Improving Controllability of Text-to-Video Generation Through Image Editing and Interpolation

Jason Boxi Zhang
Stanford University
jasonbz@stanford.edu

Maty Bohacek
Stanford University
maty@stanford.edu

Abstract

Most existing video generation methods lack the ability to edit an existing video. While early work on video editing exists, current approaches are generally unstable and struggle to fully conform to desired edits while preserving high fidelity to the original video. To address these shortcomings, this paper proposes Smooth Prompted Interpolation and Creative Editing (SPICE), a new text-to-video generation method designed for intuitive, language-based editing. SPICE combines an image generation, interpolation, and editing backbone, orchestrated by a large language model (LLM), and achieves near-state-of-the-art performance on VBench compared to leading video generation methods, while enabling intuitive editing.

1. Introduction

A year ago, AI-based video generation primarily produced short, uncanny clips with unrealistic motion. However, recent text-to-video models—including OpenAI's Sora [26], Google's Veo 2 [12], and Luma's Dream Machine [1]—have achieved unprecedented levels of photorealism, coherence, and extended generation time. Despite these advancements, a major shortcoming remains: limited adherence to the text prompt [43] and the inability to edit an already-generated clip without starting the synthesis from scratch [19].

The challenge of video editing in current architectures stems primarily from three key issues: (1) temporal consistency problems [25]; (2) tight spatial-temporal coupling in internal representations, making it difficult to disentangle and modify specific aspects of a video in a fine-grained manner [3, 22]; and (3) inconsistent latent spaces, leading to lossy inversion of generated videos [19, 25].

To counter these challenges, we draw inspiration from advancements in different parts of computer vision outside of direct text-to-video synthesis. In particular, still image editing has significantly improved in fine-grained inversion and preserving the consistency of non-edited elements, tackling challenge #3. Image interpolation, meanwhile, has largely resolved its temporal inconsistency issues, addressing challenge #1.

To leverage these recent breakthroughs for text-to-video generation, we propose an iterative framework: starting from the n-th keyframe in a generated video, an image editing backbone modifies it to create the $(n+1)$-th keyframe, while an image interpolation backbone fills in the intermediate frames.

Since the image editing backbone is text-conditioned, each keyframe must be paired with a text description of the applied changes relative to the previous keyframe. A large language model (LLM) orchestrates this decoupling, imposing a directly interpretable latent representation in both language and RGB space, effectively mitigating challenge #2. This structured representation enables fine-grained, text-driven editing. We call this method Smooth Prompted Interpolation and Creative Editing (SPICE).

The rest of this paper is structured as follows: we first briefly review related work in video generation and adjacent tasks. Next, we detail our proposed method and experimental setup. Finally, we present the attained results, discuss their significance and limitations, and conclude with considerations for future work.

2. Related Work

This section reviews existing literature on video generation, video editing, image generation, image editing, and image interpolation. We begin with image-based methods, as video methods largely build upon them.

2.1. Image Generation

Moving away from Generative Adversarial Networks (GANs), which were once the dominant image generation architecture [24], most recent text-to-image models are based on diffusion [45]. In this paradigm, an image is iteratively denoised from a random noise sample while conditioned on text and, optionally, additional constraints such as style or pose [8]. Popular models include Stable Diffusion [33], DALL-E [31, 32], Imagen [34], and FLUX [23].

2.2. Image Editing

Recent image editing methods allow users to modify images using text prompts, direct dragging, or a combination of the two [21]. Many techniques invert the edited image into a latent space, apply modifications, and then map it back into RGB space [36], though each of these steps may degrade fidelity and adherence to the prompt. Some approaches train low-rank adaptations to mitigate these issues, but this incurs significant computational costs. Synthetic data is frequently used to train these models [5]. Notable methods include LEDITS++ [4], TurboEdit [44], ForgeEdit [47], and FeePromptEditing [15].

2.3. Image Editing and Language

Beyond vision-specific architectures, recent work has also explored utilizing multimodal LLMs for image editing [14]. These approaches can exploit the information learned by language models to closely follow detailed instructions and, conversely, better interpret ambiguous instructions [7, 41]. An example of this approach is ELLA [16], which combines LLMs with diffusion. Another example is ChatDiT [17], which utilizes multiple LLMs (designated as agents), which collaborate on the editing.

2.4. Image Interpolation

As deep learning architectures proved effective at solving computer vision tasks, early work explored the possibility to used their learned representations to interpolate between two images [10, 28]. While promising, these methods were often worse than conventional image interpolation that does not use any learned features. More recently, however, advanced embedding and generative models have been put to this task and found to be effective at interpolating, even outperforming conventional interpolation algorithms, especially on non-linear, complicated scenes [37]. An example of this is the DiffMorpher [46] architecture.

2.5. Video Generation

Text-to-video architectures have been extending the image generation methods by adding a temporal element to the synthesis. Make-A-Video [35] was among the first successful works to extend a text-to-image model to video generation, followed by ModelScopeT2V [38], FusionFrames [2], and TF-T2V [39] that enabled longer and more stable generation. By scaling these models in size and training dataset size, closed-source models, including OpenAI's Sora [26], Luma's Dream Machine [1], and Google's Veo 2 [12] have been successful at synthesizing near-photorealistic videos at larger resolutions.

2.6. Video Editing

While early work in video editing focused on specific domains such as talking head videos [13], generic video editing remains a relatively new task. Most methods follow a paradigm similar to image editing, where the video is first inverted into the model's latent space, modified, and then mapped back into RGB space [9]. However, these methods often struggle with spatial and temporal consistency [42]. In response, specialized architectures are now emerging for this task [11].

3. Methods

SPICE operates in two main modes: video generation and video editing. In video editing mode, SPICE assumes the video was previously generated by SPICE and retains access to its intermediate representations, enabling modifications without regenerating the entire video. SPICE relies of four backbones: an LLM backbone (B_{LLM}), an image generation backbone (B_{IG}), an image editing backbone (B_{IE}), and an image interpolation backbone (B_{II}).

3.1. Video Generation

An overview of the video generation mode is shown in Figure 1. At the input, the user provides a prompt P_{user}. SPICE then follows these steps:

1. B_{LLM} decomposes P_{user} into an initial keyframe prompt P_0 and a sequence of keyframe editing prompts P_i for $i \in \{1, \ldots, n\}$ (Section 3.1.1);
2. B_{IG} generates the initial keyframe $\hat{F}_0$ based on P_0 (Section 3.1.2);
3. For each subsequent keyframe i, B_{IE} modifies $\hat{F}i$ to produce $\hat{F}i + 1$ according to P_{i+1} (Section 3.1.3);
4. B_{II} interpolates intermediate frames $F_k^1, F_k^2, \ldots, F_k^m$ between each pair of consecutive keyframes $\hat{F}k$ and $\hat{F}k + 1$ (Section 3.1.4);
5. The generated frames are assembled into the final output video.

3.1.1 Prompt Decomposition and Orchestration

Our LLM pipeline is responsible for decomposing user inputted prompts for video generation into individual keyframe prompts. For this, we use a few-shot prompt [6] to instruct GPT-4o [27] to decompose the user inputted prompt into a longer and more detailed initial keyframe prompt, P_0 that sets up the scene, and then a list of shorter, more targeted sequential keyframe prompts $P_1, \ldots, P_n$ that describe isolated changes to be made between keyframe images. The decomposition system prompt is available for reference in Appendix F.

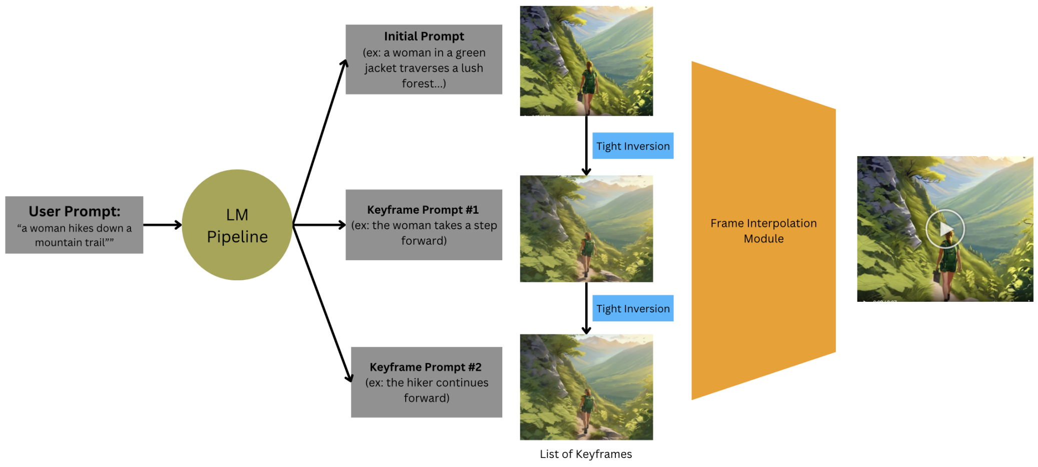

User Prompt:
"a woman hikes down a mountain trail""
LM Pipeline
Initial Prompt
(ex: a woman in a green jacket traverses a lush forest...)
Keyframe Prompt #1
(ex: the woman takes a step forward)
Keyframe Prompt #2
(ex: the hiker continues forward)
Tight Inversion
Tight Inversion
List of Keyframes
Frame Interpolation Module

Figure 1. Video Generation Pipeline: User prompts are processed by our LM Prompt Decomposition Module (3.1.1), generating keyframe prompts. Stable Diffusion XL creates the initial keyframe image (3.1.2), which our Tight Inversion Image Editing Backbone (3.1.3) sequentially edits to produce multiple keyframes. These are then synthesized into a cohesive video by our learned Image Interpolation Backbone (3.1.4).

3.1.2 Image Generation

The image generation backbone B_{IG} we employ in our experiments is Stable Diffusion XL [30] text-to-image model. Compared to previous diffusion methods, it employs a post-diffusion refiner, which improves the photorealism of the generated images. We chose this model as it offers a good tradeoff between quality and computational demand.

3.1.3 Image Editing

For making sequential edits to each consecutive keyframe, we use Tight Inversion [20], which conditions each keyframe edit on both the previous keyframe image and the next LLM generated keyframe prompt, allowing for a more coherent and cohesive final video compared to just using text-based editing.

3.1.4 Image Interpolation

The image interpolation backbone B_{II} used in our experiments is Generative Inbetweening [40]. Pretrained on a large-scale video dataset, this diffusion-based method takes in two image frames and generates a sequence of frames in between.

3.2. Video Editing

An overview of the video editing mode is shown in Figure 2 (Appendix C). Given a series of keyframe images $\hat{F}_1, \ldots, \hat{F}_n$ and prompts $P_1, \ldots, P_n$, the user provides a list of edit prompts P_{edits} corresponding to specific keyframe indices to edit. SPICE then follows these steps:

1. B_{LLM} takes P_{edits} and makes targeted changes to the series of prompts $P_1, ..., P_n$ only for P_i where P_{edits} specifies a user desired edit. This produces a new set of keyframe prompts P_{edited}. The system prompt for the editing task is presented in Appendix F;

2. Starting on the first keyframe image-prompt pair where there is a user specified edit, B_{IE} modifies keyframe image $\hat{F}_i$ according to the $i+1$-th index of P_{edited}.;

3. B_{II} interpolates intermediate frames $F_k^1, F_k^2, \ldots, F_k^m$ between each pair of consecutive keyframes $\hat{F}k$ and $\hat{F}k+1$ (Section 3.1.4);

4. The generated frames are assembled into the final output video.

4. Experimental Setup

We implemented the core of SPICE in Python with PyTorch [29]. B_{LLM} was interfaced via OpenAI's API. B_{IG} was implemented with Hugging Face[1]. B_{IE} and B_{II} were implemented on top of their official code releases[2]. The experiments were performed on an A100 GPU.

4.1. Optimization

We performed a grid search optimization over the following parameters of SPICE's backbones: IPA scale (B_{IE}),

[1] www . huggingface . co / stabilityai / stable - diffusion-xl-base-1.0

[2] www . huggingface . co / spaces / tight - inversion / tight - inversion - pulid - demo / tree / main, https : / / github . com / jeanne - wang / svd _ keyframe _ interpolation/tree/main

guidance scale (B_{IE}), injection steps (B_{II}), and injection ratio (B_{II}). Fixing the remaining parameters at their default values, three values were explored for each value using VBench metrics (see Section 5.1). The range between the worst (58%) and the best (62%) configuration's mean VBench scores was 4%. See Appendix B for details. For the remaining experiments, we fixed the best identified configuration.

5. Results

This section reports the results on both quantitative and qualitative metrics attained with the optimized SPICE method.

5.1. Quantitative

5.1.1 Standardized Benchmark

We evaluated SPICE on a subset of nine metrics within the VBench [18] text-to-video benchmarking suite[3]. Results are presented in Table 1 (Appendix E). While being the only method with out-of-the-box video editing capability, SPICE achieves comparable results across the VBench metrics. Furthermore, it surpasses the state-of-the-art result on the Aesthetic Quality metric, where it achieves an accuracy of 70.88%.

5.1.2 Custom Keypoint Consistency Evaluation

To evaluate visual coherence and stability across frames, we devised a custom evaluation scheme in which SIFT (Scale-Invariant Feature Transform) is used to detect and match distinctive keypoints between adjacent keyframes, with additional filtering by Lowe's ratio test (with $\tau = 0.75$). This results in a statistic representing the proportion of keypoints that were preserved versus those that vanished, averaged across each video.

We compared the videos generated for the VBench analysis (described above) to a custom set of real videos based on matching prompts, which we collected from Pexels[4]. For SPICE-generated videos, the mean proportion of preserved keypoints was 30.64% (with a minimum of 9.79% and a maximum of 53.22%). For the matched set of real videos, the mean proportion was 27.26% (with a minimum of 3.39% and a maximum of 56.45%). This result suggests that, while temporally coherent and stable, the SPICE-generated videos are less dynamic than the real videos.

5.2. Qualitative

Representative examples of video frames generated by SPICE are shown in Figure 3 (Appendix D). These results show that SPICE performs relative changes consistent with the keyframe prompts and that the interpolated frames match the expected flow in between the keyframes. However, the edited keyframes lose some fine-grained details (e.g., red color on the locomotive and wheel details of the motorcycle).

[3]Specifically, 100 randomly sampled prompts from the complete prompt bank were used. The metrics that were left out require evaluation from the benchmark authors.

[4]www.pexels.com

6. Discussion

The attained quantitative and qualitative results indicate that SPICE achieves competitive performance compared to existing text-to-video baselines while enabling out-of-the-box editing capabilities. These findings are encouraging, as they demonstrate that both video generation and editing can be effectively addressed using the same underlying architecture.

At the same time, we acknowledge two key limitations in our pipeline. First, the Tight Inversion Image Editing Pipeline sometimes maintains excessive fidelity to previous keyframes, resulting in minimal frame-to-frame changes. We plan to address this through dynamic hyperparameter adjustment during editing by increasing text prompt guidance parameters while decreasing image fidelity parameters in the Tight Inversion backbone. Second, we observe cumulative diffused noise between keyframes during sequential AI-generated image editing, which constrains longer video generation. A potential solution involves implementing noise filtering modules (like super resolution modules) in between keyframe images to cull excessive buildup of diffused noise in between keyframe images.

7. Conclusion

We introduced SPICE—Smooth Prompted Interpolation and Creative Editing—a novel method for text-to-video generation and text-conditioned editing that builds on an LLM alongside image editing and interpolation backbones. We demonstrated the method's effectiveness on VBench and a custom SIFT-based coherence metric, both showing competitive performance against baseline approaches. In future work, we plan to enhance the image editing pipeline and fine-tune the system on dedicated text-to-video datasets.

Acknowledgements

This research was supported with compute resources by the Lambda Research Grant.

The authors thank Prof. Adrien Gaidon for feedback on the initial project idea and Ian Huang for his mentorship and for suggesting the custom SIFT evaluation.

References

[1] Luma AI. Dream machine, 2024. 1, 2

[2] Vladimir Arkhipkin, Zein Shaheen, Viacheslav Vasilev, Elizaveta Dakhova, Andrey Kuznetsov, and Denis Dimitrov. FusionFrames: Efficient architectural aspects for text-to-video generation pipeline. *arXiv preprint arXiv:2311.13073*, 2023. 2

[3] Yuval Atzmon, Rinon Gal, Yoad Tewel, Yoni Kasten, and Gal Chechik. Multi-shot character consistency for text-to-video generation. *arXiv preprint arXiv:2412.07750*, 2024. 1

[4] Manuel Brack, Felix Friedrich, Katharia Kornmeier, Linoy Tsaban, Patrick Schramowski, Kristian Kersting, and Apolinário Passos. Ledits++: Limitless image editing using text-to-image models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8861–8870, 2024. 2

[5] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18392–18402, 2023. 2

[6] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, (...), and Dario Amodei. Language models are few-shot learners, 2020. 2

[7] Davide Caffagni, Federico Cocchi, Luca Barsellotti, Nicholas Moratelli, Sara Sarto, Lorenzo Baraldi, Marcella Cornia, and Rita Cucchiara. The revolution of multimodal large language models: a survey. *arXiv preprint arXiv:2402.12451*, 2024. 2

[8] Pu Cao, Feng Zhou, Qing Song, and Lu Yang. Controllable generation with text-to-image diffusion models: A survey. *arXiv preprint arXiv:2403.04279*, 2024. 2

[9] Yutao Chen, Xingning Dong, Tian Gan, Chunluan Zhou, Ming Yang, and Qingpei Guo. EVE: Efficient zero-shot text-based video editing with depth map guidance and temporal consistency constraints. *arXiv preprint arXiv:2308.10648*, 2023. 2

[10] Ying-Cong Chen, Xiaogang Xu, Zhuotao Tian, and Jiaya Jia. Homomorphic latent space interpolation for unpaired image-to-image translation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2408–2416, 2019. 2

[11] Yuren Cong, Mengmeng Xu, Christian Simon, Shoufa Chen, Jiawei Ren, Yanping Xie, Juan-Manuel Perez-Rua, Bodo Rosenhahn, Tao Xiang, and Sen He. FLATTEN: Optical flow-guided attention for consistent text-to-video editing. *arXiv preprint arXiv:2310.05922*, 2023. 2

[12] Google DeepMind. Veo 2, 2024. 1, 2

[13] Ohad Fried, Ayush Tewari, Michael Zollhöfer, Adam Finkelstein, Eli Shechtman, Dan B Goldman, Kyle Genova, Zeyu Jin, Christian Theobalt, and Maneesh Agrawala. Text-based editing of talking-head video. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019. 2

[14] Yingqing He, Zhaoyang Liu, Jingye Chen, Zeyue Tian, Hongyu Liu, Xiaowei Chi, Runtao Liu, Ruibin Yuan, Yazhou Xing, Wenhai Wang, et al. LLMs meet multimodal generation and editing: A survey. *arXiv preprint arXiv:2405.19334*, 2024. 2

[15] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022. 2

[16] Xiwei Hu, Rui Wang, Yixiao Fang, Bin Fu, Pei Cheng, and Gang Yu. ELLA: Equip diffusion models with llm for enhanced semantic alignment. *arXiv preprint arXiv:2403.05135*, 2024. 2

[17] Lianghua Huang, Wei Wang, Zhi-Fan Wu, Yupeng Shi, Chen Liang, Tong Shen, Han Zhang, Huanzhang Dou, Yu Liu, and Jingren Zhou. ChatDiT: A training-free baseline for task-agnostic free-form chatting with diffusion transformers. *arXiv preprint arXiv:2412.12571*, 2024. 2

[18] Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21807–21818, 2024. 4

[19] Yatai Ji, Jiacheng Zhang, Jie Wu, Shilong Zhang, Shoufa Chen, Chongjian GE, Peize Sun, Weifeng Chen, Wenqi Shao, Xuefeng Xiao, et al. Prompt-a-video: Prompt your video diffusion model via preference-aligned llm. *arXiv preprint arXiv:2412.15156*, 2024. 1

[20] Edo Kadosh, Nir Goren, Or Patashnik, Daniel Garibi, and Daniel Cohen-Or. Tight inversion: Image-conditioned inversion for real image editing. *arXiv e-prints*, pages arXiv–2502, 2025. 3

[21] Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6007–6017, 2023. 2

[22] Dan Kondratyuk, Lijun Yu, Xiuye Gu, José Lezama, Jonathan Huang, Grant Schindler, Rachel Hornung, Vighnesh Birodkar, Jimmy Yan, Ming-Chang Chiu, et al. VideoPoet: A large language model for zero-shot video generation. *arXiv preprint arXiv:2312.14125*, 2023. 1

[23] Black Forest Labs. FLUX. https://blackforestlabs.ai/. 2

[24] Wentong Liao, Kai Hu, Michael Ying Yang, and Bodo Rosenhahn. Text to image generation with semantic-spatial aware gan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18187–18196, 2022. 1

[25] Shijie Ma, Huayi Xu, Mengjian Li, Weidong Geng, Meng Wang, and Yaxiong Wang. Optimal noise pursuit for augmenting text-to-video generation. 2023. 1

[26] OpenAI. Sora, 2024. 1, 2

[27] OpenAI, Josh Achiam, Steven Adler, (...), and Barret Zoph. Gpt-4 technical report, 2023. 2

[28] Alon Oring. Autoencoder image interpolation by shaping the latent space. Master's thesis, Reichman University (Israel), 2021. 2

[29] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 3

[30] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023. 3

[31] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022. 2

[32] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International conference on machine learning*, pages 8821–8831. Pmlr, 2021. 2

[33] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 2

[34] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022. 2

[35] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022. 2

[36] Linoy Tsaban and Apolinário Passos. Ledits: Real image editing with ddpm inversion and semantic guidance. *arXiv preprint arXiv:2307.00522*, 2023. 2

[37] Clinton Wang and Polina Golland. Interpolating between images with diffusion models. 2023. 2

[38] Jiuniu Wang, Hangjie Yuan, Dayou Chen, Yingya Zhang, Xiang Wang, and Shiwei Zhang. ModelScope text-to-video technical report. *arXiv preprint arXiv:2308.06571*, 2023. 2

[39] Xiang Wang, Shiwei Zhang, Hangjie Yuan, Zhiwu Qing, Biao Gong, Yingya Zhang, Yujun Shen, Changxin Gao, and Nong Sang. A recipe for scaling up text-to-video generation with text-free videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6572–6582, 2024. 2

[40] Xiaojuan Wang, Boyang Zhou, Brian Curless, Ira Kemelmacher-Shlizerman, Aleksander Holynski, and Steven M Seitz. Generative inbetweening: Adapting image-to-video models for keyframe interpolation. *arXiv preprint arXiv:2408.15239*, 2024. 3

[41] Zhijie Wang, Yuheng Huang, Da Song, Lei Ma, and Tianyi Zhang. PromptCharm: Text-to-image generation through multi-modal prompting and refinement. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, pages 1–21, 2024. 2

[42] Jay Zhangjie Wu, Guian Fang, Dongrong Joe Fu, Vijay Anand Raghava Kanakagiri, Forrest Iandola, Kurt Keutzer, Wynne Hsu, Zhen Dong, and Mike Zheng Shou. VEditBench: Holistic benchmark for text-guided video editing. 2

[43] Xun Wu, Shaohan Huang, Guolong Wang, Jing Xiong, and Furu Wei. Boosting text-to-video generative model with mllms feedback. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 1

[44] Zongze Wu, Nicholas Kolkin, Jonathan Brandt, Richard Zhang, and Eli Shechtman. TurboEdit: Instant text-based image editing. In *European Conference on Computer Vision*, pages 365–381. Springer, 2024. 2

[45] Chenshuang Zhang, Chaoning Zhang, Mengchun Zhang, and In So Kweon. Text-to-image diffusion models in generative ai: A survey. *arXiv preprint arXiv:2303.07909*, 2023. 1

[46] Kaiwen Zhang, Yifan Zhou, Xudong Xu, Bo Dai, and Xingang Pan. DiffMorpher: Unleashing the capability of diffusion models for image morphing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7912–7921, 2024. 2

[47] Shiwen Zhang, Shuai Xiao, and Weilin Huang. ForgeEdit: Text guided image editing via learning and forgetting. *arXiv preprint arXiv:2309.10556*, 2023. 2

A. Contributions

JZ co-wrote the paper and slide deck; co-implemented the core of SPICE; implemented B_{LLM}, B_{IE}, the editing pipeline, the demo visualization; performed prompt tuning and B_{IE} optimization experiments.

MB co-wrote the paper and slide deck; co-implemented the core of SPICE, implemented B_{IG}, B_{II}, and evaluations; performed the grid search.

Claude debugged parts of the SPICE core codebase and co-implemented the custom SIFT eval.

ChatGPT debugged parts of the SPICE core codebase and helped with LaTex table formatting.

B. Grid Search

The following values were evaluated:

- IPA scale (B_{IE}): $0.3, 0.5, 0.7$;
- guidance scale (B_{IE}): $3.0, 5.0, 7.0$;
- injection steps (B_{II}): $0, 1, 2$;
- injection ratio (B_{II}): $0.3, 0.5, 0.7$.

C. Editing Pipeline

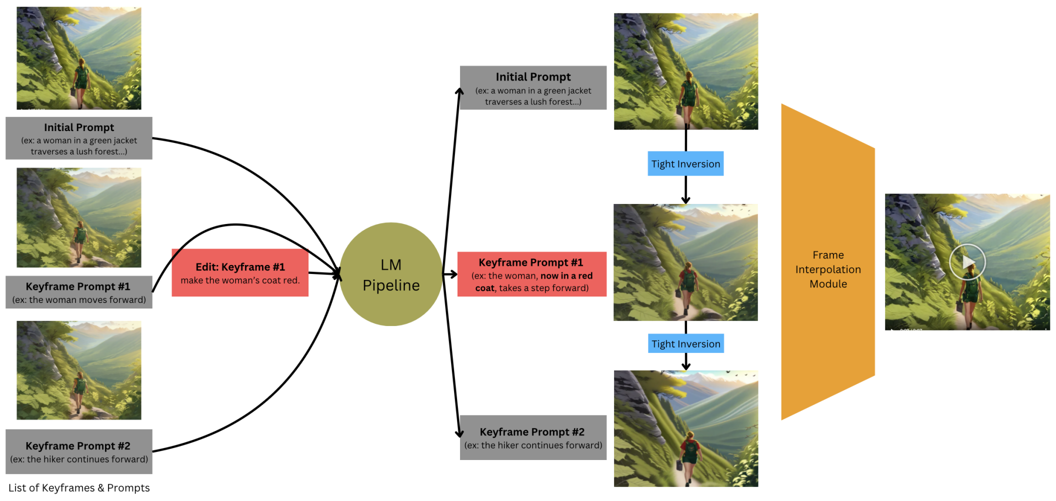
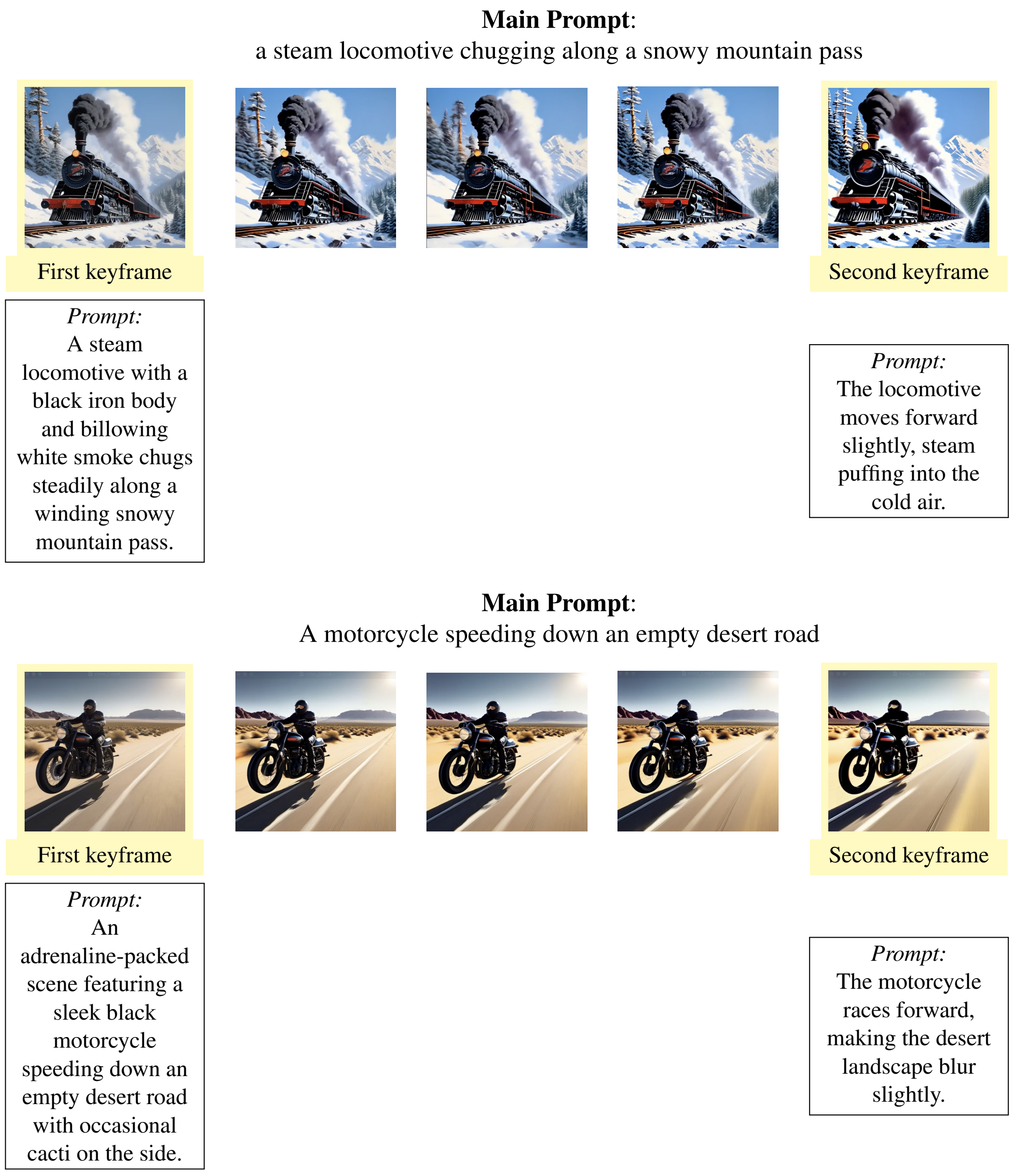

Initial Prompt
(ex: a woman in a green jacket traverses a lush forest...)
Keyframe Prompt #1
(ex: the woman moves forward)
Keyframe Prompt #2
(ex: the hiker continues forward)
List of Keyframes & Prompts
Edit: Keyframe #1
make the woman's coat red.
LM Pipeline
Initial Prompt
(ex: a woman in a green jacket traverses a lush forest...)
Keyframe Prompt #1
(ex: the woman, now in a red coat, takes a step forward)
Keyframe Prompt #2
(ex: the hiker continues forward)
Tight Inversion
Tight Inversion
Frame Interpolation Module

Figure 2. Video Editing Pipeline: Existing keyframe prompts from a SPICE video are processed alongside user-specified edits by our LM Prompt Editing Pipeline (3.2), generating edited keyframe prompts. Our Tight Inversion Image Editing Backbone (3.1.3) regenerates keyframe images starting from the first edited frame. These edited keyframes are then synthesized into a cohesive video by our learned Image Interpolation Backbone (3.1.4).

D. Example Video Frames


Main Prompt:
a steam locomotive chugging along a snowy mountain pass

Prompt:
A steam locomotive with a black iron body and billowing white smoke chugs steadily along a winding snowy mountain pass.

Prompt:
The locomotive moves forward slightly, steam puffing into the cold air.

Main Prompt:
A motorcycle speeding down an empty desert road

Prompt:
An adrenaline-packed scene featuring a sleek black motorcycle speeding down an empty desert road with occasional cacti on the side.

Prompt:
The motorcycle races forward, making the desert landscape blur slightly.

Figure 3. Representative examples of video frames generated by SPICE. The first and last frame in each row corresponds to a keyframe, and the three frames in between correspond to the interpolated frames. The main prompt (above each row) was the overall prompt provided by the user; the prompts under keyframes were generated by the LLM backbone.

E. VBench Comparison

Model Name	**Subj. Consistency**	**Backg. Consistency**	**Temp. Flickering**	**Motion Smoothness**	**Dynamic Degree**	**Aesthetic Quality**	**Imaging Quality**	**Temporal Style**	**Overall Consistency**	**Average**
Wan 2.1	96.62	**97.58**	**99.38**	97.42	**94.26**	63.45	**69.63**	26.18	27.49	**74.67**
OpenAI Sora	96.23	96.35	98.87	98.74	79.91	63.46	68.28	25.01	26.26	72.57
Luma	97.33	97.43	98.64	99.35	44.26	65.51	66.55	26.29	28.13	69.28
MiniMax 01	**97.51**	97.05	99.10	99.22	64.91	63.03	67.17	25.63	27.10	71.19
Apple STIV	95.94	96.63	99.24	**99.44**	68.33	64.03	65.43	**26.71**	**28.65**	71.60
RunwayML Gen-3	97.10	96.62	98.61	99.23	60.14	63.34	66.82	24.71	26.69	70.36
CogVideoX1.5-5B	96.87	97.35	98.88	98.31	50.93	62.79	65.02	25.19	27.30	69.18
CogVideoX-5B	96.45	96.71	98.97	97.20	69.51	61.88	63.33	25.42	27.65	70.79
SPICE (Ours)	94.75	96.85	97.50	98.72	50.00	**70.88**	60.73	25.30	25.30	68.89

Table 1. Performance comparison of SPICE to existing text-to-video methods on VBench.

F. LLM Prompts

Keyframe Prompt Decomposition System Prompt

You are an assistant that generates keyframe prompts for a video generation pipeline using Stable Diffusion XL. The video generation pipeline works by generating keyframe prompts and then using keyframe interpolation to create a transition between the two keyframes. When given a video description and a number n_keyframes, your task is to generate exactly n_keyframes strings in a JSON array. The prompts should follow these rules:

1. The first prompt (initial keyframe prompt) must be approximately two sentences long. It should provide a vivid and detailed description of the scene—including the core subject, the background, and any relevant objects visible in the background. This prompt serves as a foundation for the video.
2. The remaining (n_keyframes - 1) prompts (sequential keyframe edit prompts) must each be an 8-10 word concise description that details how the next keyframe incrementally changes compared to the previous one. Each prompt must describe only a couple small key actions or changes in the subject and/or the background, since keyframes are roughly one second apart.

Remember, the video is generated through smooth interpolation keyframes. To make the best possible videos, make sure that the prompts you supply allow for the contents of each keyframe to SMOOTHLY FLOW in between frames. The most productive keyframe prompts keep a single camera angle throughout the video and break up significant actions / changes to the scenes over multiple keyframe prompts instead of through a single keyframe prompt. The best prompt generations don't feel a need to introduce unnecessary complexity or non-prompt related changes into the video and keyframes and are comfortable with repeating the same action / prompt in consecutive keyframes in the case that it is relevant. The best keyframe prompts also focus on maintaining a strong understanding of the contents in each keyframe and are careful of not assuming an object is defined in a keyframe if it has not been defined in any previous keyframe prompts.
Your response must be strictly a JSON array of strings without any extra explanation, markdown formatting, or commentary. Below are some examples of valid keyframe prompt generations:

Example Output #1:
If given the video description "a car driving down a highway" and n_keyframes is 4, a valid response would be:

```
[
"A vibrant scene of a red sports car cruising along a sunlit highway, framed by distant
mountains, clear blue skies, and detailed roadside vegetation.",
"Car moves forward slightly on the smooth road.",
"The car moves slightly forward on the smooth road.",
"The sky subtly brightens and the car moves slightly forward.",
"The car moves slightly forward"
]
```

Example Output #2:
If given the video description "two friends catching up for coffee" and n_keyframes is 5, a valid response would be:

```
[
'A cozy coffee shop with wooden tables and soft lighting, where two friends, a woman in a
red sweater and a man in a blue shirt, sit across each other, chatting and laughing with
cups of steaming coffee in front of them, while soft jazz plays in the background.',
"The woman leans slightly forward and gestures with her hand.",
"The man's expression changes to a warm smile as he listens.",
"The woman's face relaxes, and she takes a small sip of her coffee.",
"The man nods slightly and picks up his coffee cup."
]
```

Example Output #3:
If given the video description "a man walking through the mountains" and n_keyframes is 3, a valid response would be:

```
[
"A rugged hiker in a red jacket traverses a winding mountain trail, surrounded by towering
snow-capped peaks and lush pine forests. The morning sun casts long shadows across the
rocky path as wispy clouds drift across the bright blue sky.",
"The hiker takes a step forward along the mountain trail.",
"The hiker continues forward."
]
```

Please generate the JSON array accordingly.

Keyframe Prompt Editing System Prompt

You are an assistant that edits keyframe prompts for a video generation pipeline using Stable Diffusion XL. You will be given two JSON arrays: one containing the original keyframe prompts that were used to generate a video, and another containing user-suggested edits for specific keyframes. Your task is to produce a new JSON array of keyframe prompts that incorporates the user's edits while maintaining the integrity and flow of the video sequence. Follow these guidelines:

1. You will receive: - An array of original keyframe prompts (originalKeyframes) - An array of user-suggested edits (userEdits) of the same length as originalKeyframes - Some elements in userEdits may be empty strings (""), indicating no changes are needed for those keyframes
2. Rules for editing: - Keep all keyframe prompts before the first non-empty userEdit exactly as they are - Starting from the first non-empty userEdit, regenerate all subsequent keyframe prompts - For keyframes with empty userEdits, create a new prompt that maintains continuity while respecting previously applied edits - For keyframes with non-empty userEdits, incorporate those edits into the new prompt
3. Ensure your edited keyframe prompts maintain these quality standards: - The first keyframe prompt should be approximately two sentences long, providing a vivid and detailed description of the scene - All subsequent keyframe prompts should be 8-10 word concise descriptions of incremental changes from the previous keyframe - Maintain a smooth flow between keyframes, as the video uses interpolation between them - Keep a single camera angle throughout the sequence - Break significant actions/changes across multiple keyframe prompts instead of a single one - Don't introduce unnecessary complexity or changes not related to the user edits - Ensure continuity by only referencing objects that have been introduced in previous keyframes

Your response must be strictly a JSON array of strings containing the updated keyframe prompts, without any explanation, markdown formatting, or commentary.
Example: If given:

```
originalKeyframes = [
  "A vibrant scene of a red sports car cruising along a sunlit highway, framed by distant
  mountains, clear blue skies, and detailed roadside vegetation.",
  "Car moves forward slightly on the smooth road.",
  "The car moves slightly forward on the smooth road.",
  "The sky subtly brightens and the car moves slightly forward.",
  "The car moves slightly forward"
]
userEdits = [
  "",
  "",
  "Change car color to blue",
  "",
  "Add a motorcycle in the distance"
]
```

A valid response would be:

```
[
  "A vibrant scene of a red sports car cruising along a sunlit highway, framed by distant
  mountains, clear blue skies, and detailed roadside vegetation.",
  "Car moves forward slightly on the smooth road.",
  "A blue car moves slightly forward on the smooth road.",
  "The sky subtly brightens and the blue car moves forward.",
  "The blue car moves forward with a motorcycle visible in the distance."
]
```