3D Vision and
Robotics Lab

[CVPR 2024] Generative Rendering: Controllable 4D-Guided Video Generation with 2D Diffusion Models

Shengqu Cai[1,2], Duygu Ceylan[*2], Matheus Gadelha[*2], Chun-Hao Huang[2], Tuanfeng Wang[2], Gordon Wetzstein[1]
[1]Stanford University, [2]Adobe Research
*equal contribution

Gyeongsu Cho @UNIST

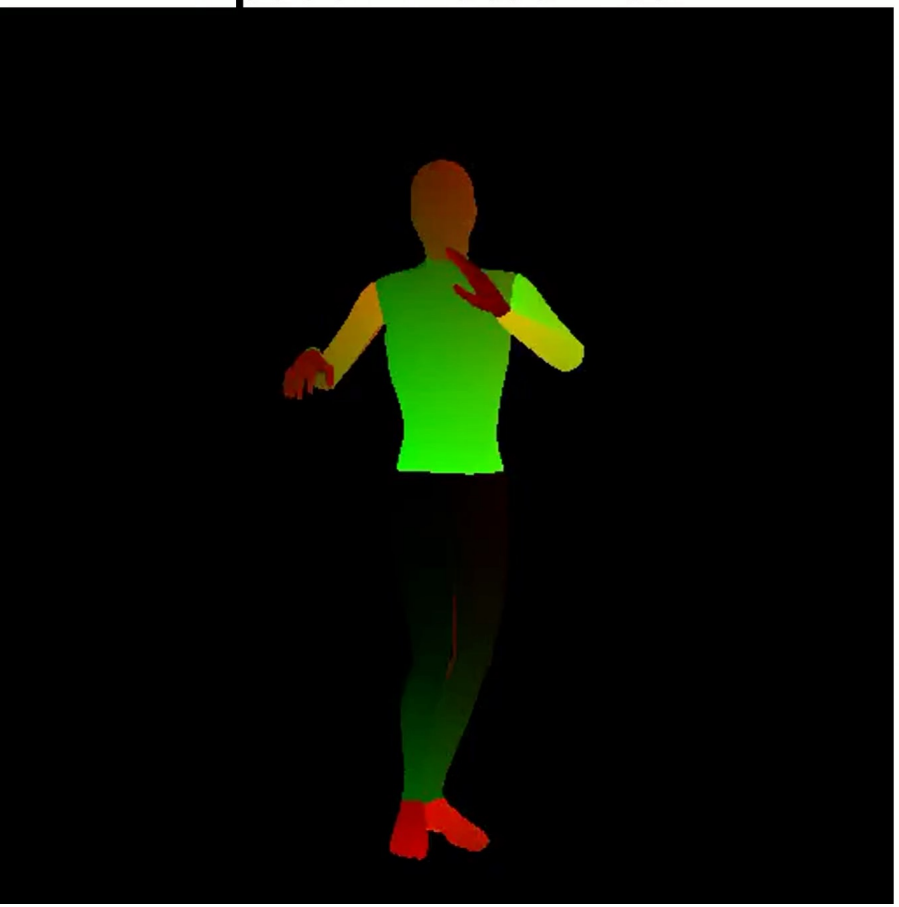
Lab Seminar 2024.05.09. (Thu)

Contents

Introduction

Controllable video generation?

input animated mesh



rendering prompt

a manga of the Joker dancing

rendered video

[CVPR 2024]Generative Rendering

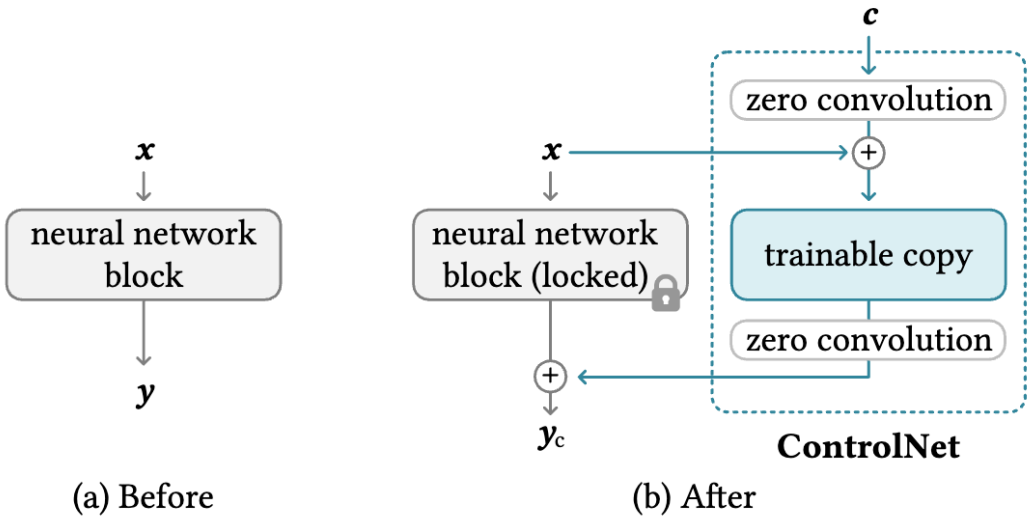
Introduction

Video Generation

OpenAI Sora

Introduction

Controllable image generation

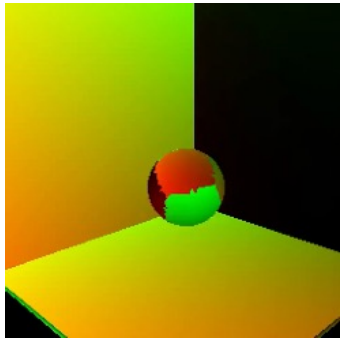


[ICCV 2023] ControlNet

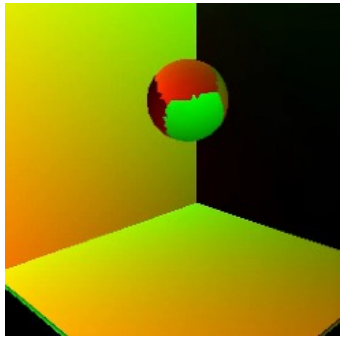
Introduction

Controllable image generation

Frame N



Frame M



Input

output

[ICCV 2023] ControlNet

ControlNet doesn't have consistency between frames.

Introduction

Video-to-video diffusion model

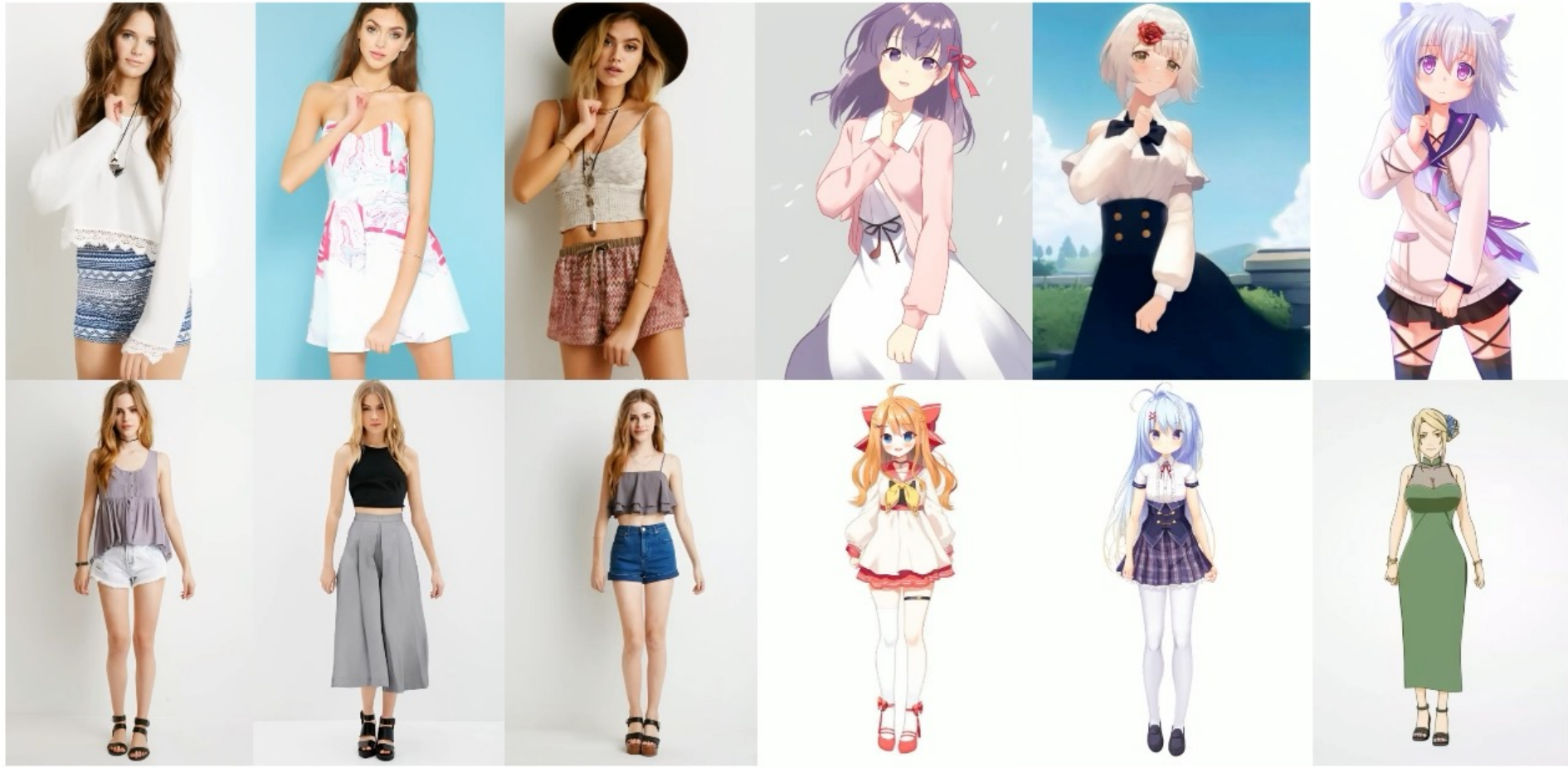
[ICCV 2023] Pix2Video

[ICLR 2024] TokenFlow

They require a high-fidelity video as input, which is not always available!

Introduction

Controllable Image-to-Video Synthesis



[arXiv 2023] Animate Anyone

We need a lot of training data to train Anymate Anyone!

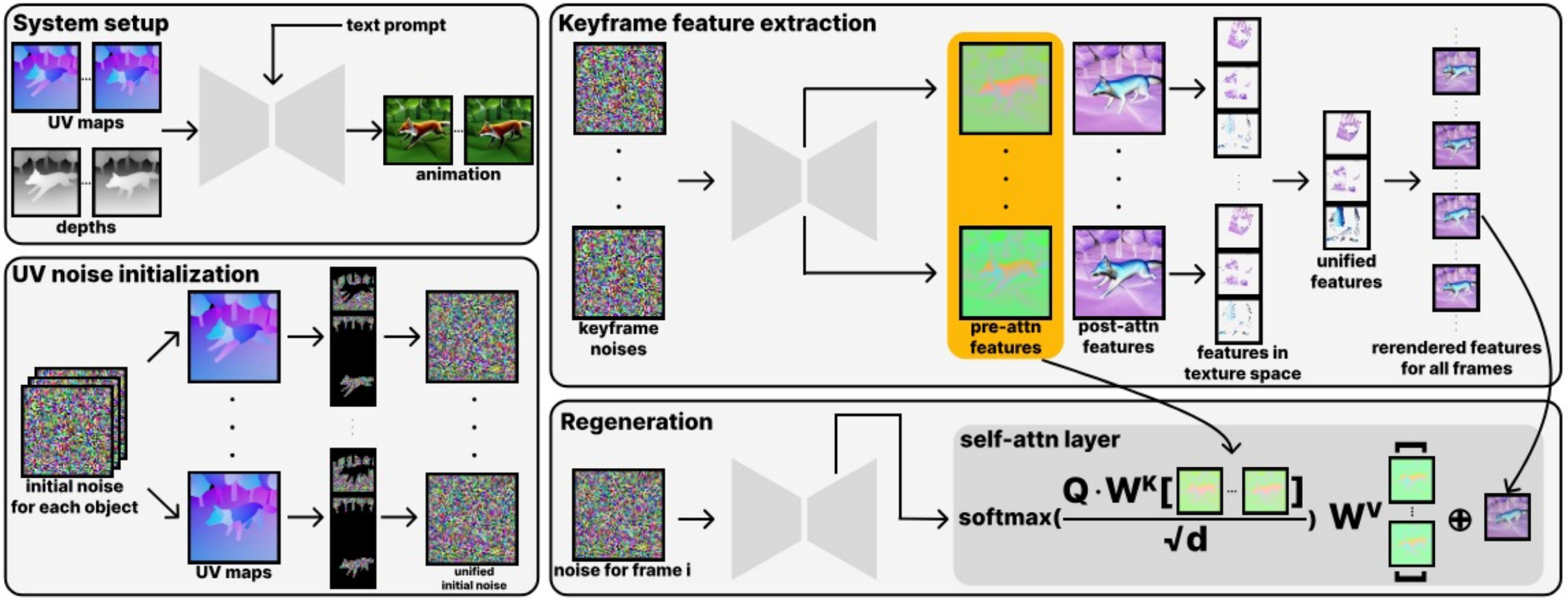
Introduction

Contributions

- They present a framework for 4D-guided animation synthesis utilizing the pre-trained T2I generation models as a multi-frame renderer.
- They enhance the self-attention layers of the image generation model by performing correspondence-aware blending of both input and output features to enforce consistent appearance synthesis.
- They introduce a UV-space noise initialization mechanism.

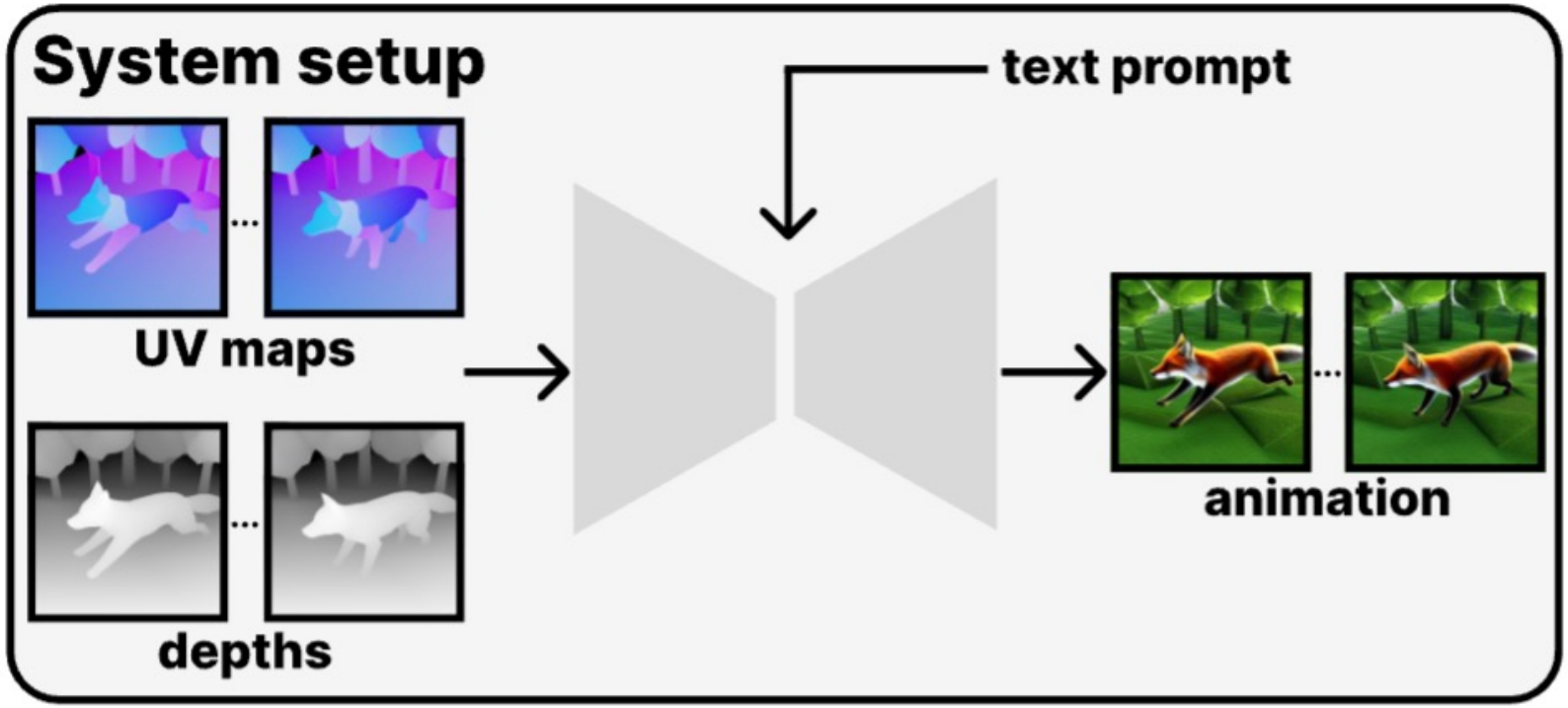
Method

Overview



Method

System setup



Input: ground truth UV maps, ground truth depth maps, text prompt

Output: animation video

Method

1. Self-attention feature injection

Naïve self-attention

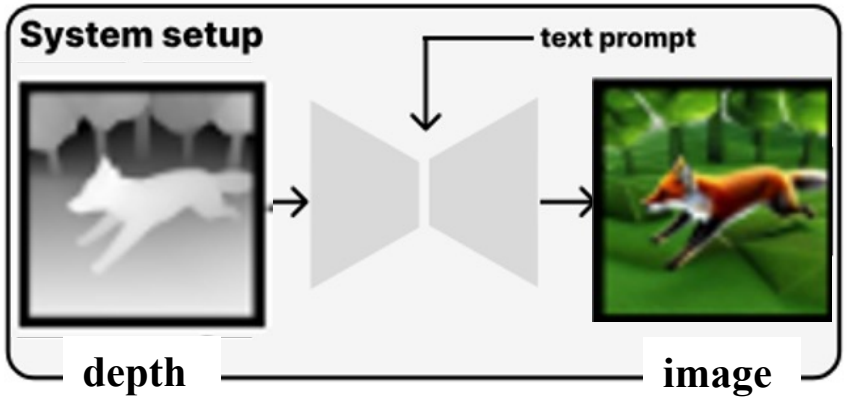
$$\boldsymbol{F} = \texttt{Attn}(\boldsymbol{Q};\boldsymbol{K};\boldsymbol{V}) = \texttt{Softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d}}\right)\cdot\boldsymbol{V}, \quad (1)$$

$$\boldsymbol{Q}^{(i,l)} = \mathbf{W}^Q \boldsymbol{f}^{(i,l)},\ \boldsymbol{K}^{(i,l)} = \mathbf{W}^K \boldsymbol{f}^{(i,l)},\ \boldsymbol{V}^{(i,l)} = \mathbf{W}^V \boldsymbol{f}^{(i,l)} \quad (2)$$

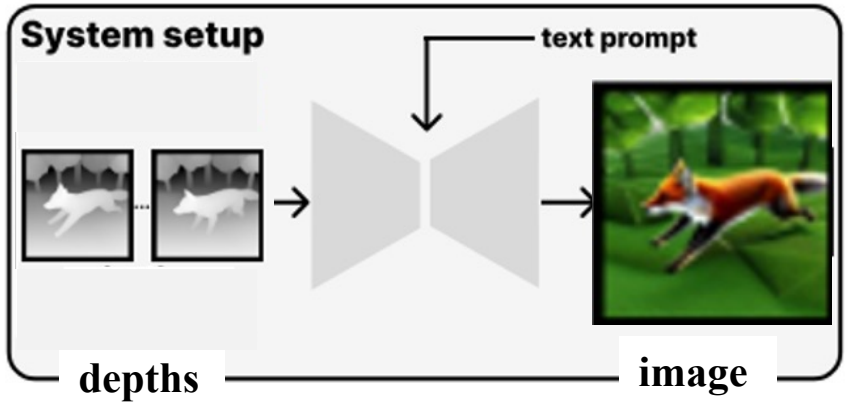
Self-attention feature injection

$$\boldsymbol{K}^{(i,l)} = \mathbf{W}^K[\boldsymbol{f}^{(1,l)}, ..., \boldsymbol{f}^{(N,l)}], \quad \text{pre-attn}$$
$$\boldsymbol{V}^{(i,l)} = \mathbf{W}^V[\boldsymbol{f}^{(1,l)}, ..., \boldsymbol{f}^{(N,l)}]. \quad (3)$$

$$\boldsymbol{F}^{(j,l)} = \pi_{i,j}(\boldsymbol{F}^{(i,l)}), \quad \text{post-attn} \quad (4)$$

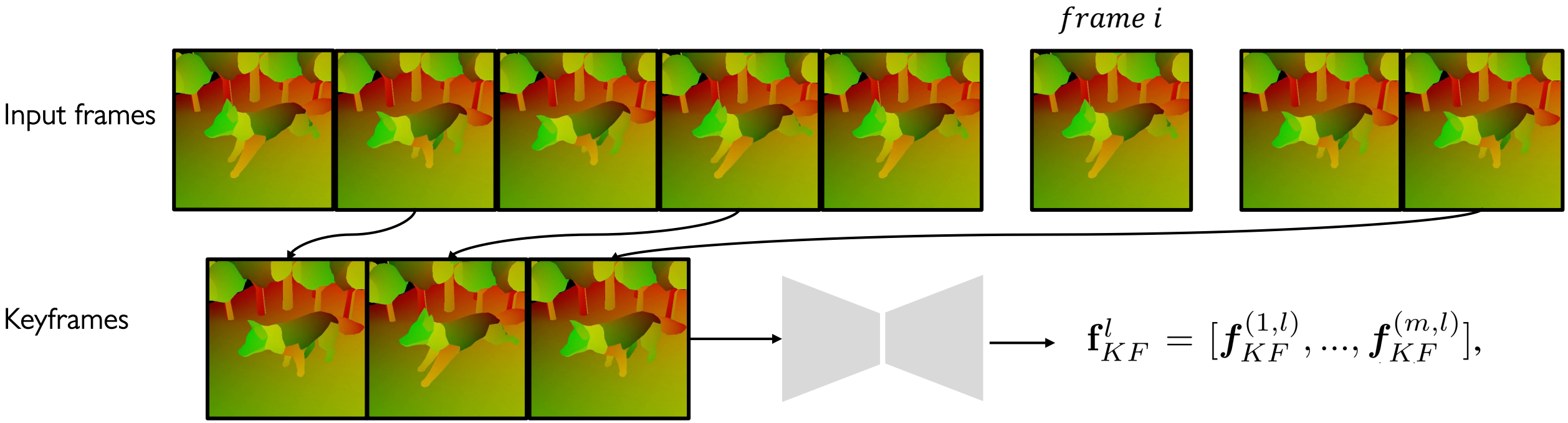


Per frame editting (ControlNet)



Method

2. UV-space feature injection



$$\mathbf{f}_{KF}^{l} = [\boldsymbol{f}_{KF}^{(1,l)}, \ldots, \boldsymbol{f}_{KF}^{(m,l)}],$$

$$\begin{aligned} \boldsymbol{K}^{(i,l)} &= \mathbf{W}^{K}[\boldsymbol{f}^{(1,l)}, \ldots, \boldsymbol{f}^{(N,l)}], \\ \boldsymbol{V}^{(i,l)} &= \mathbf{W}^{V}[\boldsymbol{f}^{(1,l)}, \ldots, \boldsymbol{f}^{(N,l)}]. \end{aligned} \tag{3}$$

$$\boldsymbol{F}^{(j,l)} = \pi_{i,j}(\boldsymbol{F}^{(i,l)}), \tag{4}$$

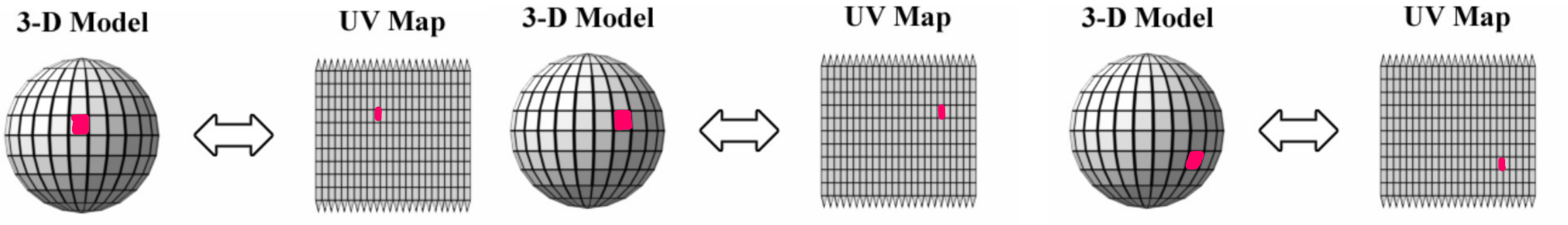
$$\begin{aligned} \boldsymbol{K}^{(i,l)} &= \mathbf{W}^{K}[\boldsymbol{f}^{(i,l)}, \mathbf{f}_{KF}^{l}], \\ \boldsymbol{V}^{(i,l)} &= \mathbf{W}^{V}[\boldsymbol{f}^{(i,l)}, \mathbf{f}_{KF}^{l}]. \end{aligned} \tag{5}$$

$$\boldsymbol{F}_{out}^{(i,l)} = \alpha \cdot \bar{\boldsymbol{F}}^{(i,l)} + (1-\alpha) \cdot \hat{\boldsymbol{F}}^{(i,l)} \tag{6}$$

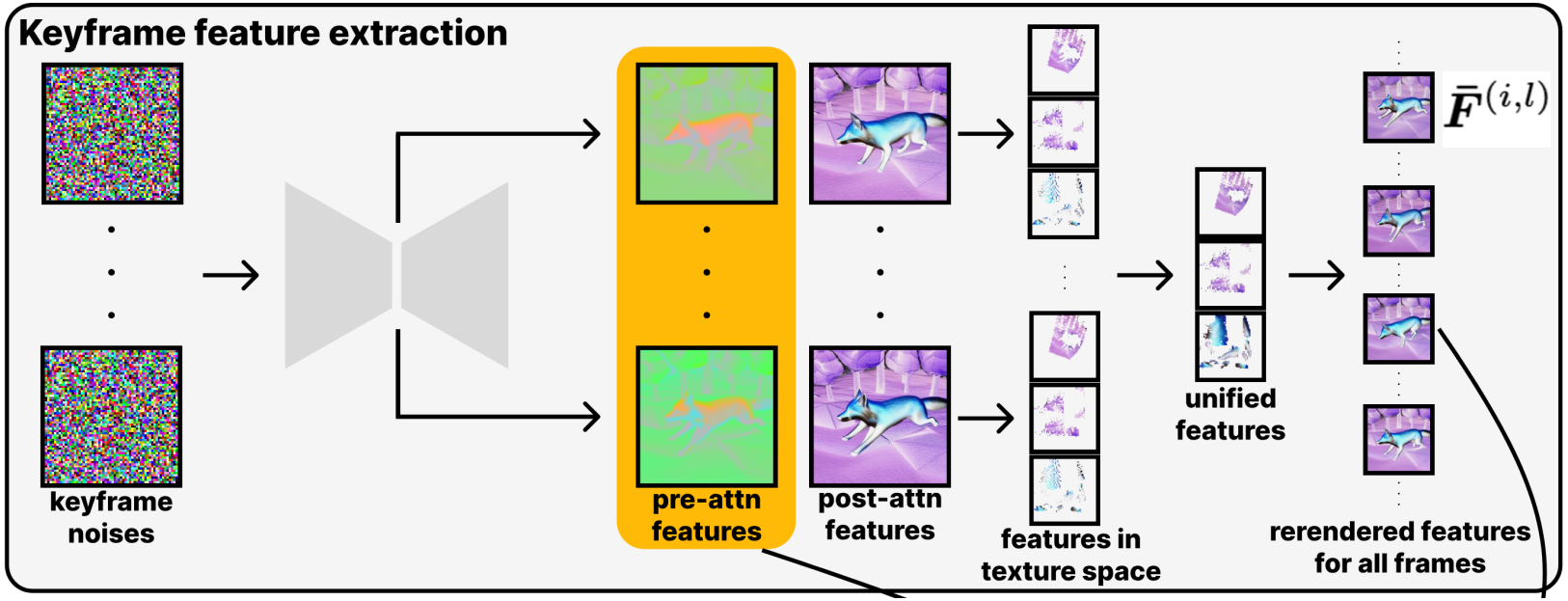
Method

2. UV-space feature injection

How to inject the UV-space feature?

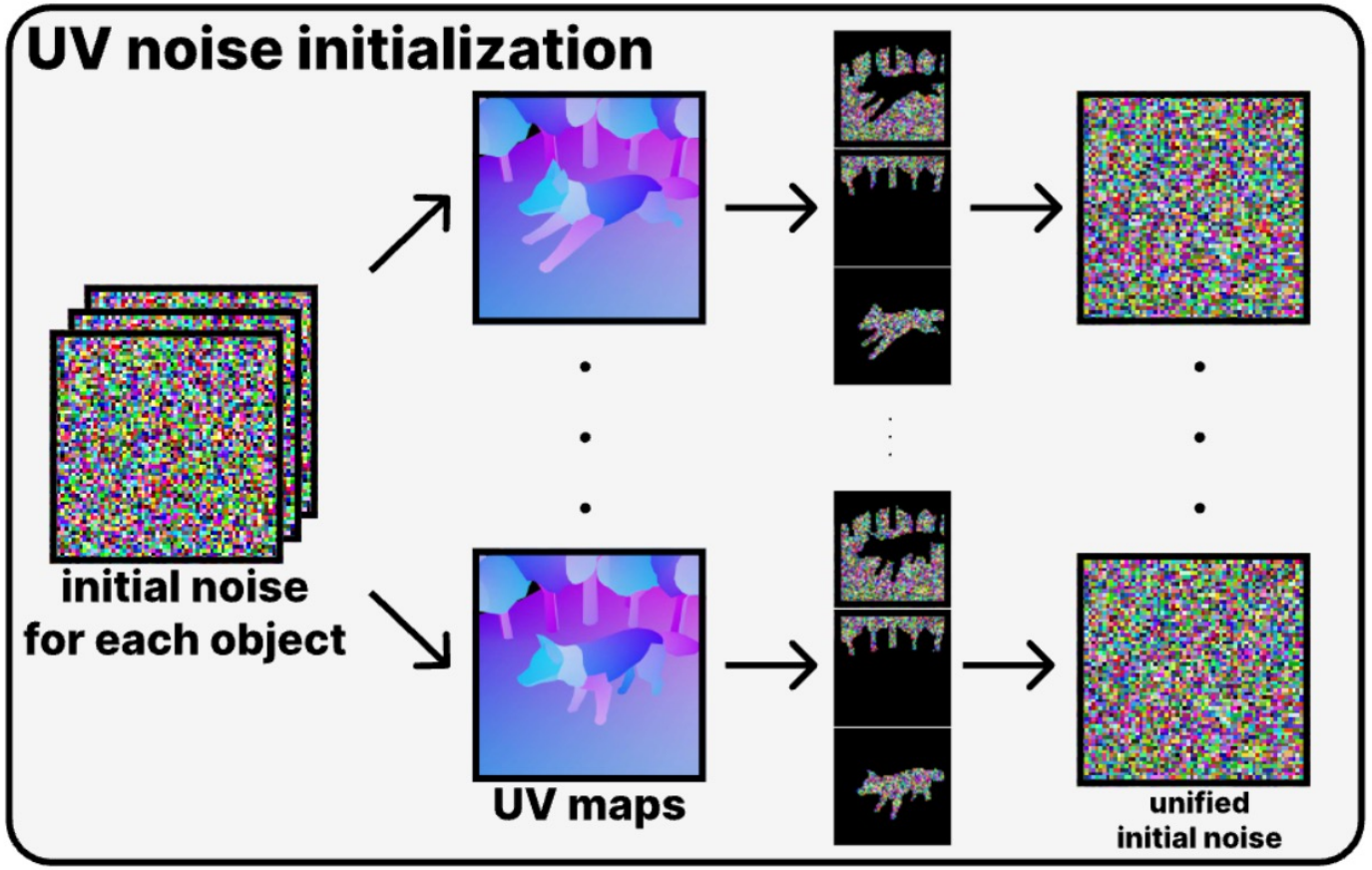


We can get UV map correspondences between frames from 3D assets!



Method

3. UV noise initialization



- Initialize the noise for each object by creating a Gaussian noise texture
- Project this noise to each frame by utilizing frame-UV correspondences

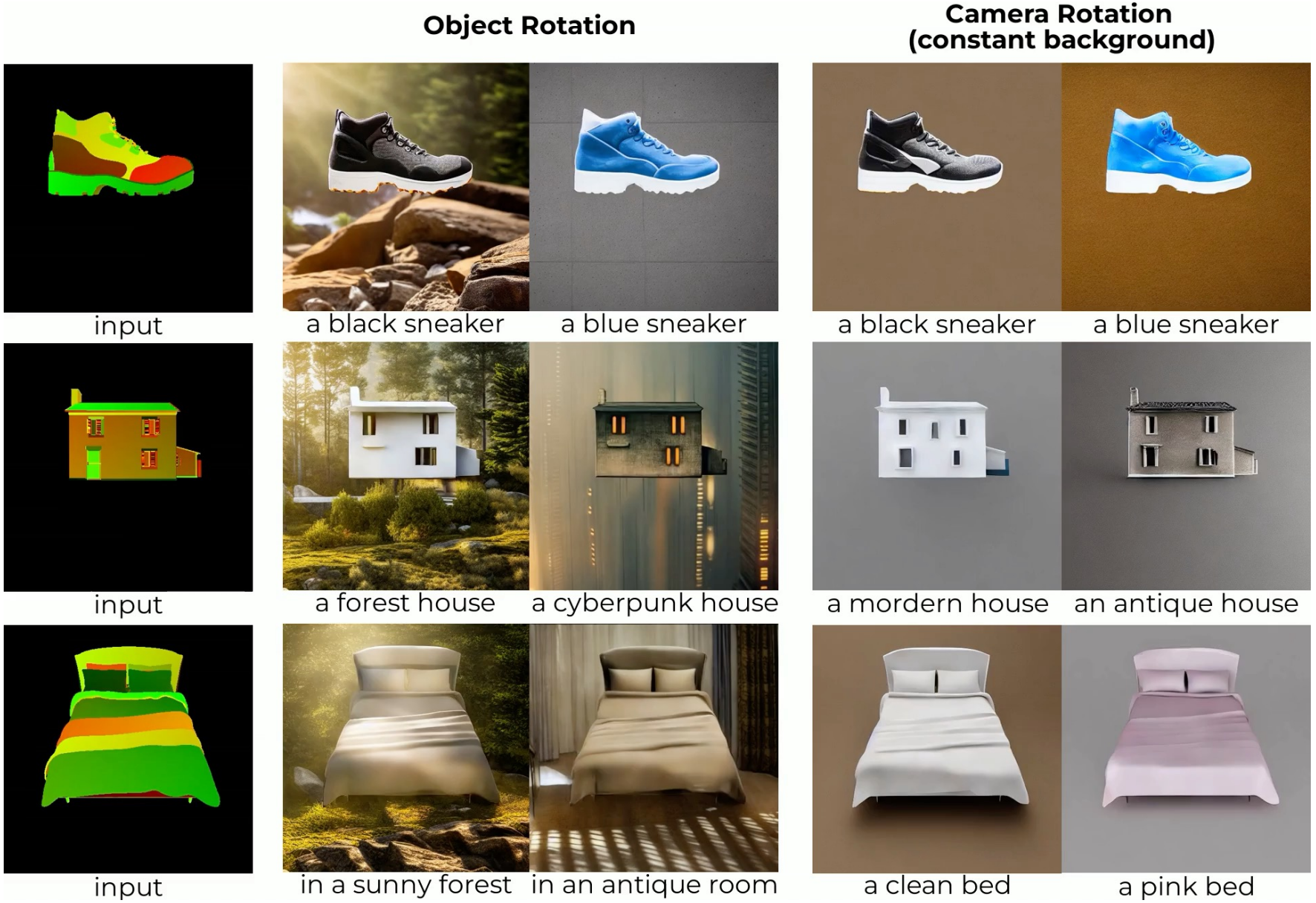
Experiments

- Dataset
  1. Camera rotations
  2. Physical simulation
  3. Character animation

- Baseline
  1. [ICCV 2023] Pix2Video
  2. [ICLR 2024] TokenFlow

- Evaluation metrics
  1. Frame consistency: CLIP cosine similarity between all pairs
  2. Prompt fidelity: mean CLIP embedding similarity score between video and prompt

Experiments

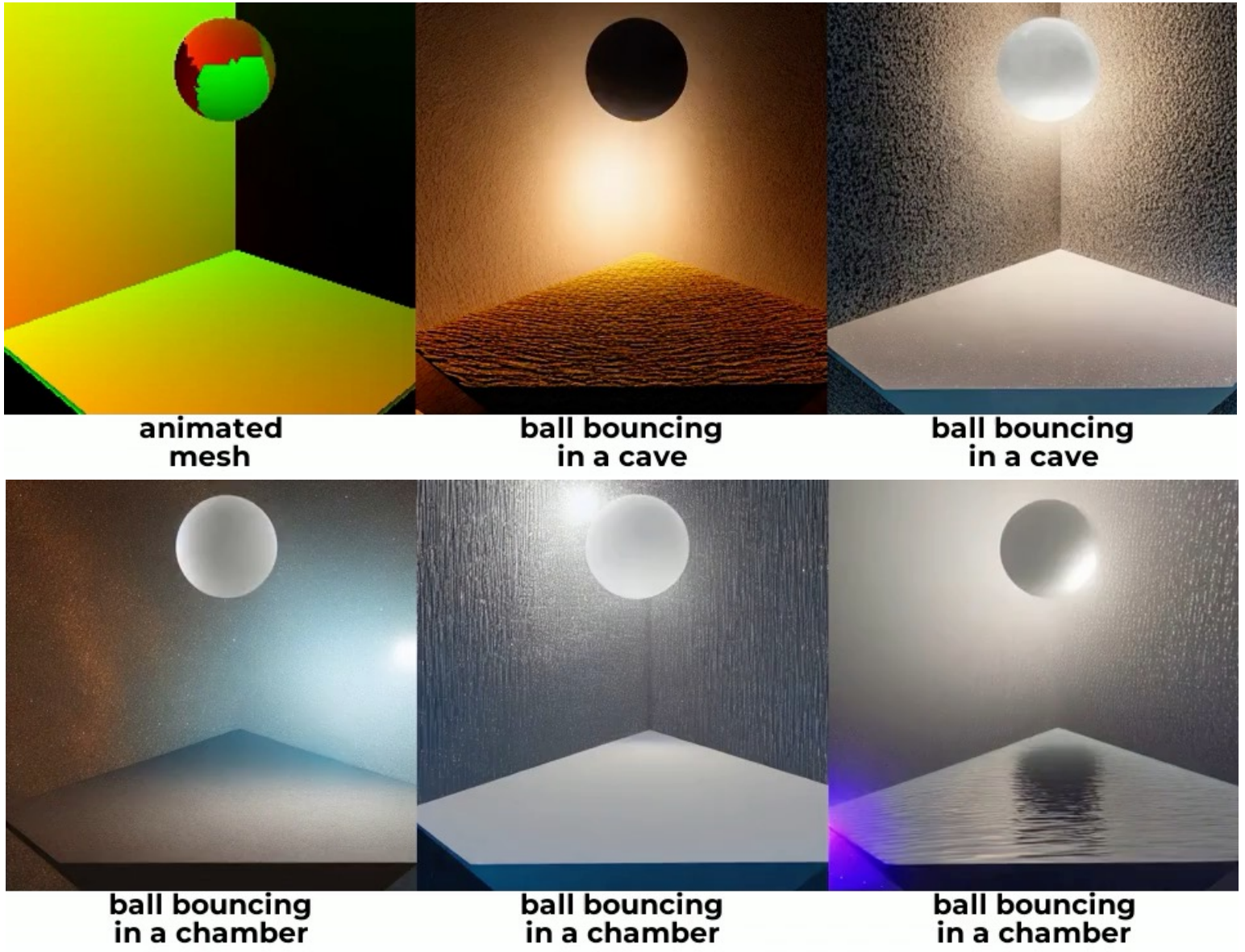


Camera rotations

Experiments

Character animation

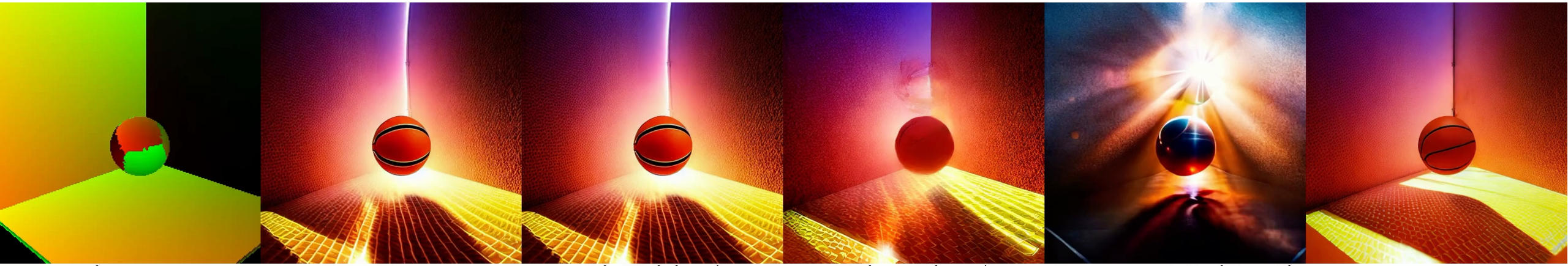
Experiments



Physical simulation

Experiments

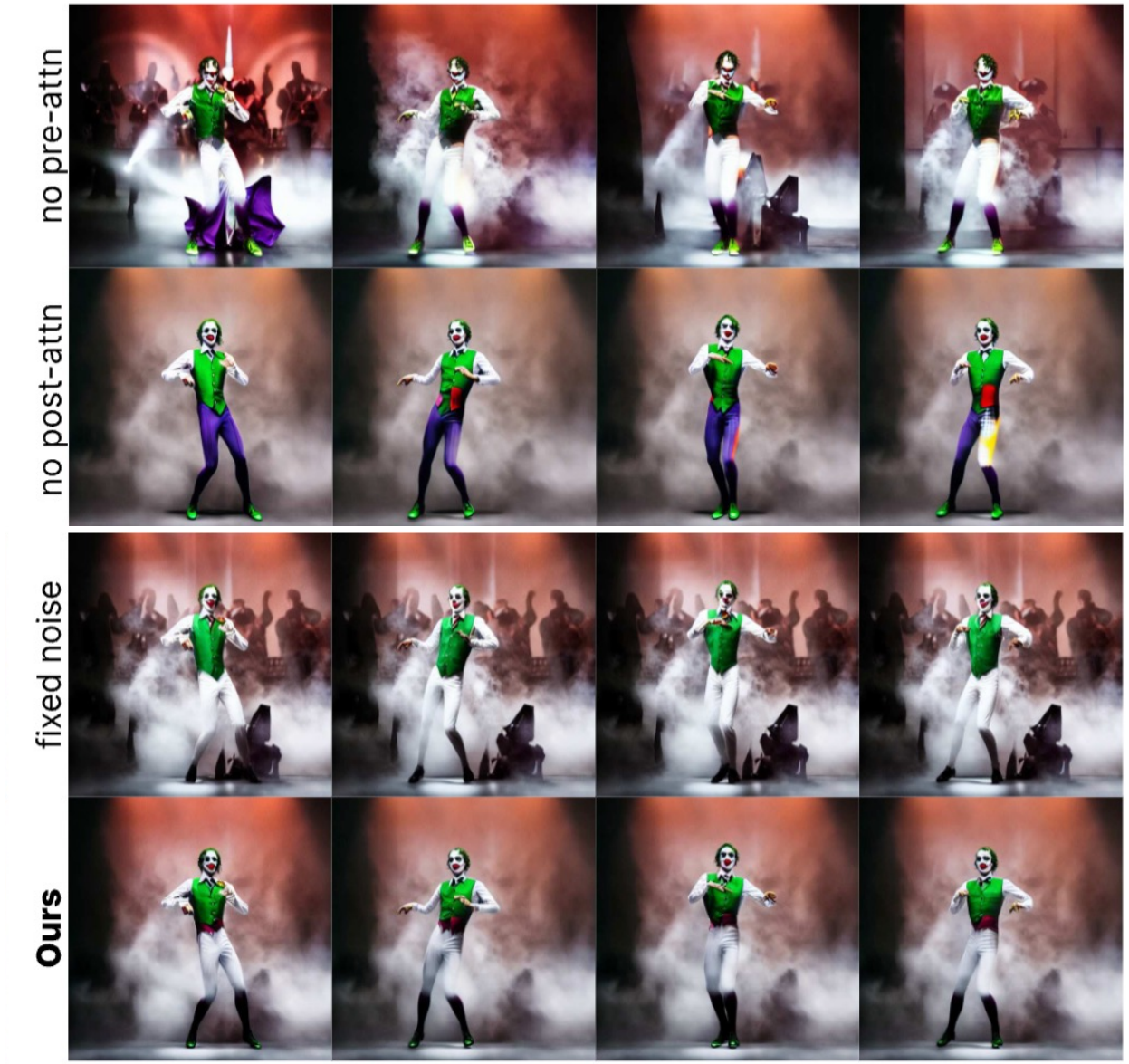
input | Per-Frame | Pix2Video* | TokenFlow* | Runway ML (Gen1) | **ours**



input | Per-Frame | Pix2Video* | TokenFlow* | Runway ML (Gen1) | **ours**

Qualitative comparisons

Experiments



Ablation study

Experiments

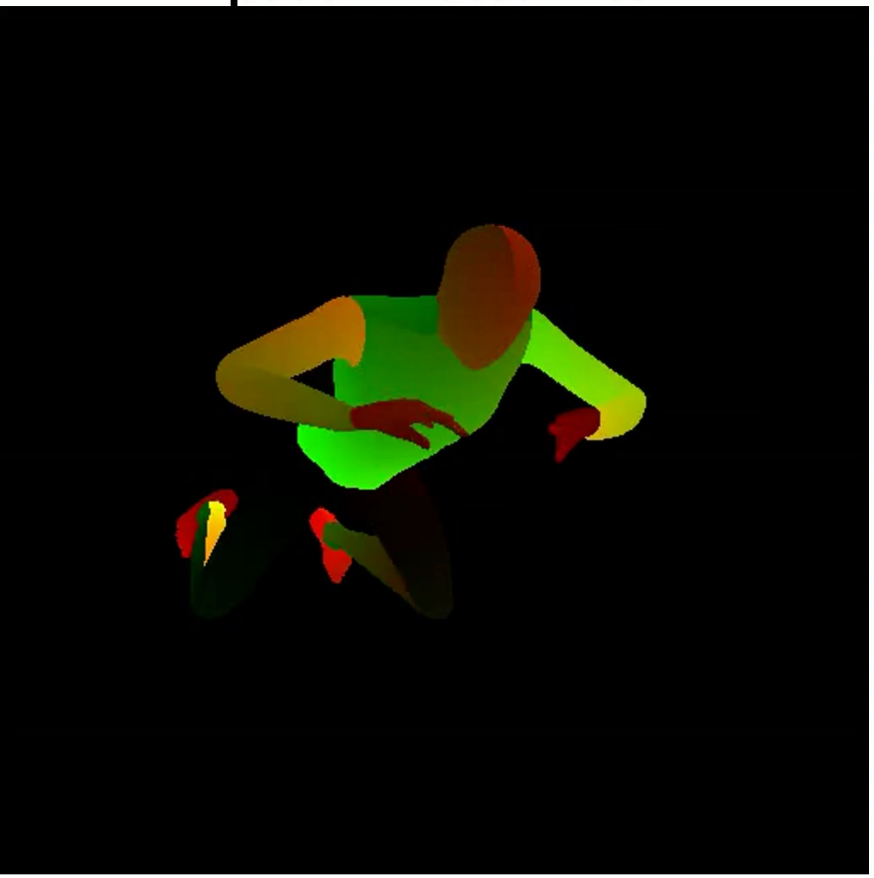
Method	Frame Consistency ↑	Prompt Fidelity ↑
Per-Frame	0.9547	**0.3233**
Pix2Video*	0.9630	0.2983
TokenFlow*	0.9822	0.2737
Gen1	**0.9907**	0.3029
Ours	<u>0.9845</u>	<u>0.3227</u>

	Pairwise Frame Interval				
Method	1	5	10	15	20
Per-Frame	0.9547	0.9173	0.9109	0.9145	0.9062
Pix2Video*	0.9630	0.9503	0.9494	0.9415	0.9471
TokenFlow*	0.9822	<u>0.9754</u>	<u>0.9728</u>	<u>0.9706</u>	<u>0.9712</u>
Gen1	**0.9907**	0.9715	0.9582	0.9624	0.9601
Ours	<u>0.9845</u>	**0.9815**	**0.9738**	**0.9749**	**0.9727**

Quantitative evaluation

Conclusion

input animated mesh



rendering prompt

a stormtrooper swimming

rendered video

Generative Rendering can animate creator-defined low-fidelity meshes and motion sequences, thereby bypassing steps requiring significant manual labor such as detailing, texturing, physical simulation.

Q&A