4D Instruct-GS2GS

Eldor Fozilov
Computer Science and Engineering, UNIST
eldorfozilov@unist.ac.kr

MD Khalequzzaman Chowdhury Sayem
Computer Science and Engineering, UNIST
khalequzzamansayem@unist.ac.kr

Mubarrat Tajoar Chowdhury
Computer Science and Engineering, UNIST
mubarrattajoar@unist.ac.kr

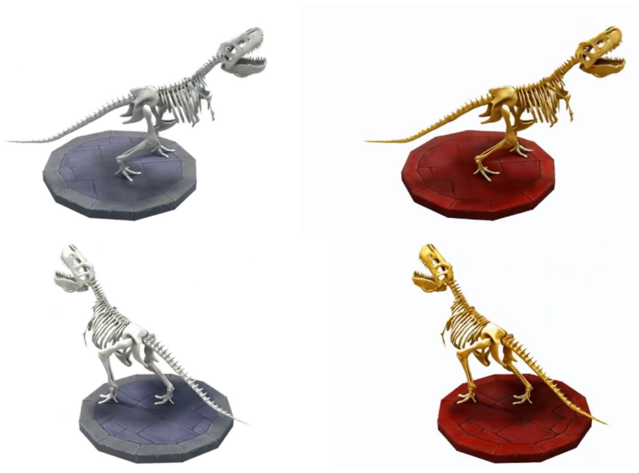
Figure 1. **Prompt**: *Turn the skeleton into gold and the stone into red*

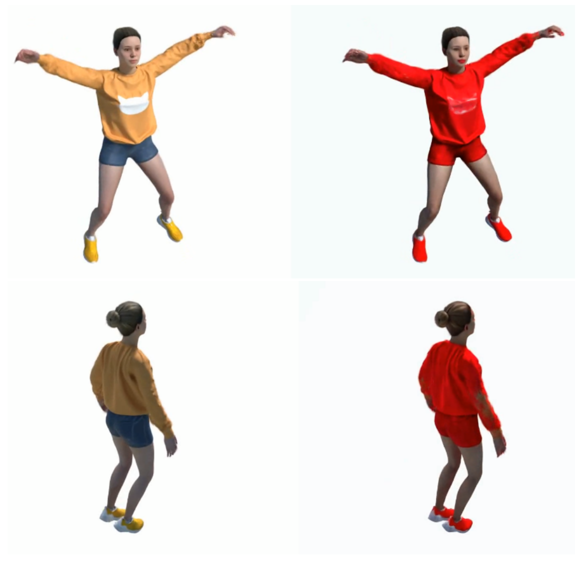
Figure 2. **Prompt**: *Change the outfit color into red*

Abstract

Editing dynamic scenes is crucial in various real-world applications, such as virtual reality, augmented reality, and film production, where precise control and modification of scenes are necessary. Building on the advancements of Instruct-NeRF2NeRF and Instruct-GS2GS, we propose 4D Instruct-GS2GS, which extends semantic editing capabilities to dynamic scenes. This method leverages an iterative dataset update technique for consistent 3D and temporal editing of Gaussian splatting scenes via text instructions. Our approach enables realistic global text edits on large real-world scenes and individual subjects, incorporating the temporal dimension while maintaining rapid training times (approximately 30 minutes per scene) and high rendering speeds (up to 82 FPS). This work significantly enhances the practicality and versatility of dynamic scene editing in 3D vision and computer graphics.

1. Introduction

The realm of 3D vision and computer graphics has long sought efficient methods for rendering dynamic scenes. Traditional approaches often struggle with the complexity of accurately modeling motions and ensuring high efficiency. To address these challenges, 4D Gaussian Splatting (4D-GS) [9] emerges as a holistic representation for dynamic scenes. It extends the concept of 3D Gaussian Splatting by incorporating time as an additional dimension, allowing for the modeling of complex motions with both spatial and temporal sparsity. In the work, they introduce gaussian deformation field, which predicts deformations of 3D Gaussians over time, allowing for the transformation of the scene's geometry to render novel views at different timestamps. To encode the features, a decomposed neural voxel encoding algorithm inspired by HexPlane [2] was used to

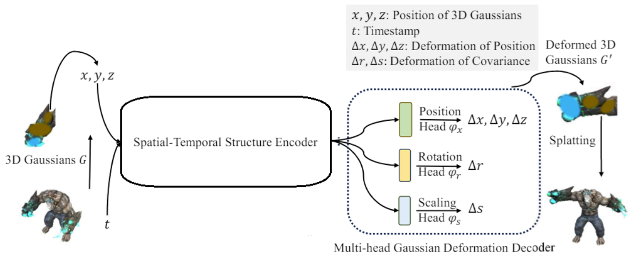
x, y, z: Position of 3D Gaussians
t: Timestamp
$\Delta x, \Delta y, \Delta z$: Deformation of Position
$\Delta r, \Delta s$: Deformation of Covariance
Deformed 3D Gaussians G'
x, y, z
3D Gaussians G
t
Spatial-Temporal Structure Encoder
Position Head φ_x
$\Delta x, \Delta y, \Delta z$
Rotation Head φ_r
Δr
Scaling Head φ_s
Δs
Splatting
Multi-head Gaussian Deformation Decoder

Figure 3. 4D Gaussian Splatting module

build Gaussian features from 4D neural voxels efficiently. This allows the method to achieve high rendering speeds (up to 82 FPS at 800×800 resolution) while maintaining quality, making it suitable for applications like VR and AR.

Recent advancements in photorealistic 3D representations, such as Neural Radiance Fields (NeRF) [5] and 3D Gaussian Splatting (3DGS) [4], have paved the way for innovative exploration in 3D generation, neural reconstruction, and practical applications. However, editing these novel 3D representations remains a challenge due to the incompatibility of traditional 3D tools with these formats. Building upon the success of Instruct-NeRF2NeRF [3] and Instruct-GS2GS [8], which introduced semantic editing of NeRFs and gaussian splating respectively using text instructions, we propose 4D Instruct-GS2GS which extends the capability of editing beyond static scenes.

4D Instruct-GS2GS not only captures spatial dimensions but also incorporates the temporal aspect, allowing for dynamic scene representation and editing. Our method extends Instruct-GS2GS [8], originally designed for 3DGS [4], and is now adapted to handle the complexities of 4D editing. This work introduces a modified technique that applies the iterative dataset update method for consistent 3D and temporal editing of Gaussian splatting scenes using text-based instructions. In order to update the dataset, InstructPix2Pix [1] was employed to perform editing on the rendered images from 4D-GS conditioned on the real dataset images and the text prompt. For more details, please refer to Figure 4 and Section 2.2.

We demonstrate that our approach can perform realistic global text edits on large real-world scenes and individual subjects, with the added dimension of time, while maintaining a faster training (around 30 minutes per scene) and rendering speed (up to 82 FPS in a RTX 3090 GPU, same rendering speed of [9]) compared to its predecessors.

2. Method

2.1. Preliminaries

2.1.1 3D gaussians

3D Gaussians [4] is an explicit 3D scene representation in the form of point clouds, characterized by a covariance matrix Σ and a center point $\mathcal{X}$, the mean value of a Gaussian:

$$G(\mathcal{X}) = e^{-\frac{1}{2}\mathcal{X}^T\Sigma^{-1}\mathcal{X}} \tag{1}$$

Each 3D Gaussian is characterized by position $\mathcal{X} \in \mathbb{R}^3$, color defined by spherical harmonic (SH) coefficients $C \in \mathbb{R}^k$ (where k represents nums of SH functions), opacity $\alpha \in \mathbb{R}$, rotation factor $r \in \mathbb{R}^4$, and scaling factor $s \in \mathbb{R}^3$. Given these attributes we can define a gaussian and then proceed to blending and rendering step to for 2D image a viewing point from these gaussians. For details we refer the readers to [4].

2.1.2 4D gaussians

For 4D gaussian representation of video scenes in this work, we adopt [9]. Given a view matrix $M = [R, T]$ at a specific timestamp t, the 4D Gaussian splatting framework consists of 3D Gaussians G alongside a Gaussian deformation field network $\mathcal{F}$. The process of rendering a novel-view image $\hat{I}$ involves differential splatting, denoted by S, and follows the formula $\hat{I} = S(M, G')$, where G' is defined as $G' = \Delta G + G$. Fundamentally, the 4D Gaussian splatting approach converts the initial set of 3D Gaussians G into a modified set G' through the application of the Gaussian deformation field network $\mathcal{F}$. The overview of the 4D gaussian splatting framework that we adopt for this work has been shown in Figure 3. As an input, for a given viewing direction, positions of the existing Gaussians, x, y, z and time t of the video that is to be viewed are taken as an input. Then the batch of x, y, z, t are encoded into the Spatial-Temporal Structure Encoder module which adopts the K-planes [2] to reduce computational and memory complexity. After this encoding step, the multi-head Gaussian deformation decoder computes the deformation of position $\Delta\mathcal{X}$, rotation Δr, and scaling Δs. Then, the new deformed features $(\mathcal{X}', r', s')$ that are needed to update the gaussians, and represent the scene at t, can be addressed as:

$$(\mathcal{X}', r', s') = (\mathcal{X} + \Delta\mathcal{X}, r + \Delta r, s + \Delta s) \tag{2}$$

Following this, blending, rendering and training can be performed similarly as any 3D gaussians with these new parameters.

2.2. Our Proposed Pipeline

The 4D Gaussian Splatting (4DGS) module in our illustrated pipeline in Figure 4 is the same as [9], At first,

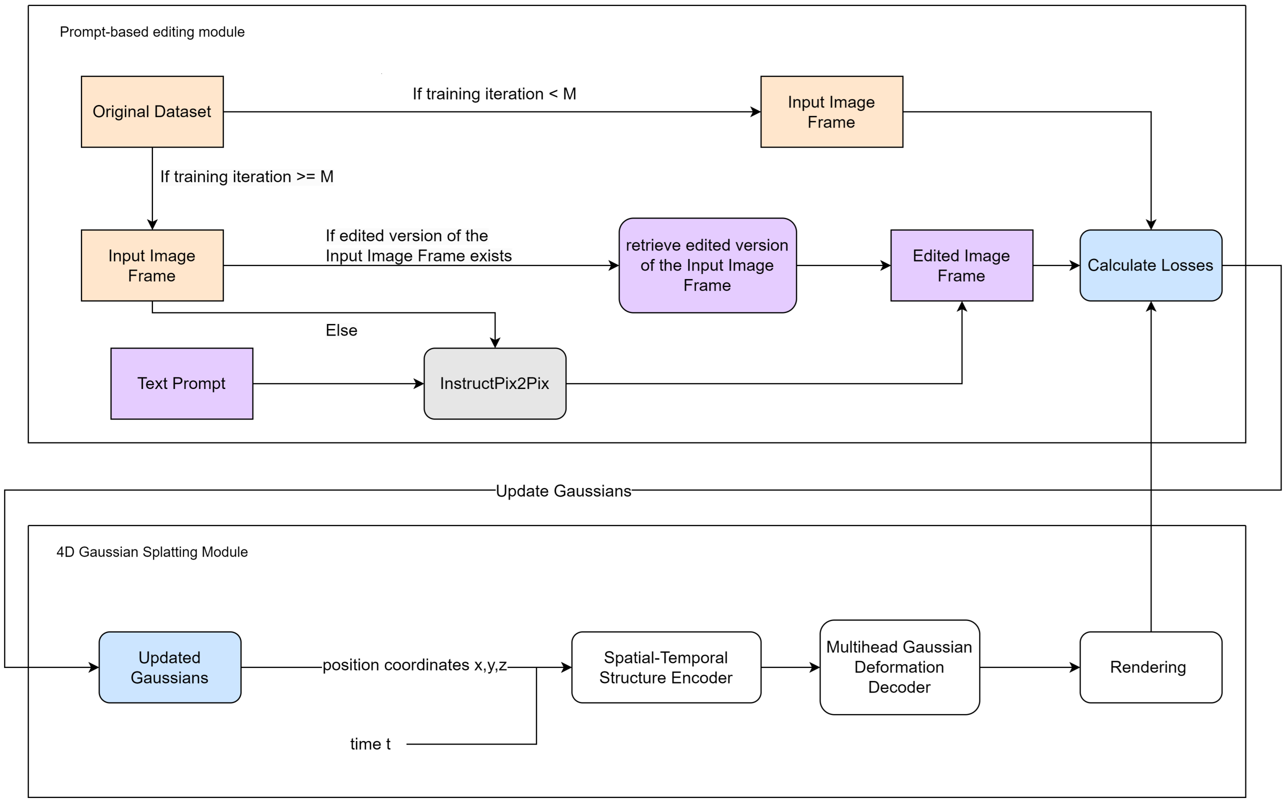
Prompt-based editing module
Original Dataset
If training iteration < M
Input Image Frame
If training iteration >= M
Input Image Frame
If edited version of the Input Image Frame exists
retrieve edited version of the Input Image Frame
Edited Image Frame
Calculate Losses
Else
Text Prompt
InstructPix2Pix
Update Gaussians
4D Gaussian Splatting Module
Updated Gaussians
position coordinates x,y,z
time t
Spatial-Temporal Structure Encoder
Multihead Gaussian Deformation Decoder
Rendering

Figure 4. Until iteration M, the 4D gaussian splatting (4DGS) module (shown at the bottom) uses multiview video data to train its gaussians. In gaussian update step, losses are calculated using the image rendered by 4DGS module and the input image from the original dataset. After iteration M, the entire multiview video dataset is eventually edited using the 2D diffusion module InstructPix2Pix which uses frames from the original dataset and a given text prompt (edit instruction) to create edited version of the original dataset. From iteration M and onwards, input images used to train the existing 3D gaussians are the edited versions of the input image frame instead.

upto a specified iteration *M*, training of gaussians proceeds normally with frames from the original dataset. where the 4DGS module learns to recreate novel views at novel time point by recreating image frames from viewing angles at time points available in the dataset. This is done to ensure gaussians trained upto that point are multiview consistent, sharp, and legible enough to perform edits on. Starting iteration *M*, when the losses, $L1$ color loss and grid-based total-variational loss L_{tv}, are calculated as usual in 4DGS splatting module, the ground-truth is replaced with the edited version of the ground-truth image instead. The edited image, guided by the user-given text prompt, is generated using InstructPix2Pix [1] following InstructGS2GS [8]. InstructPix2Pix is involved in the training pipeline until all the original input frames in the dataset (across all time points) have been edited in the past training iterations. Beyond this point, simply the edited images from the past are used, without needing InstructPix2Pix to be involved in the training routine. The entire routine involving InstructGS2GS and hence, InstructPix2Pix have been illustrated in the *Prompt-based editing module* in Figure 4. While InstructGS2GS, with InstructPix2Pix at its core, enabled consistent editing of static multiview scene represented using gaussians at a single single time point, with our design choices that include integrating InstructGS2GS with the 4D Gaussian Splatting module, our proposed pipeline goes beyond editing static 3D scenes and is capable of editing dynamic multiview 3D videos, ie, a 4D scene.

3. Experiments and Results

We perform edits on two datasets: D-NeRF [7] consists of synthetic videos with 50 to 200 frames, and HyperNeRF [6] consists of real-world video clips, from which we randomly extract 200 frames and construct the points using structure-from-motion for initialization.

We adapt similar parameters for the diffusion model from Instruct-GS2GS [8]. We vary the classifier-free guidance scales per edit and scene, using a range of values from sI = (1.0, 1.5) and sT = (7.5, 12.5). We edit the entire dataset and then train the scene for 2.5K iterations. We train our

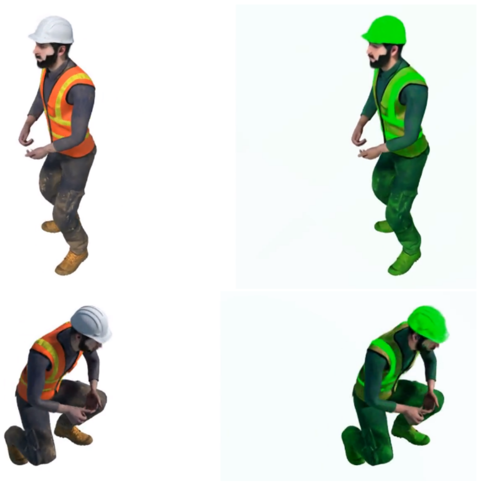
Figure 5. **Prompt**: *Change the outfit color into green*

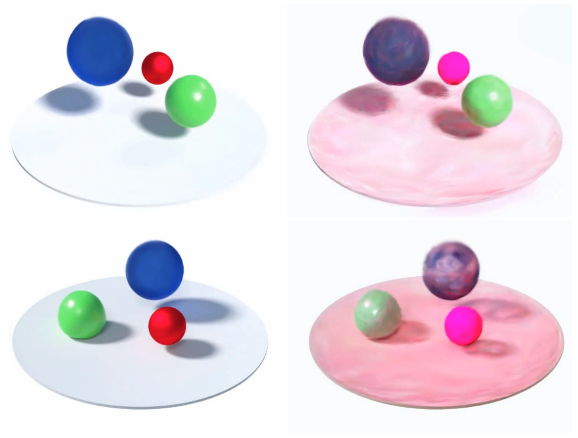
Figure 6. **Prompt**: *Turn the smallest ball into pink*

method for a maximum of 30K iterations: for the first 20K iterations, we train the 4D Gaussian Splatting module on the original dataset, and after that we update the dataset per each iteration, performing the editing in the last 10K iterations.

In Figures 1, 5, 2, where the left images come from the 4D GS module trained on the original data, we can see that the edits successfully follow the text instructions. Notably, in case of Figures 5, and 2 the hands and faces remain unedited, keeping the scene semantically meaningful. In Figure 7, which is a real-world scene, we can also observe that the edit was performed in an expected way.

3.1. Limitations

Though our method performs reasonably well in most of the cases, the instructed edits are at times applied beyond just the specified objects, which can be seen in Figures 6 and 8. This is due to the limitation of the InstructPix2Pix diffusion model adopted in the pipeline, which cannot be remedied by a simple data-update approach such as ours. Solving this could be deemed as an interesting future research direction.

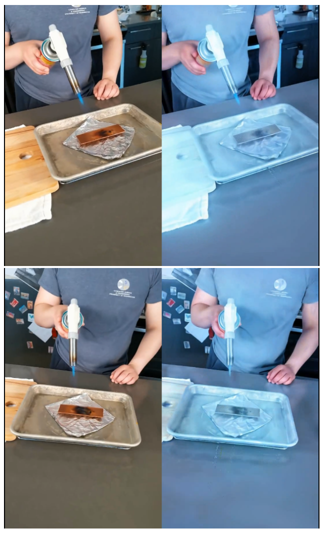
Figure 7. **Prompt**: *Turn the chocolate into ice*

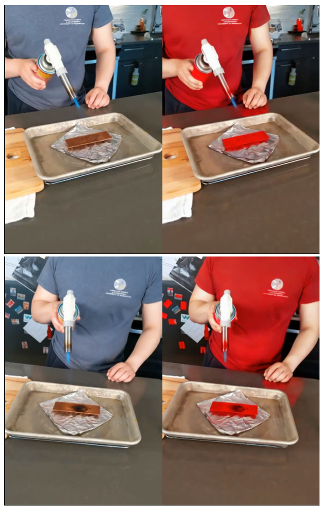
Figure 8. **Prompt**: *Turn the t-shirt into red*

4. Conclusion

In this paper, we have efficiently extended Instruct-GS2GS to 4D scenes, enabling real-time rendering and faster training times. Our method allows for intuitive and simple editing of 4D scenes using text instructions. On a variety of synthetic and real-world examples, we showed that our method can successfully edit the scene and maintain 4D-consistency in most cases.

References

[1] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. *arXiv preprint arXiv:2211.09800*, 2022. 2, 3

[2] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12479–12488, 2023. 1, 2

[3] Ayaan Haque, Matthew Tancik, Alexei Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. 2

[4] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. 2

[5] 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. *Communications of the ACM*, 65(1):99–106, 2021. 2

[6] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *ACM Trans. Graph.*, 40(6), dec 2021. 3

[7] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. *arXiv preprint arXiv:2011.13961*, 2020. 3

[8] Cyrus Vachha and Ayaan Haque. Instruct-gs2gs: Editing 3d gaussian splats with instructions, 2024. 2, 3

[9] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Wang Xinggang. 4d gaussian splatting for real-time dynamic scene rendering. *arXiv preprint arXiv:2310.08528*, 2023. 1, 2