REGEN: Learning Compact Video Embedding with (Re-)Generative Decoder - Supplementary Material

Yitian Zhang[1,2] Long Mai[1] Aniruddha Mahapatra[1] David Bourgin[1]
Yicong Hong[1] Jonah Casebeer[1] Feng Liu[1] Yun Fu[2]

[1]Adobe Research [2]Northeastern University

https://bespontaneous.github.io/REGEN/

1. Additional Video Examples

Due to the space limit in the main paper, we provide additional video examples to validate our method and please check our project website for all the results displayed in a web UI.

2. Implementation Detail

2.1. Dataset

For video embedder training, the dataset is composed of 15 million videos and 300 million images. For text-to-video generation, we keep the same image resources and leverage 1 million videos to construct the dataset. The images are at the resolution of 256× with various aspect ratios, while all videos are at the resolution of 192×320.

2.2. Training

2.2.1. MAGVIT-v2

We reimplement MAGVIT-v2 [15] at the temporal compression ratio of $4\times$, $8\times$, $16\times$, $32\times$, and train the models at the resolution of 128×128 on our dataset with over 200K iterations. For $4\times$ temporal compression rate, we adopt the original 4-stage design and encode 17 frames into 5 latent frames at each chunk. For higher compression rate experiments, we add an additional stage to both encoder and decoder, and encode 17 frames into 3, 2 latent frames for temporal compression rate $8\times$, $16\times$, respectively. As for the extreme $32\times$ temporal compression rate, every 33 frames will be encoded into 2 latent frames. We train both MAGVIT-v2 and its discriminator with the AdamW [6] optimizer and we feed mixed images and videos to the network based on the pre-defined ratios (Image: 20%, Video: 80%). All models are trained with 32 A100 GPUs and each GPU holds either 6 image inputs or 2 video sequences.

2.2.2. REGEN

Our method is trained at the temporal compression ratio of $4\times$, $8\times$, $16\times$, $32\times$ on our dataset with over 100K iterations. We train our method under the setting of reconstruction, interpolation and extrapolation, and each scenario will be randomly sampled at every iteration based on pre-defined probabilities. Specifically, we tend more towards reconstruction in the early training stage, and progressively increase the probabilities of interpolation and extrapolation with more training iterations. Our spatiotemporal video encoder adopts the same architectural design as our implemented MAGVIT-v2 but encodes each chunk into 2 latent frames with varying compression rates. We train the spatiotemporal video encoder and the generative decoder in an end-to-end fashion with the AdamW optimizer. Similar with MAGVIT-v2 training, the model will be trained with mixed images and videos based on the pre-defined ratios (Image: 20%, Video: 80%). All models are trained using 32 A100 GPUs, but we leverage larger batch sizes compared to MAGVIT-v2 as we did not utilize 3D convolution layers in the decoder and the GPU can process more samples. The image batch size is set to be 28 for all experiments and the video batch size is 6, 4, 3, 1 for temporal compression ratio of $4\times$, $8\times$, $16\times$, $32\times$, respectively.

2.3. Architecture Details

We illustrate the architecture detail of our method at $4\times$ temporal compression rate in Fig.1. The spatiotemporal video encoder consists of four stages and it will encode every 5 raw frames into 2 latent frames with the spatial compression rate of $8\times$. The base channels are 128 and the channel multiplier for different stages is 1, 2, 4, 6, respectively. The generative decoder comprises 24 diffusion transformer blocks and takes the output of encoder as the conditioning signal. For higher compression rate experiments, we add an additional stage to the encoder while keeping the same number of diffusion transformer blocks in the de-

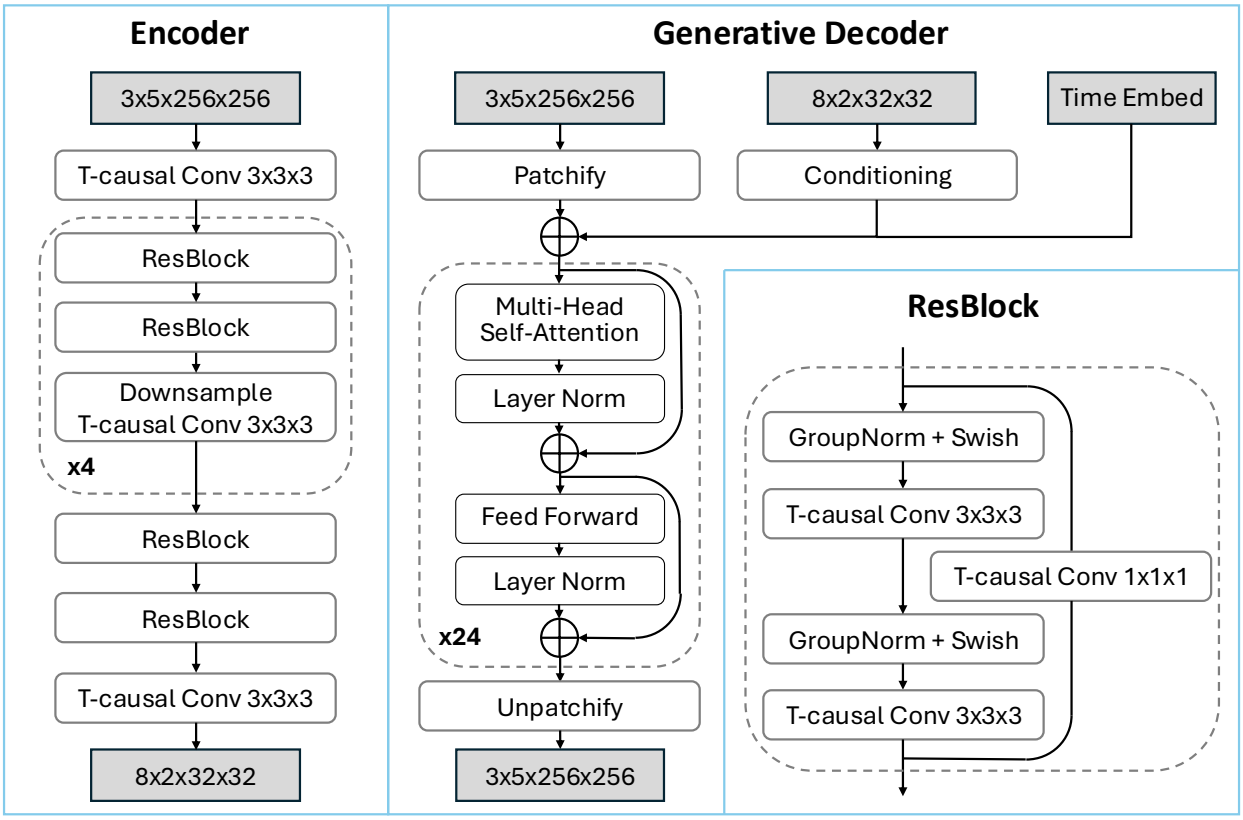


Figure 1. Architecture details of our method at $4\times$ temporal compression rate. T-causal Conv stands for temporal causal convolution where stride = $1 \times 1 \times 1$. Downsample T-causal Conv stands for temporal causal convolution where stride will be selected from $\{1 \times 1 \times 1, 1 \times 2 \times 2, 2 \times 1 \times 1, 2 \times 2 \times 2\}$, depending on the target compression ratios. The spatiotemporal video encoder is composed of 4 stages and the generative decoder is made up of 24 diffusion transformer blocks.

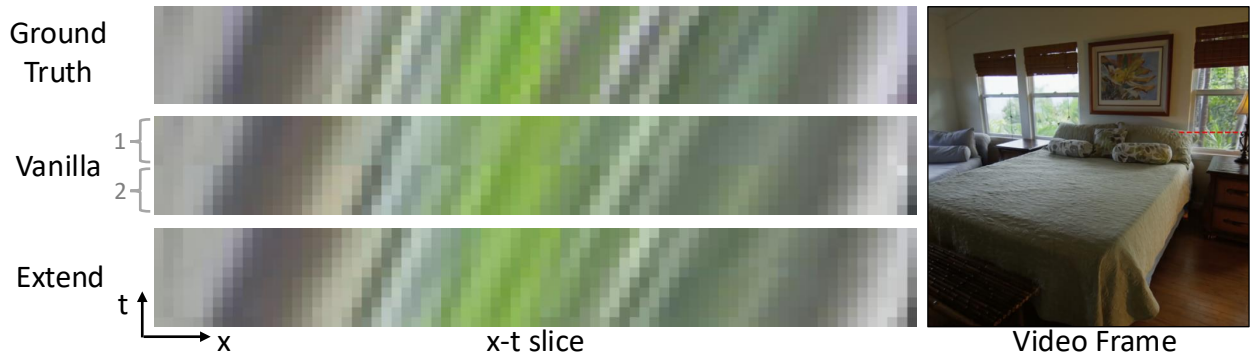
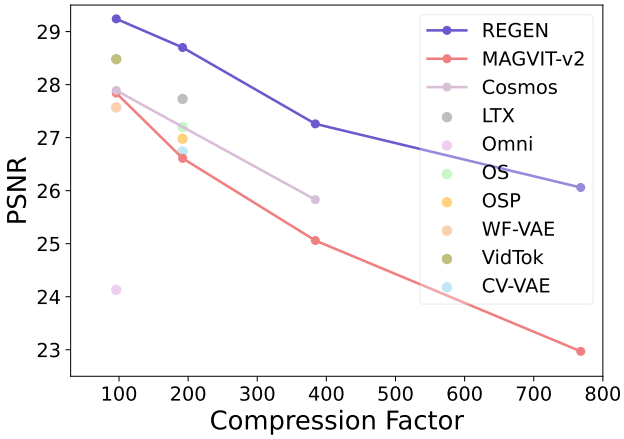


Figure 2. Alleviating the chunking issue with latent extension. The x-t slice is obtained by extracting a short segment (shown as the red line in the video frame) from 2 chunk frames.

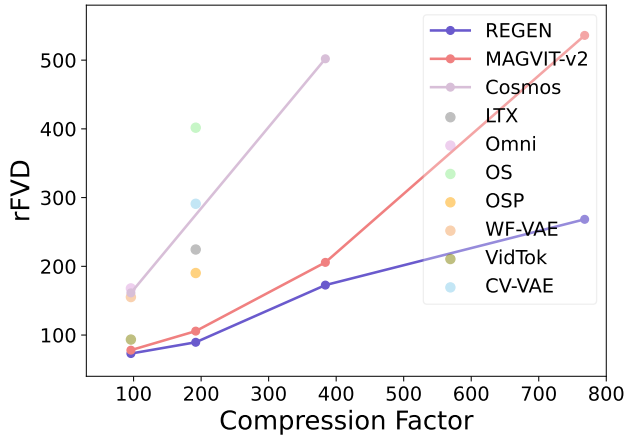
coder. The channel multiplier is adjusted to 1, 2, 4, 6, 6, accordingly.

2.4. Alleviating Chunking Issue with Latent Extension

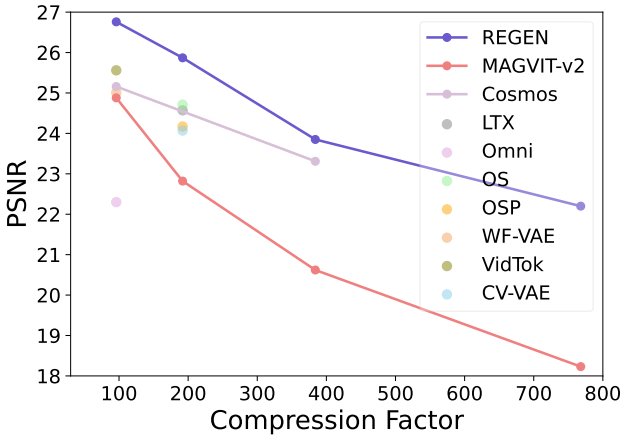
Following the idea of SDEdit [8], we utilize the last frame prediction from the previous chunk to guide the generation of the next chunk in an auto-regressive manner. Specifically, given a set of latent frames, we decode the first chunk in the reconstruction setting and preserve the intermediate results of the last frame at each sampling step. When decoding later chunks, we extend the latent by extrapolation and control the decoder to generate $T_c + 1$ frames with the time shift of -1, where T_c represents the length of each chunk. In this manner, we can ensure there is one overlapped frame between consecutive chunks. During the generation of later chunks, we leverage the intermediate results of the last frame from the previous chunk, which is also



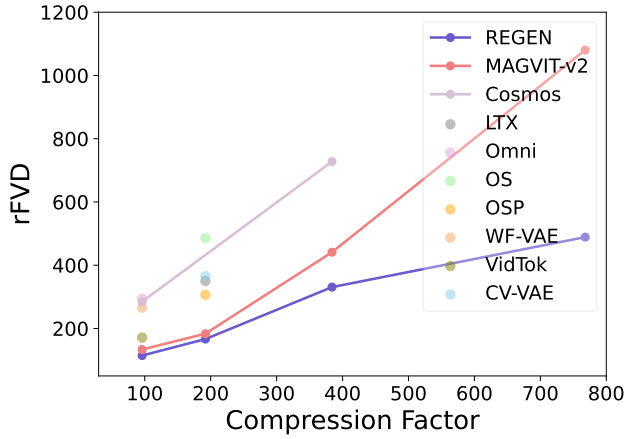
(a) PSNR comparisons with state-of-the-art video embedders on MCL-JCV dataset under 256×256 inputs.



(b) rFVD comparisons with state-of-the-art video embedders on MCL-JCV dataset under 256×256 inputs.



(c) PSNR comparisons with state-of-the-art video embedders on DAVIS 2019 dataset under 256×256 inputs.



(d) rFVD comparisons with state-of-the-art video embedders on DAVIS 2019 dataset under 256×256 inputs.

Figure 3. Holistic comparison with state-of-the-art video embedders with different latent channel dimensions. The compression factor is calculated by $r = \frac{C \times H \times W \times (T-1)}{c \times h \times w \times t}$. REGEN obtains better PSNR and rFVD compared to existing video embedders at various compression factors on different datasets.

the first frame of the current chunk, to guide the generation of the current chunk so that we could mitigate the jumping issue in an auto-regressive way. We provide an additional example in Fig. 2 to demonstrate the effect.

3. Additional Comparisons with SOTA Video Embedders

Existing video embedders have different latent channel dimensions. Some video embedders, like Open-SORA (OS) [18] and Open-SORA-Plan (OSP) [4] only have 4 latent channels, while video embedders like CogVideoX [14], HunyuanVideo [3], Wan [12] have 16 latent channels.

For fairness of comparison, we compare against these SOTA embedders by calculating the compression factor following LTX-Video [2]: $r = \frac{C \times H \times W \times (T-1)}{c \times h \times w \times t}$. Shown in Fig. 3, we further compare REGEN with Cosmos-Tokenizer [1], LTX-Video [2], OmniTokenizer [13], Open-SORA (OS) [18], Open-SORA-Plan (OSP) [4], WF-VAE [5], VidTok [11] and CV-VAE [17]. One can observe that our method exhibits better performance at various compression factors on both metrics and datasets.

To compare with video embedders which use 16-channel latents and have a smaller compression factor, we train a 16-channel variant of REGEN and the results can be found in Tab. 1: our method outperforms other SOTA video embedders on all metrics and datasets, demonstrating the effectiveness of our generative decoder and the soundness of our experimental setup.

4. Class-Conditional Video Generation

Following prior work Lattle [7], we perform class-conditional video generation on public benchmark UCF101 [10]. We train the Latte-L/2 model with MAGVIT-v2 and REGEN as the video embedder on the ultra-compact latent with 32× temporal compression. Both models are trained with 33 input frames at the resolution of 256×256 under

Method	MCL-JCV			DAVIS 2019		
	PSNR	SSIM	rFVD↓	PSNR	SSIM	rFVD↓
CV-VAE-SD3	33.15	0.861	22.62	30.23	0.803	40.21
CogVideoX	33.68	0.870	19.26	30.42	0.810	41.15
HunyuanVideo	<u>34.07</u>	<u>0.876</u>	<u>14.13</u>	<u>31.19</u>	<u>0.824</u>	<u>27.95</u>
Wan-2.1	33.50	0.867	19.85	30.45	0.809	35.32
REGEN	**35.55**	**0.898**	**10.34**	**32.93**	**0.860**	**23.34**

Table 1. **Reconstruction comparison at base 4× temporal compression.** We compare REGEN with various SOTA 16-channel video embedders at 4× temporal compression on MCL-JCV and DAVIS 2019 datasets under 512×512 inputs. The best results are bold-faced and the second best results are underlined.

the same resources for fair comparison. Tab. 2 shows that our method outperforms MAGVIT-v2 in FVD which validates that our strength in reconstruction can be translated into video generation.

Metric	Compression	Resolution	MAGVIT-v2	REGEN
FVD_{33}↓	8×8×32	256×256	708.36	**448.64**

Table 2. Quantitative evaluation of MAGVIT-v2 and REGEN for class-conditional video generation on UCF101 dataset under 256×256 inputs. The best results are bold-faced.

5. Efficiency Analysis of REGEN

For the training efficiency of latent generative models, we have the same efficiency with MAGVIT-v2 [15] as we leverage the same encoder with it and decoders will not affect the training of latent generative models.

Although operating on the noise input, REGEN is equipped with a large patch size to accommodate the costs and ensure efficient inference. Since REGEN supports one-step sampling, we measure the running time (milliseconds) of REGEN on one A100 GPU with one-step decoding and compare with MAGVIT-v2 at various compression rates under 256×256 inputs. One can observe from Tab. 3 that REGEN costs similar or fewer running times except at the compression rate of 8×8×32. The reason is that we have to adjust the location of upsampling layers in MAGVIT-v2 decoder at 8×8×32 and shift all upsampling layers towards the end blocks to avoid out of GPU memory during training due to the causal 3D convolutional layers, even on 80GB A100s. Note that we avoid doing this for other compression ratios of MAGVIT-v2 as it is not an optimal design and lowers reconstruction quality, as also mentioned in MAGVIT-v2 work [15]. For 32× temporal compression variant, this design choice is unavoidable to ensure the decoding process fits within the available GPU memory. Consequently, the computational complexity of this model is lower than that of the ideal design. Even so, this version of 32× MAGVIT-v2 cannot decode videos of even 512x512 resolution which highlights the scalability of our transformer-based decoder.

Method	Latency (ms)			
	8×8×4	8×8×8	8×8×16	8×8×32
MAGVIT-v2	88	317	343	153*
REGEN	89	159	295	548

Table 3. Comparison of decoder latency at various compression rates on one A100 GPU under 256×256 inputs. *MAGVIT-v2 32× has small latency because we have a different decoder design where we move all upsampling blocks toward the end layers. This is not optimal from the perspective of reconstruction quality for the decoder, but this design is unavoidable otherwise the model gives Out Of GPU Memory on 80GB A100 GPUs.

6. Effectiveness of Scaling Convolution-Based Video Embedders

While we have shown that REGEN exhibits significant advantage over MAGVIT-v2 at high temporal compression rates, e.g., 16× and 32×, we further scale up the MAGVIT-v2 decoder to explore whether larger model size could help to mitigate the performance gap. Specifically, we scale up the MAGVIT-v2 decoder through the width dimension until it matches the parameter of REGEN or reaches the GPU memory. Tab. 4 shows that simply scaling up the model size do not always translate to better results at high compression rates, e.g., the expanded MAGVIT-v2 has worse SSIM and rFVD compared to the original version at 16× compression. Although scaling up model size has shown some improvement at 32× compression, it still lags behind REGEN obviously, suggesting that the idea of using generative decoders to escape the compression-reconstruction trade-off is effective.

Method	8×8×16			8×8×32		
	PSNR	SSIM	rFVD↓	PSNR	SSIM	rFVD↓
MAGVIT-v2	20.62	0.527	441.24	18.23	0.419	1080.15
MAGVIT-v2◊	20.83	0.508	486.48	18.98	0.437	1020.80
REGEN	**23.85**	**0.635**	**328.83**	**22.20**	**0.575**	**488.89**

Table 4. Comparisons of expanded MAGVIT-v2 with REGEN on DAVIS 2019 dataset under 256×256 inputs. We expand the MAGVIT-v2 decoder by scaling up the width dimension and ◊ denotes the expanded version. The best results are bold-faced.

7. Latent Interpolation and Extrapolation

As described in the Method, our INR-based latent conditioning module not only supports reconstruction, but can generalize to interpolation and extrapolation as well with a uniform design. To examine the interpolation ability of our method, we compare it with two baselines: (1) Frame Averaging: it averages the ground truth to get the interpolated

frames; (2) Ours + External Interpolation: it applies off-the-shelf interpolation model [16] on our reconstructed frames. Fig. 4 shows that simply averaging the frames results in clear artifacts on the interpolated frames, while the results of our model and EMA display a smoother transition and align well with the ground truth. Apart from interpolation, our design supports extrapolation as well, where the model is going to predict the previous or future frames based on the given input. Shown in Fig. 5, our approach forecasts future motion based on prior frame sequences and the results demonstrate strong alignment with the ground truth, which highlights the generation ability of our decoder. Note that our method is able to predict the past frames as well which is shown in Fig. 6, promising the application of our method for chunk-free generation to mitigate the jumping issue.

8. Relation to Stable Cascade and LTX-Video

Stable Cascade [9] performs in the latent domain which requires an additional decoder for decoding, while REGEN leverages a DiT decoder operating in the pixel space and targets learning temporally compact latent space in videos. While sharing the goal of generating details from a compact latent, LTX-Video [2] is a noise-conditioned decoder (retaining the traditional latent-to-video mapping) rather than a full diffusion model, and could only be applied at the last denoising step in inference. Our decoder is a standard diffusion model which starts from pure noise during inference. Moreover, LTX-Video relies on a 128-dim latent for quality whereas we restrict to only 8 channels.

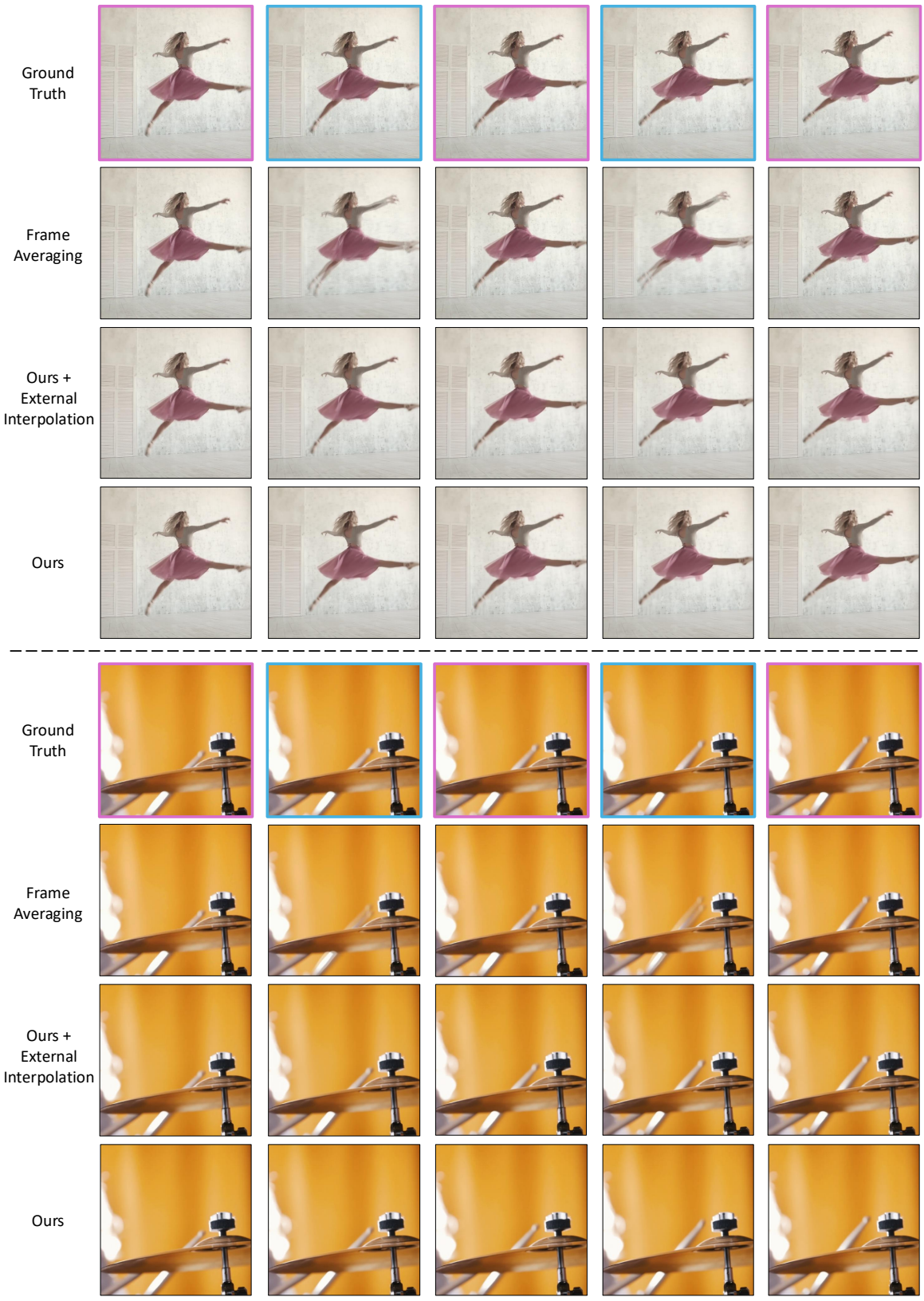


Figure 4. 2× interpolation results. Given input frames with purple bounding boxes, the model is asked to conduct interpolation to predict the frame with blue bounding box and we compare our method with frame averaging and external interpolation model.

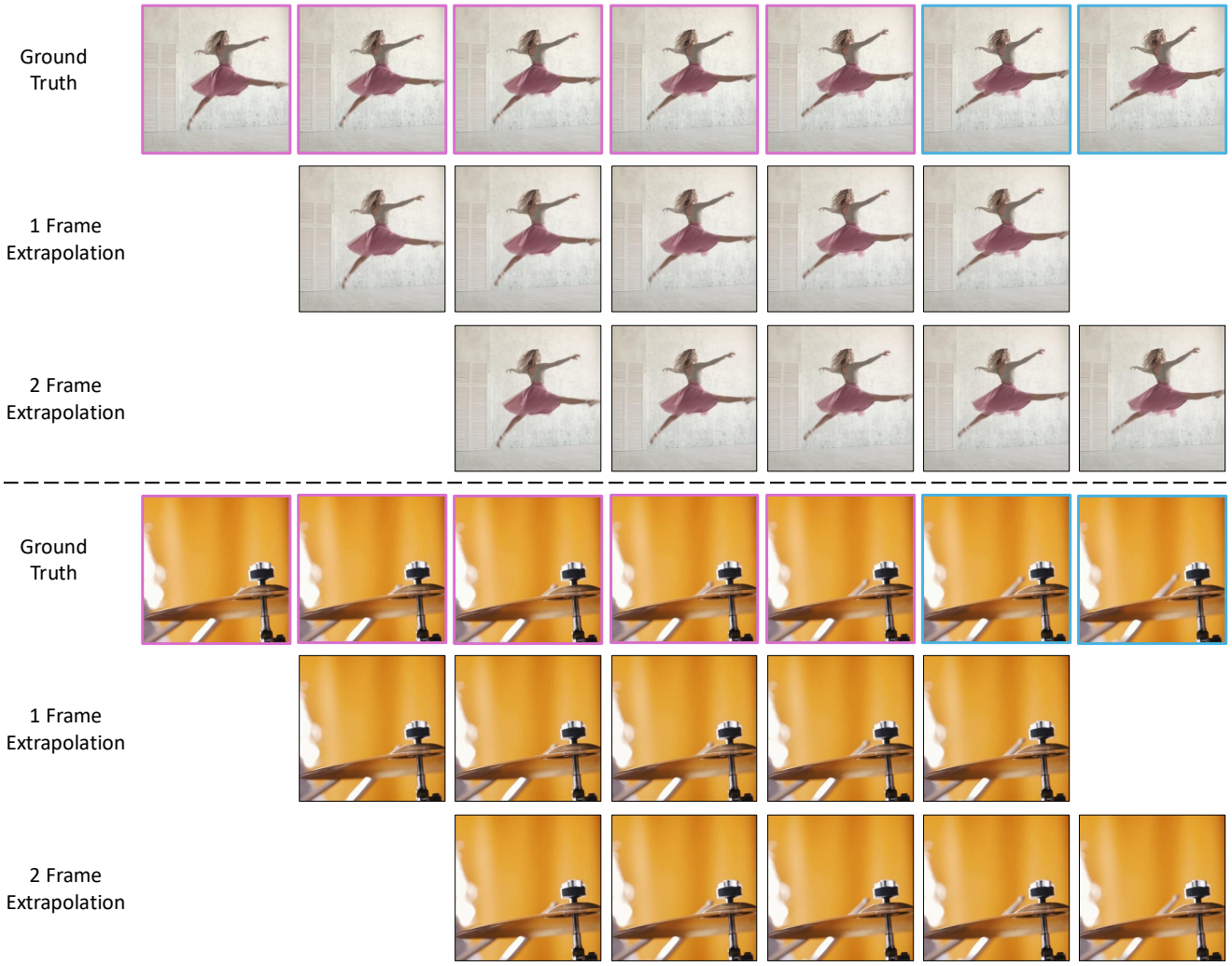


Figure 5. Forward latent extrapolation results. Given input frames with purple bounding boxes, the model is asked to conduct extrapolation to predict the future frame with blue bounding box.

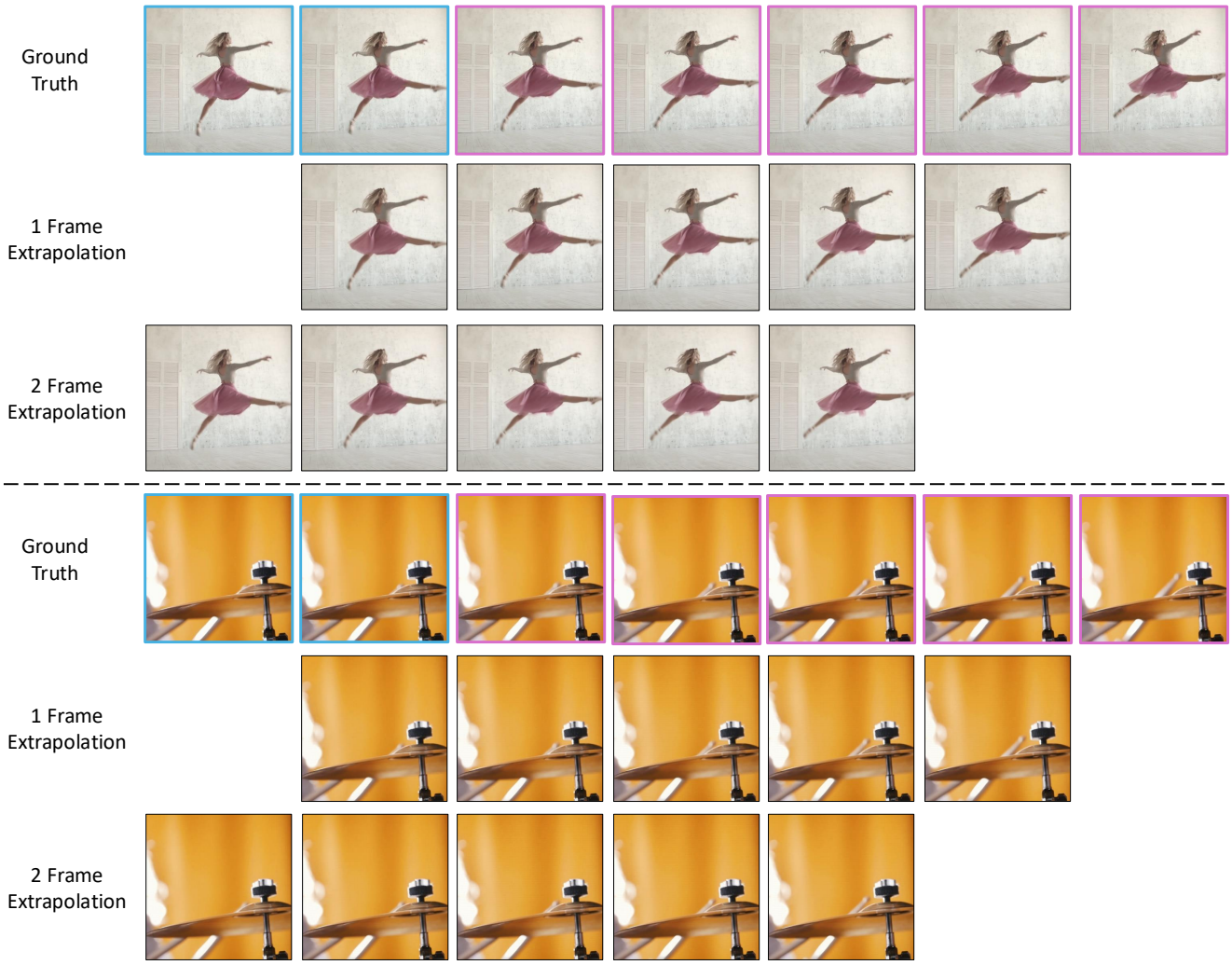


Figure 6. Backward latent extrapolation results. Given input frames with purple bounding boxes, the model is asked to conduct extrapolation to predict the past frame with blue bounding box.

References

[1] Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, et al. Cosmos world foundation model platform for physical ai. *arXiv preprint arXiv:2501.03575*, 2025. 3

[2] Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, et al. Ltx-video: Realtime video latent diffusion. *arXiv preprint arXiv:2501.00103*, 2024. 3, 5

[3] Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024. 3

[4] PKU-Yuan Lab and Tuzhan AI etc. Open-sora-plan, 2024. 3

[5] Zongjian Li, Bin Lin, Yang Ye, Liuhan Chen, Xinhua Cheng, Shenghai Yuan, and Li Yuan. Wf-vae: Enhancing video vae by wavelet-driven energy flow for latent video diffusion model. *arXiv preprint arXiv:2411.17459*, 2024. 3

[6] I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 1

[7] Xin Ma, Yaohui Wang, Gengyun Jia, Xinyuan Chen, Ziwei Liu, Yuan-Fang Li, Cunjian Chen, and Yu Qiao. Latte: Latent diffusion transformer for video generation. *arXiv preprint arXiv:2401.03048*, 2024. 3

[8] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021. 2

[9] Pablo Pernias, Dominic Rampas, Mats L Richter, Christopher J Pal, and Marc Aubreville. Würstchen: An efficient architecture for large-scale text-to-image diffusion models. *arXiv preprint arXiv:2306.00637*, 2023. 5

[10] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 3

[11] Anni Tang, Tianyu He, Junliang Guo, Xinle Cheng, Li Song, and Jiang Bian. Vidtok: A versatile and open-source video tokenizer. *arXiv preprint arXiv:2412.13061*, 2024. 3

[12] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. 3

[13] Junke Wang, Yi Jiang, Zehuan Yuan, Bingyue Peng, Zuxuan Wu, and Yu-Gang Jiang. Omnitokenizer: A joint image-video tokenizer for visual generation. *NeurIPS*, 2025. 3

[14] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024. 3

[15] Lijun Yu, José Lezama, Nitesh B Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Vighnesh Birodkar, Agrim Gupta, Xiuye Gu, et al. Language model beats diffusion–tokenizer is key to visual generation. *arXiv preprint arXiv:2310.05737*, 2023. 1, 4

[16] Guozhen Zhang, Yuhan Zhu, Haonan Wang, Youxin Chen, Gangshan Wu, and Limin Wang. Extracting motion and appearance via inter-frame attention for efficient video frame interpolation. In *CVPR*, 2023. 5

[17] Sijie Zhao, Yong Zhang, Xiaodong Cun, Shaoshu Yang, Muyao Niu, Xiaoyu Li, Wenbo Hu, and Ying Shan. Cv-vae: A compatible video vae for latent generative video models. *NeurIPS*, 2025. 3

[18] Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. Open-sora: Democratizing efficient video production for all, 2024. 3