

LoViC: Efficient Long Video Generation with Context Compression

Supplementary Material

In this material, we present inference time comparison with baselines, analysis of time complexity, limitations, more implementation details and a PPT with demo videos.

Comparison of Inference Time. Table 5 shows the inference time comparison between baselines and our model. We follow the baselines’ default configs. CausVid is the fastest among all methods. It distills the Wan text-to-video model to four time steps with DMD, and applies a block-autoregressive generation process. SkyReel-V2 applies a generation process similar to CausVid, but without distillation acceleration. FramePack is the slowest due to the biggest model size. For LTX-Video and our model, we generate the 771 frames in three segments. This comparison is not completely fair, as the models have different model sizes and video VAE compression ratios. The block-autoregressive models with small context lengths, *e.g.* SkyReel-V2, should have faster speed, but LTX-Video and our model are actual faster due to the high compression rate of the pretrained VAE. Comparing to the video extension feature of LTX-Video-V0.9.6, our model is faster because of the context compression design.

Method	Model Size	Context Length	Inference Time
CausVid	1.4B	16	17.2 sec
SkyReel-V2	1.4B	17	283.6 sec
FramePack	13B	19	619.9 sec
LTX-Video-V0.9.6	1.9B	*	115.2 sec
Ours	2.3B	*	86.0 sec

Table 5. **Comparison of inference time.** The model size does not take into account VAE and text encoder. Context length is measured by maximum frame number of conditioning video, and * represents unfixed context length. Inference time is measured by generating a total of 771 frames at 360P resolution.

Analysis of Time Complexity. We have demonstrated that our method enables the generation of longer videos than the vanilla DiT under the same memory constraint, in the main paper. Here we provide a complexity analysis showing that our architecture can also achieve faster inference when generating a video of the same length. When generating a video consisting of T tokens in total, we divide the video into n segments, each containing $T_0 = \frac{T}{n}$ tokens. Let $0 < \alpha < 1$ denote the compression ratio of the FlexFormer encoder. Given that the computational cost of the FlexFormer is negligible compared to the DiT, we omit it

in the analysis. The time complexity for generating all n segments with our architecture is:

$$\mathcal{O}\left(\sum_{i=1}^n (T_0 + \alpha(i-1)T_0)^2\right) \quad (9)$$

$$= \mathcal{O}\left(T_0^2 \sum_{i=1}^n (1 + \alpha(i-1))^2\right) \quad (10)$$

$$= \mathcal{O}\left(\frac{T^2}{n^2} \sum_{i=1}^n (\alpha^2 + 2\alpha(i-1) + 1)\right) \quad (11)$$

$$= \mathcal{O}\left(\frac{T^2}{n^2} \left(\alpha^2 \cdot \frac{(n-1)n(2n-1)}{6} + 2\alpha \cdot \frac{n(n-1)}{2} + n\right)\right) \quad (12)$$

$$= \mathcal{O}\left(\frac{\alpha^2 n}{3} T^2\right) \quad (13)$$

Since the factor $\frac{\alpha^2 n}{3}$ can be a small constant in practice, our model can achieve faster inference than vanilla DiT, given a proper implementation.

More Implementation Details. We train the model in three stages. In the first stage, we train the lightweight FlexFormer autoencoder from scratch for 20K steps in a batch size of 128, with MSE loss as shown in Equation 8. In the second stage, we freeze the FlexFormer and train the DiT for 30K steps, also using a batch size of 128. For each mini-batch, we randomly sample one of the tasks: prediction, interpolation or retrodiction. In the third stage, we train the model on multi-shot data for 10K steps. A temporal gap of 20 latent frames is empirically chosen to enable multi-shot capabilities. The results in Table 1 are obtained using a linearly-scaled compression strategy.

Limitations Due to resource constraint, we only collected 360P video and trained our model at that resolution, which hinders the model from generating videos of larger resolution. And each set of videos only contains two or three clips. Besides, there is still a gap between our pseudo multi-shot videos collected by filtering the random segments of long videos and the real multi-shot videos collected by previous work [24, 77], which is reflected by the naturalness of the generated videos. A more sophisticated video captioning strategy proposed by LCT [24] which labels each subject with a distinct ID may also help to improve the subject ID consistency of the multi-shot videos.