A. Extended Technical Specification

In this part, we provide some extended technical details of the proposed methods.

A.1. Specification on T2V Latent Diffusion Model

First, we can formalize the T2V task as generating an video X=$\{x_1, \cdots, x_F\} \in \mathbb{R}^{F\times H\times W\times C}$ given the input prompt text Y=$\{w_1, \cdots, w_S\}$. Here F, H, W, C are the frame length, height, width and channel number of the video respectively. We use the existing latent diffusion model (LDM) to accomplish the task. As shown in Figure 2, LDM consists of a diffusion (forward) process and a denoising (reverse) process in the video latent space. An encoder $\mathcal{E}$ maps the video frames into the lower-dimension latent space, i.e., $Z_0 = \{\mathcal{E}(X)\}$, and later a decoder $\mathcal{D}$ remaps the latent variable to the video, $X = \{\mathcal{D}(Z_0)\}$.

Diffusion Process. The diffusion process transforms the input video into noise. Given the compressed latent code Z_0, LDM gradually corrupts it into a pure Gaussian noise over T steps by increasingly adding noisy, i.e., Markov chain. The noised latent variable at step $t \sim [1, T]$ can be written as:

$$Z_t = \sqrt{\hat{\alpha}_t}x + \sqrt{1-\hat{\alpha}_t}\epsilon_t\,, \tag{8}$$

with

$$\hat{\alpha}_t = \sum_{k=1}^{t} \alpha_t\,, \quad \epsilon_t \sim \mathcal{N}(0, I)\,, \tag{9}$$

where $\alpha_t \in (0, 1)$ is a corresponding diffusion coefficiency. The diffusion process can be simplified as: $q(Z_{1:T}|Z_0) = \prod_{t=1}^{T} q(Z_t|Z_{t-1})$.

Denoising Process. The denoising process then restores the noise into the video reversely. The learned reverse process $p_\theta(Z_{0:T}) = p(Z_T)\prod_{t=1}^{T} p_\theta(Z_{t-1}|Z_t, Y)$ gradually reduces the noise towards the data distribution conditioned on the text Y. Our T2V LDM is trained on video-text pairs (X, Y) to gradually estimate the noise ϵ added to the latent code given a noisy latent Z_t, timestep t, and conditioning text Y:

$$\mathcal{L}_{\text{LDM}} = \mathbb{E}_{Z\sim\mathcal{E}(X),Y,\epsilon,t}\left[\,||\epsilon - \epsilon_\theta(Z_t, t, \mathcal{C}(Y))||^2\,\right]\,, \tag{10}$$

where $\mathcal{C}(Y)$ denotes a text encoder that models the conditional text, and the denoising network $\varepsilon_\theta(\cdot)$ is often implemented via a 3D-UNet.

The original 3D-UNet [23] has a spatial-temporal feature modeling. The practical implementation is to use a 3D convolution along the spatial dimension, and followed by a temporal attention. Specifically, given a set of F video frames, we apply a shared 3D convolution for all the frames to extract the spatial features. After that, we assign a set of distribution adjustment parameters to adjust the mean and variance for the intermediate features of every single frame via:

$$Z_t^i = \boldsymbol{W}^i\text{Conv3D}(Z_t^i) + b^i\,, \tag{11}$$

where the convolutions are performed over 3D patches Z_t^i. Then, with a set of given video frame features, $Z_t \in \mathbb{R}^{F\times H\times W\times C}$, we apply the temporal attention to the spatial location across different frames to model their dynamics. Specifically, we first reshape Z_t into shape of $HW \times \#Heads \times F \times \frac{C}{\#Heads}$. We then obtain their query Q_t, key K_t, and value V_t embeddings used in the self-attention via three linear transformations. We calculate the temporal attention matrix H_t via:

$$H_t = \text{Softmax}(\frac{Q_t \cdot K_t}{\sqrt{d}}) \cdot M\,, \tag{12}$$

where M is an lower triangular matric with $M_{i,j} = 0$ if $i > j$ else 1. With the implementation of the mask, the present token is only affected by the previous tokens and independent from the future tokens since the frames are arranged based on their temporal sequence.

A.2. Specification on Dynamic Scene Graph Representation

DSG [26] is a list of single visual SG [27] of each video frame, organized in time-sequential order. We can denote an DSG as G=$\{G_1, \cdots, G_M\}$, with each single SG (G_m) corresponding to the frame (x_m). An SG contains three types of nodes, i.e., *object*, *attribute*, and *relation*, in which some scene objects are connected in certain relations, forming the spatially semantic triplets '*subject-predicate-object*'. Also, objects are directly linked with the attribute nodes as the modifiers. Figure 10 illustrates the visual SG.

Since a video comes with inherent continuity of actions, the SG structure in DSG is always temporal-consistent across frames. This characterizes DSGs with spatial&temporal modeling. Figure 11 visualizes a DSG of a video.

A.3. Prompting ChatGPT with In-context Learning

In §4.1 we introduce the Dysen module for action planning, event-to-DSG conversion, and scene enrichment, during which we use the in-context learning (ICL) to elicit knowledge from ChatGPT. Here we elaborate further on the specific designs, including 1) the ICL for action planning, and 2) two ICLs for scene enrichment, containing the step-wise scene imagination and global scene polishment.

For each task, we write the prompts, including 1) a job description (`Instruction`), 2) a few input-output in-context examples (`Demonstration`), and 3) the desired testing instance (`Test`). By feeding the ICL prompts into ChatGPT, we expect to obtain the desired outputs in the demonstrated format. Note that we pre-explored many other different job instruction prompts, and the current (shown in

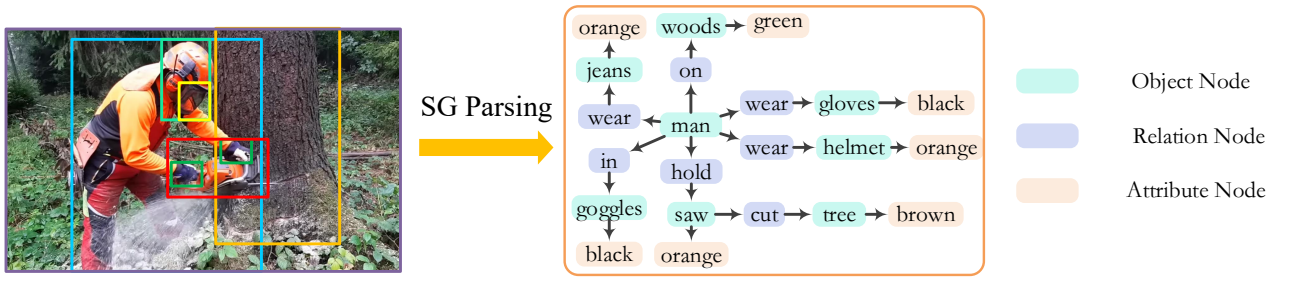

Figure 10. Illustration of a visual SG.

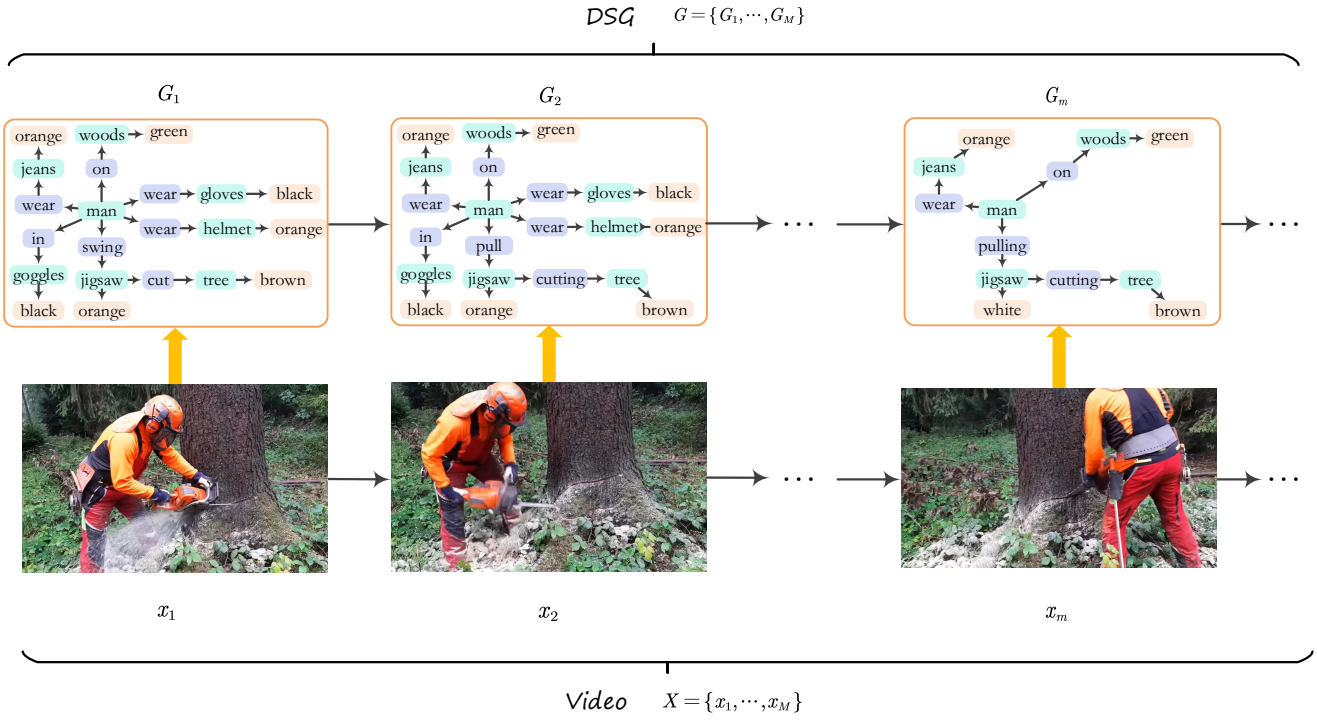

Figure 11. DSG is a list of single visual SG in the temporal order.

this paper) version of instructions helps best elicit task outputs. Also, for each ICL, we select five examples as demonstrations, which can be enough to prompt the ChatGPT.

ICL Design for Action Planning. In Figure 12 we show the complete ICL illustration for action planning. Note that the demonstration examples should come with the 'gold' annotations, so as to correctly guide the ChatGPT to induce high-quality output. In our implementation, for the action planning, we first obtain the majorly-occurred events (i.e., actions) from the input texts via ChatGPT. Note that this can be a very easy task within the natural language processing area, and we simply treat this as the gold annotations. Then, we use an off-the-shelf best-performing video moment localization model [89] to parse the video starting and ending positions for each event expression. Via this, we obtain the action planning annotations for the demonstrations.

ICL Design for Step-wise Scene Imagination. In Figure 13 we show the full illustration of the ICL for step-wise scene imagination. We note that the output triplets after imagination are the full-scale triplets, including the raw ones of the unenriched SG. This means, we will overwrite the raw SG with the output triplets, which cover both the *add* and *change* operations.

ICL Design for Global Scene Polishment. In Figure 14 we show the full illustration of the ICL for global scene polishment. Same to the ICL for step-wise scene imagination, we also take the output DSG as the refined DSG. The 'gold' annotations for demonstrations of the scene imagina-

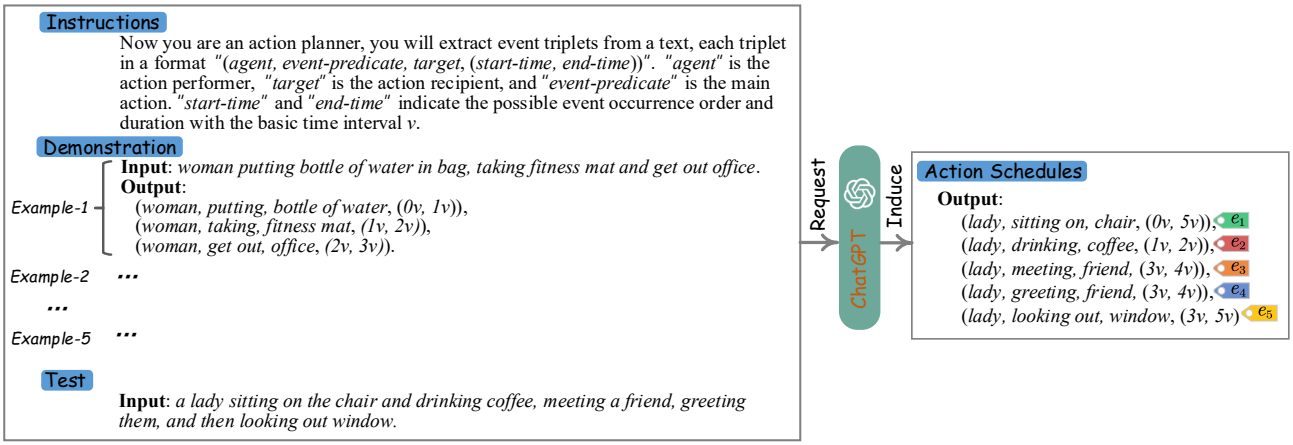

Figure 12. Illustration of the ICL for action planning.

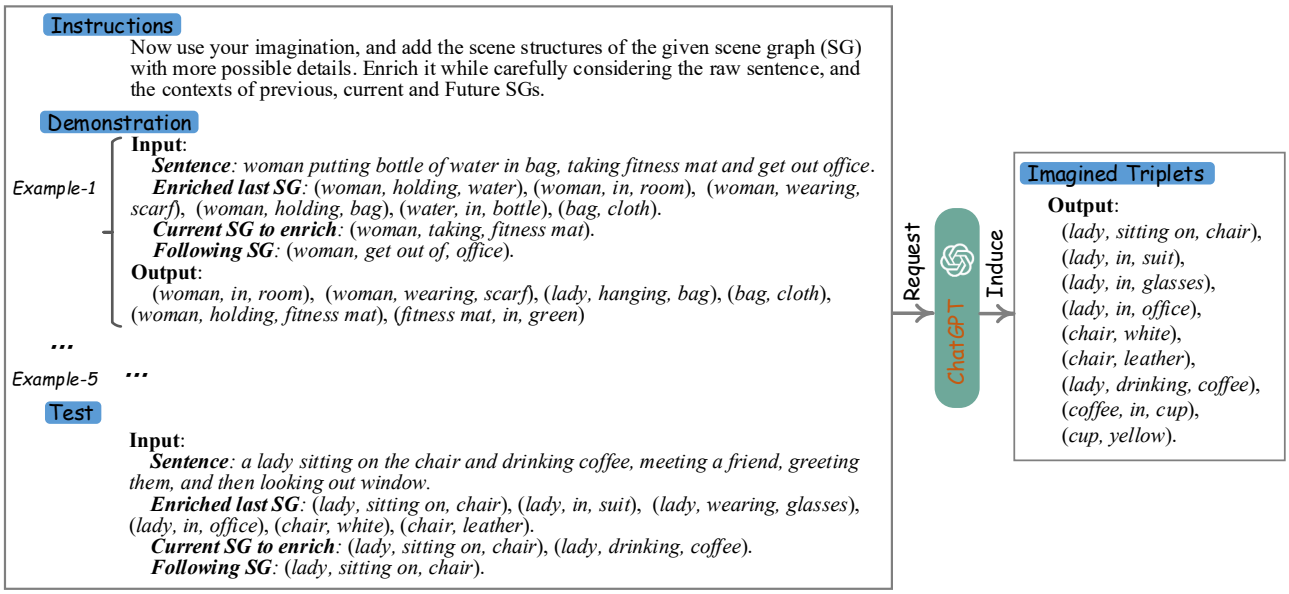

Figure 13. Illustration of the ICL for step-wise scene imagination.

tion are constructed by parsing the video via external DSG parser [26]. We employ the FasterRCNN [50] as an object detector to obtain all the object nodes, and use MOTIFS [59] as a relation classifier to obtain the relation labels (nodes) as well as the relational edges. We then use an attribute classifier to obtain attribute nodes. We filter the object, attribute, and relation annotations by keeping those that appear more than 500 times in the training set. This helps screen the less-informative noises.

B. Experiment Specification

In this part we extend the description of the experimental settings.

B.1. Evaluation

In our experiments, we used the following three types of evaluation metrics.

Automatic Metrics. Following previous work [2, 5, 18], we use the Inception Score (IS) and Fréchet Video Distance (FVD) for UCF-101, and Fréchet Image Distance (FID) and CLIP similarity (CLIPSIM) for MSR-VTT.

- **IS** [54] evaluates the distribution of the frame images of our generated videos.[3] Following previous work on video synthesis, we used a C3D [63] model trained on UCF-101 to calculate a video version of the inception

[3] https://torchmetrics.readthedocs.io/en/stable/image/inception_score.html

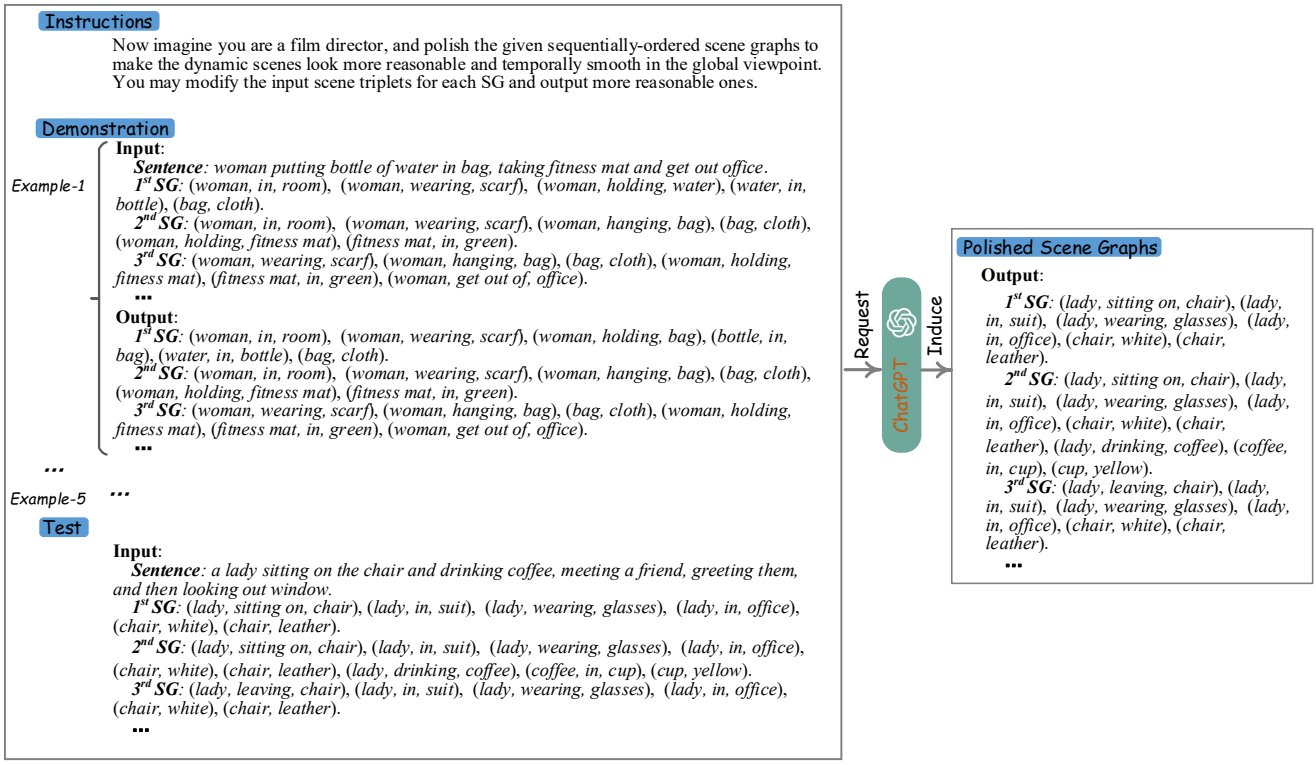

Figure 14. Illustration of the ICL for global scene polishment.

score, which is calculated from 10k samples using the official code of TGANv2.[4]

- **FVD** measures the similarity between real and generated videos [64]. For the generated videos (16 frames at 30 FPS), we extract features from a pre-trained I3D action classification model.[5]
- **FID** [20] measures the Fréchet Distance between the distribution of the frames between synthetic and gold videos in the feature space of a pre-trained Inception v3 network. Practically, we employ *pytorch-fid*[6] to calculate the FID score.
- **CLIPSIM** [19] is also used for the quantitative analysis of the semantic correctness of the text-to-video generation on MSR-VTT data. We take into account the reference-free scores via CLIP [48]. In this paper, we use the officially released code[7] to calculate the CLIP score. We generate 2,990 videos (16 frames at 30 FPS) by using one random prompt per example. We then average the CLIPSIM score of the 47,840 frames. We use the ViT-B/32 [48] model to compute the CLIP score.

Human Evaluation Criterion. In §5.4 we also adopt the human evaluation, i.e., user study, for more intuitive assessments of video quality. On the ActivityNet test set, we compare our model with baseline systems. We randomly select 50 text-video pairs (videos containing both the gold-standard ones and the generated ones), and ask ten participants (native English speakers) who have been trained with rating guidelines, to rate a generated video. Specifically, we design a Likert 10-scale metric to measure the target aspect: 1-Can't be worse, 2-Terrible, 3-Poor, 4-Little poor, 5-Average, 6-Better than average, 7-Adequate, 8-Good, 9-Very good, 10-Excellent. For each result, we take the average. We mainly measure the quality of videos in terms of *action faithfulness*, *scene richness* and *movement fluency*, each of which is defined as:

- **Action faithfulness**: Do the visual actions played in the video coincide with the raw instruction of the input texts? Is there any point missed or incorrectly generated?
- **Scene richness**: Are the visual scenes rich? Are there vivid and enough background or foreground details in the video frames?
- **Movement fluency**: Are the video dynamics of actions fluent? Is the video footage smooth? Are the behaviors presented in a continuous and seamless manner?

[4] https://github.com/pfnet-research/tgan2
[5] https://www.dropbox.com/s/ge9e5ujwgetktms/i3d_torchscript.pt?dl=1
[6] https://github.com/mseitzer/pytorch-fid
[7] https://github.com/jmhessel/clipscore

Triplet Recall Rate. In Figure 7 we use the *Triplet Recall* (TriRec.) to measure the fine-grained '*subject-predicate-object*' structure recall rate between the SGs of input texts and video frames. Technically, TriRec. measures the percentage of the correct relation triplet among all the relation triplets between two given SGs. Given a set of ground truth triplets (*subject-relation-object*), denoted G^{GT}, and the TriRec. is computed as:

$$\text{TriRec.} = \frac{|G^{PT} \cap G^{GT}|}{|G^{GT}|}, \tag{13}$$

where G^{PT} are the relation triplets of the SG in the generated video DSG by a visual SG parser.

B.2. Baseline Specification

We mainly compare with the currently strong-performing T2V systems as our baselines, which are divided into two groups: non-diffusion-based T2V, and diffusion-based T2V.

- **Non-diffusion-based T2V Methods**
  - **VideoGPT** [82] is a two-stage model: it encodes videos as a sequence of discrete latent vectors using VQ-VAE and learns the autoregressive model with these sequences via Transformer.
  - **TGANv2** [53] is a computation-efficient video GAN based on designing submodules for a generator and a discriminator.
  - **DIGAN** [86] is a video GAN which exploits the concept of implicit neural representations and computation-efficient discriminators.
  - **MoCoGAN-HD** [61] uses a strong image generator for high-resolution image synthesis. The model generates videos by modeling trajectories in the latent space of the generator.
  - **TATS** [11] is a new VQGAN for videos and trains an autoregressive model to learn the latent distribution.
  - **CogVideo** [24] is a large-scale pre-trained text-to-video generative model based on Transformer with dual-channel attention.
  - **InternVid** [73] is a video-text representation learning model based on ViT-L via contrastive learning.
- **Diffusion-based T2V Methods**
  - **VDM** [23] extends the image diffusion models for video generation by integrating a 3D-UNet architecture based on 3D convolutional layers.
  - **LVDM** [18] is built upon the latent diffusion models with a hierarchical diffusion process in the latent space for generating longer videos.
  - **MakeVideo** [55] directly translates the tremendous recent progress in Text-to-Image (T2I) generation to T2V without training T2V from scratch.
  - **MagicVideo** [91] is a latent diffusion based T2V model, which takes 2D convolution + adaptor block operation and a directed self-attention module for the spatial-temopral learning.
  - **AlignLatent** [5] is also a latent diffusion based T2V model, which leverages the pre-trained image DMs for video generators by inserting temporal layers that learn to align images in a temporally consistent manner.
  - **ED-T2V** [37] is an efficient training framework for diffusion-based T2V generation, which is built on a pretrained text-to-image generation model.
  - **VideoGen** [33] is a cascaded latent diffusion module conditioned on both the reference image and the text prompt, for generating latent video representations, followed by a flow-based temporal upsampling step to improve the temporal resolution.
  - **VideoFactory** [70] strengthens the interaction between spatial and temporal perceptions by utilizing a swapped cross-attention mechanism in 3D windows that alternates the "query" role between spatial and temporal blocks, enabling mutual reinforcement for each other.
  - **Latent-VDM**: we implement a T2V baseline of latent video diffusion model based on the latent diffusion [52], with the widely-adopted spatial convolution and temporal attention.
  - **Latent-Shift** [2] adds a parameter-free temporal shift module onto the existing latent video diffusion model to enhance the motion dynamics learning of video generation.

To enable further customized evaluations and experiments, we also re-implement some open-sourced baselines, including CogVideo[8] [24], VDM[9] [23] and Latent-VDM[10] [52].

C. Extended Experiments

C.1. Visualization of DSG-guided Controllable Video Synthesis

In Figure 15 we show the video frames generated by our Dysen-VDM, along with which we visualize the DSGs induced and enriched by the Dysen module.

[8] https://github.com/THUDM/CogVideo

[9] https://github.com/lucidrains/video-diffusion-pytorch

[10] https://github.com/nateraw/stable-diffusion-videos

Figure 15. Visualization of generated videos with the induced and enriched DSG structures.

C.2. Failure Analysis

While Dysen-VDM helps achieve overall improved performance in most of cases, it will err sometimes. Here we summarize the typical failure cases of Dysen-VDM that were made during our experiments.

- **Type-1:** Due to the limitations of LLM, sometimes it may hallucinate, leading to errors in scene understanding. The imagined DSG quality is relatively low, which, in turn, affects the quality of the generated video.
- **Type-2:** DSG is very proficient at generating realistic videos. However, there are some abstract scenes, such as cartoon-style videos, that cannot be supported by the structured triplet representations of SG. DSG struggles to effectively improve specific artistic styles in certain frames.

But fortunately, in most of the T2V scenarios, Dysen-VDM can advance. The integration of structured DSG representations with rich details (from LLMs' imagination) empowers the system with highly controllable content generation and high-quality video dynamics.