TransPixeler: Advancing Text-to-Video Generation with Transparency

Supplementary Material

1. Limitations

Our DiT-based method for RGBA generation incurs quadratic computational costs due to sequence expansion. However, our method achieves an optimal balance between generation and alignment when trained with a limited dataset. Numerous studies [1, 5, 7] have addressed the computational overhead of long sequences, with many optimizations reducing complexity to a linear scale. To enhance the efficiency of our method, we plan to incorporate these optimizations in future work. Additionally, our performance is influenced by the generative priors provided by the chosen T2V model, which affects the quality and consistency of our outputs.

2. Comparisons with Video Matting

We compare our method with video matting methods BiMatting [4] and Robust Video Matting (RVM) [3], as well as the image matting method Matte-Anything [6]. From the results, it is evident that most methods, trained on the VideoMatte240k [2] dataset, struggle to produce valid outputs for non-human objects, often resulting in empty results. Even image matting methods trained on large-scale datasets fail to handle certain visual effects correctly. Results are shown in the attached HTML source files.

3. Data Preprocessing

Color Decontamination. In our method, we preprocess the training data by applying a color decontamination step to enhance the quality of the RGBA video generation. Color contamination typically occurs when there is an undesired blending of foreground and background colors, especially along the edges of an object, due to imperfect alpha masks. This blending causes color bleeding, where the foreground and background colors mix, resulting in lower quality RGBA frames with inaccurate color representation. To address this issue, we refine the alpha mask using parameters such as gain ($\gamma = 1.1$) and choke ($\chi = 0.5$) to adjust the sharpness and influence of the mask edges. The decontaminated RGB values are then computed as follows:

$$\text{RGB}_{\text{decon}} = \text{RGB} \times (1 - \text{mask}_{\text{refined}}) + \text{mask}_{\text{refined}} \times \text{Background}$$

This equation ensures that unwanted color contamination is minimized, providing a more precise distinction between foreground and background regions. By performing this preprocessing step, we generate high-quality training data that significantly improves the performance of our RGBA video generation model.

Background Blurring. Unlike typical training strategies in video matting methods, where objects are composited with complex backgrounds to increase the difficulty of the task, our goal is to support joint generation of alpha and RGB channels while ensuring alignment between them. Instead of emphasizing complex matting, we focus on generating consistent and high-quality output by compositing objects with simple, static backgrounds that match the black areas in the alpha channel. Specifically, we apply a large Gaussian blur kernel of size 201 to the first frame to create a blurred background and blend each subsequent frame with this static background. This approach helps simplify the training conditions, allowing the model to better align the RGB and alpha components while maintaining high-quality output.

4. Optical Flow Difference

To evaluate the alignment between the RGB and alpha channels in generated videos, we introduce a metric based on optical flow difference. Optical flow measures the apparent motion of objects between consecutive frames, and comparing the optical flow fields of RGB and alpha frames provides insight into the consistency of motion across these modalities. Specifically, we use the Farneback method (`cv::calcOpticalFlowFarneback`) to compute the optical flow for both RGB and alpha frames, and then calculate the average Euclidean distance between their flow vectors as a measure of misalignment. This approach quantifies the degree to which the RGB and alpha channels align in terms of motion.

Pseudo Code Overview:

1. **Load consecutive RGB and alpha frames** from the input video.
2. **Convert the frames to grayscale** for optical flow computation, as optical flow is typically calculated on intensity values.
3. **Compute optical flow using the Farneback method** (`cv::calcOpticalFlowFarneback`) for both the RGB and alpha frames.
4. **Calculate the Euclidean distance** between the RGB and alpha flow vectors for each pixel.
5. **Average the differences** across all pixels and frames to obtain the final optical flow difference.

The average optical flow difference provides a quantitative metric for evaluating the alignment between RGB and alpha channels, helping to ensure that both modalities exhibit consistent motion.

5. Video Results

For all video results shown in the main paper, please see the attached HTML source files.

6. Additional Visual Results

In addition to the video results in the main paper, we provide more generated results in the supplementary files, including various objects and visual effects. Please find the corresponding results in the supplementary files.

References

[1] Jiayu Ding, Shuming Ma, Li Dong, Xingxing Zhang, Shaohan Huang, Wenhui Wang, Nanning Zheng, and Furu Wei. Longnet: Scaling transformers to 1,000,000,000 tokens 2023. *arXiv preprint arXiv:2307.02486*, 2023. 1

[2] Shanchuan Lin, Andrey Ryabtsev, Soumyadip Sengupta, Brian L Curless, Steven M Seitz, and Ira Kemelmacher-Shlizerman. Real-time high-resolution background matting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8762–8771, 2021. 1

[3] Shanchuan Lin, Linjie Yang, Imran Saleemi, and Soumyadip Sengupta. Robust high-resolution video matting with temporal guidance. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 238–247, 2022. 1

[4] Haotong Qin, Lei Ke, Xudong Ma, Martin Danelljan, Yu-Wing Tai, Chi-Keung Tang, Xianglong Liu, and Fisher Yu. Bimatting: Efficient video matting via binarization. *Advances in Neural Information Processing Systems*, 36:43307–43321, 2023. 1

[5] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020. 1

[6] Jingfeng Yao, Xinggang Wang, Lang Ye, and Wenyu Liu. Matte anything: Interactive natural image matting with segment anything model. *Image and Vision Computing*, 147: 105067, 2024. 1

[7] Chen Zhu, Wei Ping, Chaowei Xiao, Mohammad Shoeybi, Tom Goldstein, Anima Anandkumar, and Bryan Catanzaro. Long-short transformer: Efficient transformers for language and vision. *Advances in neural information processing systems*, 34:17723–17736, 2021. 1