Learning Inclusion Matching for Animation Paint Bucket Colorization

Supplementary Material

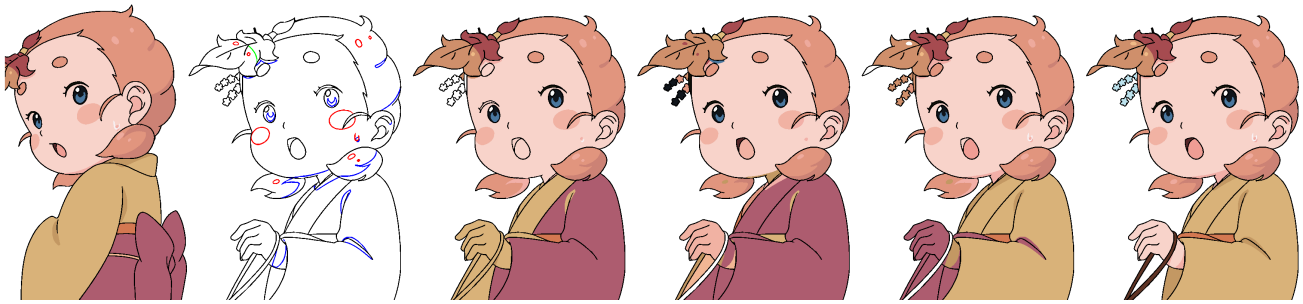


Figure 11. Visual comparison of different methods' performances on challenging scenes.

A. Data Augmentation

Different from optical flow annotations provided in AnimeRun [30], index labels in our dataset can produce direct inclusion matching correspondences even for non-adjacent frames. Notably, even when the reference frame and target frame are the same frame, the inclusion relationship still may not be apparent to the network. During the training process, we set the inter-frame intervals as 0, 1, and 2, respectively, as a form of data augmentation. Additionally, we employ separate random translation in $U(-400, 400)$, random rotation in $U(-\pi/9, \pi/9)$, and random resize in $U(1/2, 1/3)$ for paired frames to emulate typical movements in animation. In the resizing augmentation, a min pooling strategy is implemented to prevent the removal of some pixels in black lines through nearest or bilinear interpolation. After these augmentation steps, all training data undergo cropping to achieve a consistent size of 640×640.

B. Limitation

While our inclusion matching pipeline addresses many-to-many matching scenarios, it may still encounter challenges in complex occlusion situations and when faced with exaggerated motion or distortion, potentially leading to incorrect matching. Existing reference-based pipelines necessitate the presence of all segments from the target frame in the reference frame, making it difficult to accurately colorize new segments in the scenes like opening eyes or turning around, as the hand illustrated in Fig. 11. Despite these challenges, our method still shows superior performance compared to existing approaches in such demanding scenarios.

C. Color Line Colorization

In the animation industry, color lines (such as the blue and red lines in Fig. 12) are often used to mark shadows and highlights. In general, these color lines also need to be colorized as the adjacent shadow or highlight's colors [21]. This routine ensures a seamless integration of the color lines with the surrounding visual elements. Furthermore, the animators are required to label the highlight and shadow regions with yellow and blue as shown in Fig. 12(b) to prevent any potential confusion among downstream digital painters in animation production standards [22]. Thus, the segments of color lines can be merged into the adjacent highlight or shadow segment for inclusion matching. This approach eliminates the necessity to independently predict the color of each color line. For evaluation in the Table 3 and Table 4 of the main paper, the annotations of our segment map which we use to calculate the segment accuracy are derived from calculating the connected component of each color in the colorized ground truth. This design prevents our benchmark from penalizing methods that do not specifically cater to color line colorization.

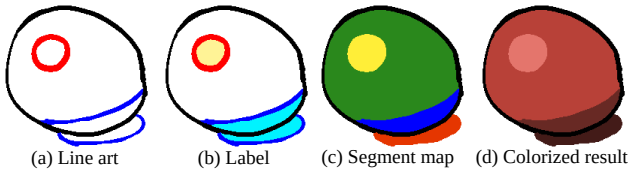
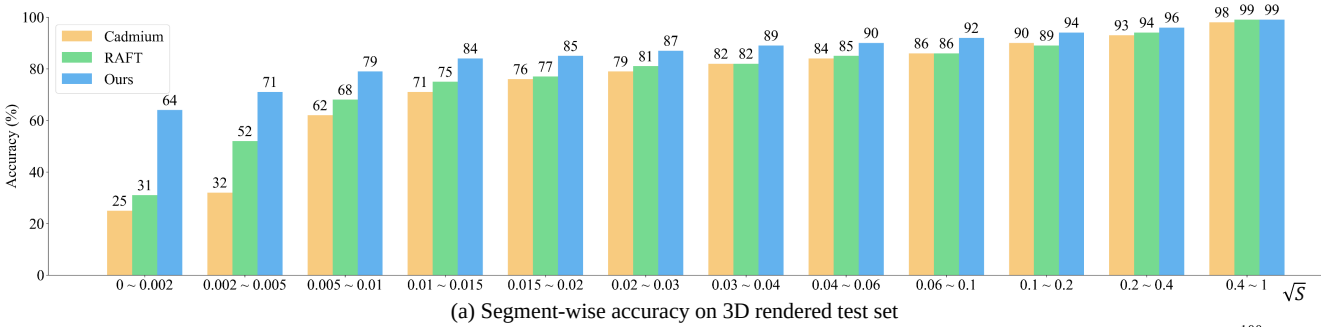


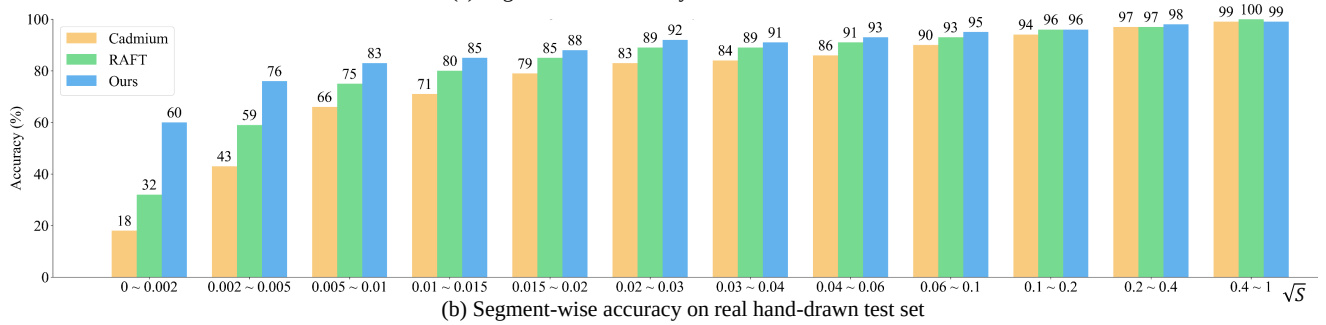
Figure 12. In the animation industry, color lines need to be colorized as the colors of the adjacent highlight and shadow regions as (d). Besides, the animators will label the highlight and shadow with yellow and blue as shown in (b) of each segment to avoid confusion. Thus, in the evaluation, we merge the color lines with the adjacent highlight and shadow as the segment map (c) to avoid estimating these color lines' colors.

D. More Analysis

Colorization for Tiny Segments. Colorizing tiny segments poses a significant challenge in paint bucket coloriza-



(a) Segment-wise accuracy on 3D rendered test set



(b) Segment-wise accuracy on real hand-drawn test set

Figure 13. Accuracy comparisons for segments of different sizes. We calculate the segment-wise accuracy in different bins, each partitioned according to the proportional area of the respective segment, denoted as S. Since the majority of segments have small areas, we partition the intervals using $\sqrt{S}$ and ensure a roughly equal number of segments in each bin. Same as the experiment in the main paper, the RAFT [35] is finetuned on the AnimeRun [30] dataset, and the official implementation of AnT [8] is realized through the Cadmium application [2].

Figure 14. Visual comparisons of tiny segment colorization across different methods. Even in scenarios with no corresponding segment in the reference frame, our approach still excels in colorizing tiny segments accurately.

tion, consuming a considerable amount of time for digital painters. Despite the existence of methods based on segment matching [8] and optical flow, accurate colorization of tiny segments remains challenging. In Fig. 13, we evaluate the accuracy of segment colorization across various sizes. Our proposed method outperforms current optical-flow-based approaches [30, 35] and AnT [8], particularly excelling in tiny segments. Even when dealing with segments as small as a few pixels, our inclusion matching training pipeline demonstrates its effectiveness in guiding the network to produce accurate and reliable results. The visual comparisons in Fig. 14 highlight the superior performance of our method in handling these challenging tiny segments.

Visualization of Ablation Study. In Fig. 15, we train our baseline model with AnimeRun and our proposed *PaintBucket-Character* dataset. Compared with the base-

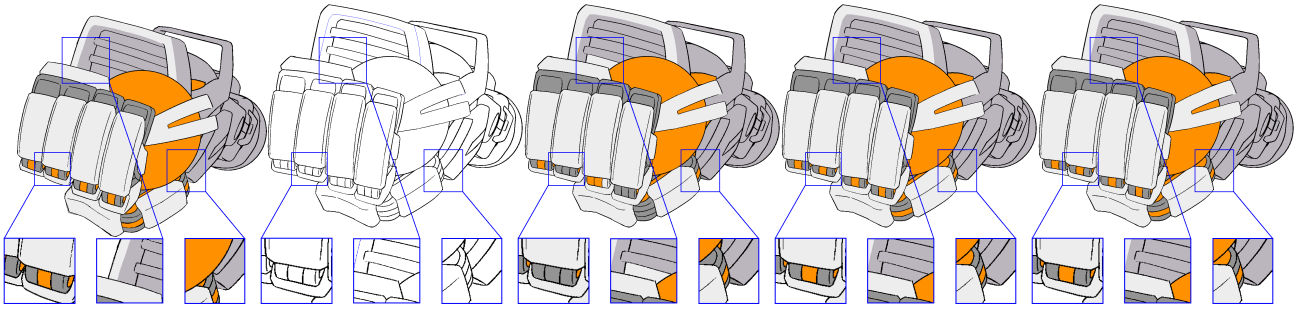


Figure 15. Visual comparison on training network with different datasets. 'PBC' represents our proposed dataset *PaintBucket-Character*.

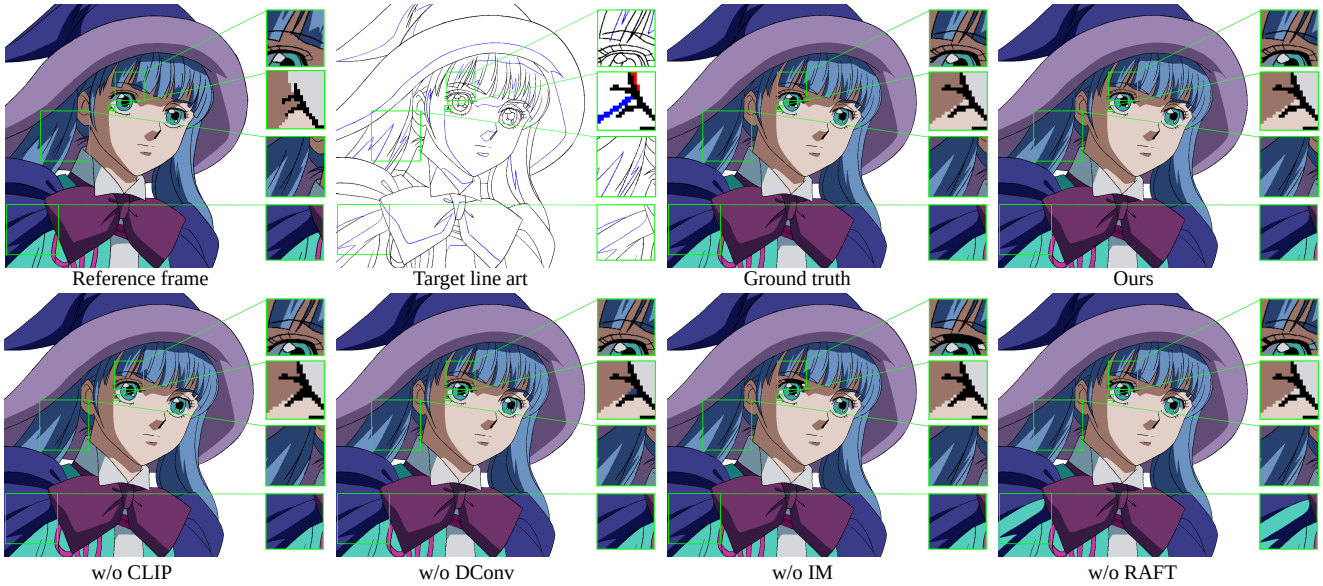


Figure 16. Ablation study for different modules of our method. Zoom in for better visualization.

line model trained on AnimeRun, our method can achieve more robust visualization results. Although our dataset only consists of anime characters, Fig. 15 demonstrates that our method can also generalize well on different scenes. Also, we conduct an ablation study on deformable convolution (w/o DConv), coarse warping module (w/o RAFT), inclusion matching pipeline (w/o IM), and CLIP image encoder (w/o CLIP). Fig. 16 particularly highlights the crucial role of the coarse warping module and inclusion matching. The warped color feature introduces a leakage of grouping information from the reference frame, facilitating the feature extraction network in learning inclusion relationships more effectively compared to relying solely on urging the multiplex transformer to acquire this representation. Thus, lacking coarse warping or inclusion matching will all lead to mismatching while there is no one-to-one correspondence as shown in the eyelash part of Fig. 16.

Reference-based Pixel-wise Approaches. Latent diffusion [24] can support different types of ways to encode the condition of the reference image. In our experiments, we compare the performance of IP-Adapter [38] and ControlNet's reference-only preprocessor [42] as reference frame encoder in Fig. 17. Also, we test using or not using text prompts in this figure. The target line art is encoded using ControlNet's line art model. Meanwhile, finetuning-based methods such as LoRA [13] also serve as a potential solution for reference-based colorization. We also finetune the additional LoRA [13] for each character based on 9 frames of each character's T-pose and 1 reference frame following DreamBooth [26]. Then, we use ControlNet's line art model and finetuned LoRA to colorize the target line as shown in Fig. 18 with text prompt 'A [V] character'. The visualization results demonstrate that these ControlNet-based methods cannot restore accurate colors even with the color mapping strategy. Besides, the limited resolution in latent diffusion's denoising process results in the fact that the colors of character details such as eyes are difficult to maintain. Compared with these methods, our method can achieve more accurate results and can even function well on pixel-level details.

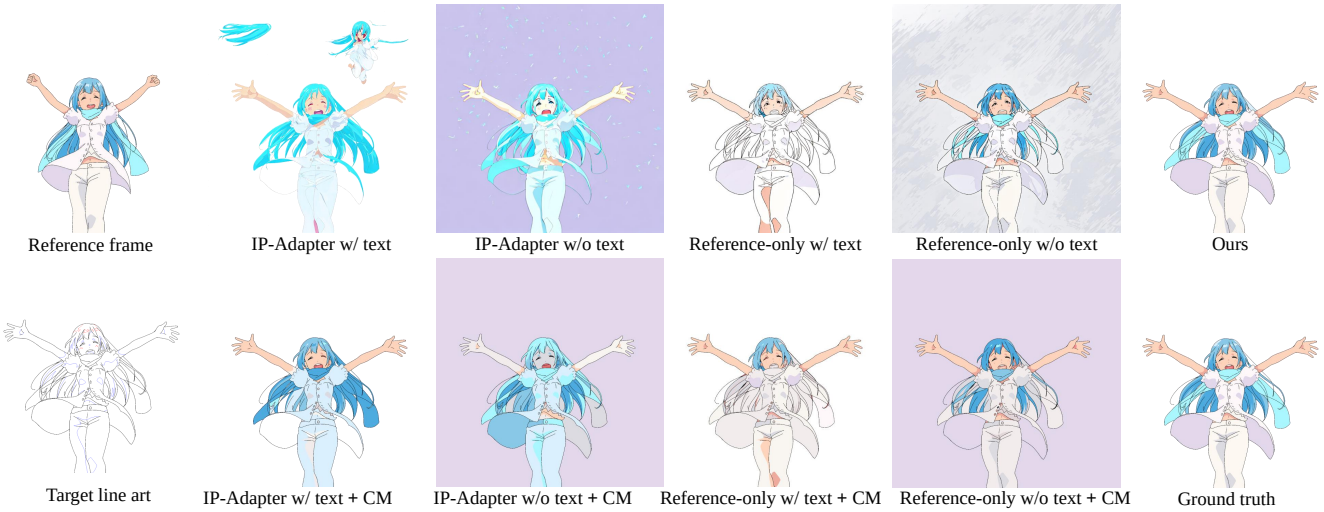


Figure 17. Visual comparisons of our method and different reference-based pixel-wise line art colorization methods. 'w/ & w/o text' means using or not using the text prompt 'A character in the white background'. In the second row, we employ 'CM' (color mapping), a technique that maps the average color in each segment to the closest color in the reference frame. Compared with these pixel-wise colorization methods, our method can yield more reliable results by avoiding filling similar colors in segments.

Figure 18. Visual comparisons of our method and finetuning-based pixel-wise line art colorization method. We finetune the LoRA [13] for each character using DreamBooth's pipeline and apply ControlNet [42] to control the contour of the colorized result.