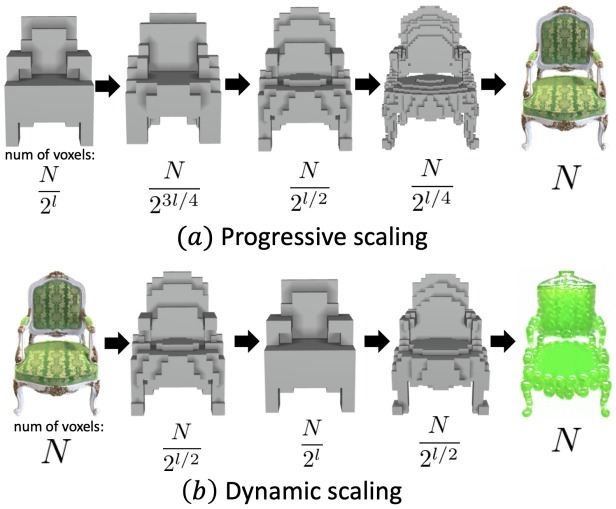

Figure 3: Comparison of progressive scaling and proposed dynamic scaling. In dynamic scaling, we first gradually decrease the number of voxels to change the global structure, then increase them to change to the local structure.

where s_I is a hyper-parameter that determines the degree of fidelity to the information of the source image, and s_T is a hyper-parameter that determines the degree of fidelity to the text instruction. In this way, our proposed method explicitly incorporates the source image and the text instruction. Similar to DreamFusion, we update the target model by the following gradient.

$$\nabla_{\theta_{\mathrm{tgt}}} \mathcal{L}_{\mathrm{SDS}} = \mathbb{E}_{t,\epsilon}\left[w(t)(\tilde{\epsilon}_\phi(\boldsymbol{x}_t; y, I_{\mathrm{src}}, t) - \boldsymbol{\epsilon}) \frac{\partial L_{\mathrm{tgt}}}{\partial \theta_{\mathrm{tgt}}} \right] \tag{6}$$

where θ_{tgt} is the parameters of the target model, ϕ is the parameters of InstructPix2Pix, and $w(t)$ is a scheduling coefficient and we set it as $1 - \bar{\alpha}_t$ in the experiments.

By repeating this procedure from randomly chosen camera viewpoints, an arbitrary viewpoint image of the target model becomes the image appropriately converted from the same viewpoint image of the source model. As a result, we can obtain a target 3D scene converted from the source 3D scene along with the text instruction.

3.2. Dynamic Scaling

In this study, we use DVGO [33] to perform fast 3D-to-3D conversions. DVGO is one of the voxel grid-based implicit 3D representations and maintains density and color information in the form of a 3D voxel grid. The voxel grid is a discrete partition of 3D space, with each vertex holding color and density information. Volume rendering is performed with the interpolated information of the vertices around the ray.

The resolution of the 3D scene is determined by the number of voxels used in the model. DVGO performs progressive scaling [17] to gradually increase the number of voxels during training as shown in Figure 3 (a). It encourages the model to learn the rough geometry of the scene first, and then gradually move on to learning the fine details of the scene.

In our 3D-to-3D task, the number of voxels in the target model is initially set to N, the same as in the source model. In this situation, the voxel grid is too fine and it is difficult to change the geometry, and only the appearance changes. Therefore, we propose dynamic scaling, which gradually reduces the number of voxels from N to $N/2^l$ during the 3D-to-3D conversion process and then gradually returns it to N. Figure 3 (b) shows this process. This allows gradual changes in the global structure, followed by changes in the detailed structure accordingly. Also, we can adjust the magnitude of the structure transformation by adjusting the scaling factor l. For large l, the number of voxels becomes small, which allows for a large structural transformation. Conversely, for smaller l, the structural transformation is suppressed.

4. Experiments

4.1. Experimental Settings

We implemented the proposed Instruct 3D-to-3D with PyTorch [25] and performed 3D-to-3D conversion with NeRF synthetic dataset [21] and Local Light Field Fusion (LLFF) dataset [20]. We resized the NeRF synthetic dataset to 256×256 and the LLFF dataset to 378×504 for training. In both datasets, we first trained the source models and then converted them with Instruct 3D-to-3D. The text instructions were manually designed. The source and target models are constructed with DVGO and the number of voxels N is set as $1,024,000$.

In the 3D conversion process, we trained the target model for 2,000 iterations. We set the dynamic scaling factor $l = 4$. With dynamic scaling, we reduced the voxel number by a factor of $1/2^{l/5}$ every 150 iterations for the first 750 iterations and increased it by $2^{l/5}$ every 150 iterations for the next 750 iterations. After that, we kept the voxel number the same and trained for the remaining 500 iterations. Our proposed method can be completed in 15 minutes with a single NVIDIA A100 GPU.

We set the text guidance scale s_T of Eq. 5 to 100 according to [26]. The image guidance scale s_I was set to 5 for the NeRF synthetic dataset and 100 for the LLFF dataset. We used the larger s_I for the LLFF dataset to strongly maintain 3D consistency since it has only front-view images.

We also used CLIP-NeRF and DreamFusion as our baseline methods. Both were set up with the same experimental settings as the proposed method except for the loss function

Figure 4: Qualitative comparison between the proposed method and baselines.

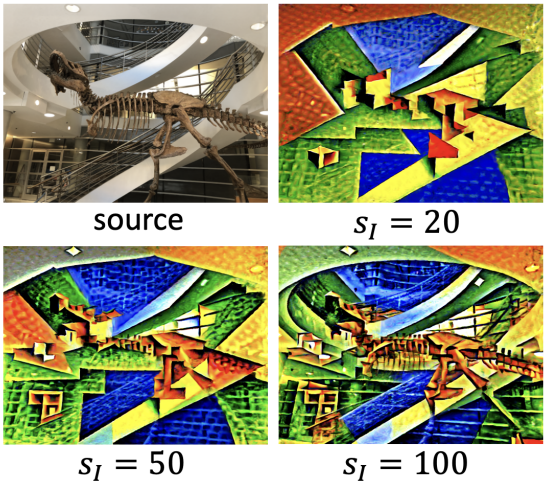

Figure 5: Effects of s_I in the converted 3D scenes. We can preserve the structure of the source scene with larger s_I.

and the target texts. The target texts were also manually designed to match what the text instructions used in Instruct 3D-to-3D pointed to.

When fine-tuning source 3D scenes with DreamFusion, we used open-source StableDiffusion [29] instead of Imagen [30].

4.2. Qualitative Evaluations

We show the qualitative comparison between the proposed method and baseline methods in Figure 4. The results in the top four rows of Figure 4 are examples from the NeRF synthetic dataset and results in the bottom three rows are examples from the LLFF dataset. The text instructions used in Instruct 3D-to-3D and the target texts used in DreamFusion and CLIP-NeRF are also shown in Figure 4. As a whole, it can be seen that the proposed method can produce converted 3D scenes that accurately reflect both the text instructions and structures of the source 3D scenes. Although CLIP-NeRF can convert 3D scenes from the NeRF synthetic dataset relatively cleanly, it cannot convert 3D scenes from the LLFF dataset well, resulting in noisy 3D scenes. In addition, DreamFusion does not explicitly incorporate source 3D scenes as conditions, so it converts the LLFF 3D scenes into completely different 3D scenes.

Figure 5 shows the differences by changing the image guidance scale s_I in Eq. 5. We can manipulate the degree to which the structure of the source 3D scene is reflected by adjusting the value of s_I. For large s_I, the source image condition is strongly incorporated during the noise estimation and the structure of the source 3D scene is strongly reflected.

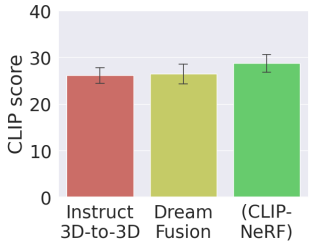

Figure 6: Average of CLIP scores (higher is better).

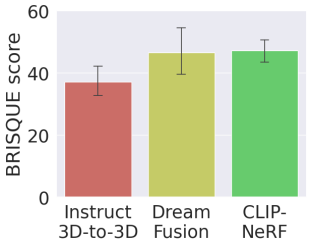

Figure 7: Average of BRISQUE scores (**lower is better**).

4.3. Quantitative Evaluations

For quantitative evaluation, we measured CLIP score [27] and BRISQUE score [23]. The CLIP score is a measure of the semantic alignment of image-text pairs, where higher is better. The BRISQUE score is a measure of image quality, where lower means better.

We performed 3D-to-3D conversion by creating ten 3D scene-text pairs for each of the proposed and baseline methods. The input texts were manually designed so that the target texts used in DreamFusion and CLIP-NeRF match what the text instructions in Instruct 3D-to-3D point to. The list of scene-text pairs used in this experiment is shown in the supplementary material. For each converted 3D scene, we rendered images from 100 viewpoints and used them to measure scores. We did this for 10 converted 3D scenes, for a total of 1,000 rendered images used in the evaluation. The CLIP score was calculated using the rendered images and the target texts, while the BRISQUE score was calculated from the rendered images only.

Figure 7 shows the BRISQUE score measurement results, and Figure 6 shows the CLIP score measurement results. Regarding the BRISQUE score, Instruct 3D-to-3D performs better than DreamFusion and CLIP-NeRF and achieves higher-quality 3D-to-3D conversion than these baseline methods. On the other hand, the CLIP scores for Instruct 3D-to-3D and DreamFusion are comparable. Although CLIP-NeRF achieves the best CLIP score, we cannot compare CLIP-NeRF with the other methods in terms of CLIP score in a fair manner. This is because CLIP-NeRF directly uses CLIP for training to improve the CLIP score. From these results, we confirmed that Instruct 3D-to-3D is able to convert high-quality 3D scenes with high text fidelity comparable to DreamFusion. The reason for this is considered to be that Instruct 3D-to-3D simultaneously incorporates the source 3D scene as a condition in addition to the text instruction, which contributes to a natural 3D scene conversion.

4.4. User Study

To evaluate the perceptual quality of the converted 3D scenes, we conducted a user study. We used the results of

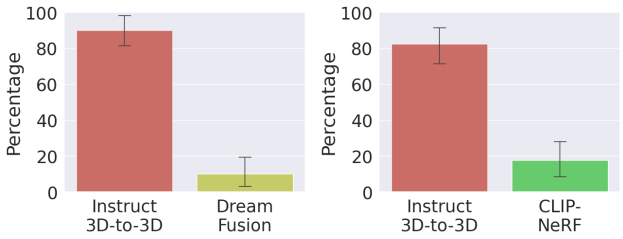

Figure 8: User study results. Preference rates for 3D conversion quality of Instruct 3D-to-3D over DreamFusion and CLIP-NeRF.

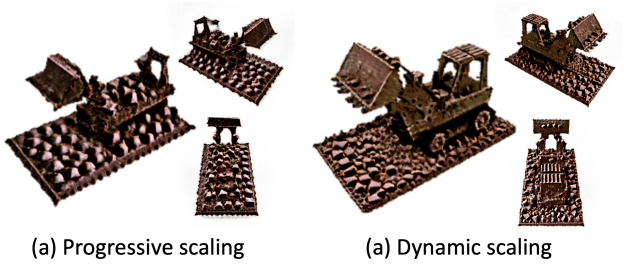

Figure 9: Comparison of 3D conversion results by scaling method. our dynamic scaling does not break the 3D structure.

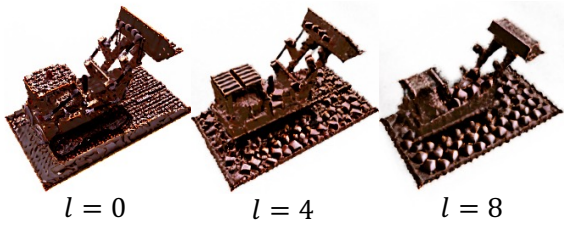

Figure 10: Effects of dynamic scaling factor l in the 3D conversion. The large l causes a major change in structure.

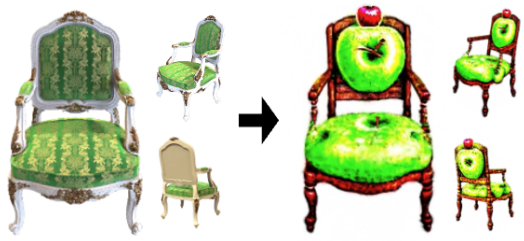

Figure 11: A failure case of 3D-to-3D conversion. This 3D scene is converted from a 3D scene of a chair with the text instruction "put an apple on the chair".

the 10 patterns of 3D-to-3D conversions used in the section 4.3 as test cases for the user study. In each test case, we first showed the source 3D scene as a video and the text instruction. Then we showed two of each of the three 3D scenes transformed with the proposed method and baseline methods in random order. The participants chose the better converted 3D scene by jointly considering the following four aspects: image quality from any viewpoint, 3D consistency, fidelity to the text instruction, and fidelity to the source 3D scene. We did not set a time limit. We finally collected 25 questionnaires. Figure 8 shows the results of the user study. Our proposed method outperforms the baseline methods with a much higher user preference rate.

4.5. Sensitivity to the Scaling Strategy

We proposed dynamic scaling to gradually change the 3D structure. However, it is also possible to use conventional progressive scaling in 3D-to-3D conversion. Figure 9 shows a comparison of 3D conversion results using progressive scaling and dynamic scaling. As progressive scaling greatly reduces the number of voxels at the beginning of the conversion, the detailed 3D structure cannot be maintained. Our dynamic scaling gradually changes the number of voxels, resulting in small and repairable 3D structural damage and preserving the 3D structure.

Figure 10 shows the differences in 3D-to-3D conversion results by changing the dynamic scaling factor l. For larger l, the resolution of the voxel becomes smaller during the conversion process and the structure can be changed significantly.

4.6. Limitations

Figure 11 is an example of a failure case. We gave the instruction to place an apple on the chair, but the seat and backrest of the chair changed to apples. Similar to InstructPix2Pix, our proposed method is difficult to follow instructions that require spatial reasonings. To solve this problem, it is necessary to handle spatial information, for example, by taking the depth information of the 3D scene into account. We leave this improvement to future work.

5. Conclusion

In this study, we proposed Instruct 3D-to-3D, which achieves a high-quality 3D-to-3D conversion following the text instruction. We proposed dynamic scaling, which allows manipulation of the strength of the 3D structure transformation and smooth 3D-to-3D conversion. We performed quantitative and qualitative evaluations and showed that our Instruct 3D-to-3D outperforms the baseline methods in terms of the quality of the converted 3D scenes and the fidelity to the source 3D scenes and text instructions. Our proposed method makes 3D content easier to edit and use, and will contribute to greatly expanding the scope of various content productions.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas J. Guibas. Learning representations and generative models for 3d point clouds. In *ICML*, 2018.

[2] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *ICCV*, 2021.

[3] André Brock, Theodore Lim, James M. Ritchie, and Nick Weston. Generative and discriminative voxel modeling with convolutional neural networks. *CoRR*, abs/1608.04236, 2016.

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

[5] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, 2020.

[6] Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat gans on image synthesis. In *NeurIPS*, 2021.

[7] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, 2022.

[8] Matheus Gadelha, Subhransu Maji, and Rui Wang. 3d shape induction from 2d views of multiple objects. In *3DV*, 2017.

[9] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *CoRR*, abs/2208.01626, 2022.

[10] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey A. Gritsenko, Diederik P. Kingma, Ben Poole, Mohammad Norouzi, David J. Fleet, and Tim Salimans. Imagen video: High definition video generation with diffusion models. *CoRR*, abs/2210.02303, 2022.

[11] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *CoRR*, abs/2207.12598, 2022.

[12] Qingqing Huang, Daniel S. Park, Tao Wang, Timo I. Denk, Andy Ly, Nanxin Chen, Zhengdong Zhang, Zhishuai Zhang, Jiahui Yu, Christian Havnø Frank, Jesse H. Engel, Quoc V. Le, William Chan, and Wei Han. Noise2music: Text-conditioned music generation with diffusion models. *CoRR*, abs/2302.03917, 2023.

[13] Ajay Jain, Ben Mildenhall, Jonathan T. Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. In *CVPR*, 2022.

[14] Angjoo Kanazawa, Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, 2018.

[15] Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models. *CoRR*, abs/2210.09276, 2022.

[16] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. *CoRR*, abs/2211.10440, 2022.

[17] Stephen Lombardi, Tomas Simon, Jason M. Saragih, Gabriel Schwartz, Andreas M. Lehrmann, and Yaser Sheikh. Neural volumes: learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 38(4):65:1–65:14, 2019.

[18] Lars M. Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, 2019.

[19] Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. Latent-nerf for shape-guided generation of 3d shapes and textures. *CoRR*, abs/2211.07600, 2022.

[20] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: practical view synthesis with prescriptive sampling guidelines. *ACM Trans. Graph.*, 38(4):29:1–29:14, 2019.

[21] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[22] Shailesh Mishra and Jonathan Granskog. Clip-based neural neighbor style transfer for 3d assets. *CoRR*, abs/2208.04370, 2022.

[23] Anish Mittal, Anush Krishna Moorthy, and Alan Conrad Bovik. No-reference image quality assessment in the spatial domain. *IEEE Trans. Image Process.*, 21(12):4695–4708, 2012.

[24] Gautam Mittal, Jesse H. Engel, Curtis Hawthorne, and Ian Simon. Symbolic music generation with diffusion models. In *ISMIR*, 2021.

[25] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

[26] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *ICLR*, 2023.

[27] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[28] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with CLIP latents. *CoRR*, abs/2204.06125, 2022.

[29] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

[30] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. In *NeurIPS*, 2022.

[31] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, Devi Parikh, Sonal Gupta, and Yaniv Taigman. Make-a-video: Text-to-video generation without text-video data. In *ICLR*, 2023.

[32] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015.

[33] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *CVPR*, 2022.

[34] Dani Valevski, Matan Kalman, Yossi Matias, and Yaniv Leviathan. Unitune: Text-driven image editing by fine tuning an image generation model on a single image. *CoRR*, abs/2210.09477, 2022.

[35] Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Clip-nerf: Text-and-image driven manipulation of neural radiance fields. In *CVPR*, 2022.

[36] Can Wang, Ruixiang Jiang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Nerf-art: Text-driven neural radiance fields stylization. *CoRR*, abs/2212.08070, 2022.

[37] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A. Yeh, and Greg Shakhnarovich. Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. *CoRR*, abs/2212.00774, 2022.

[38] Chen Henry Wu and Fernando De la Torre. Unifying diffusion models' latent space, with applications to cyclediffusion and guidance. *CoRR*, abs/2210.05559, 2022.

[39] Kai Zhang, Nicholas I. Kolkin, Sai Bi, Fujun Luan, Zexiang Xu, Eli Shechtman, and Noah Snavely. Arf: Artistic radiance fields. In *ECCV*, 2022.