Deep Geometrized Cartoon Line Inbetweening
Supplementary File

Li Siyao[1] Tianpei Gu[2*] Weiye Xiao[3] Henghui Ding[1] Ziwei Liu[1] Chen Change Loy[1] ✉
[1]S-Lab, Nanyang Technological University [2]Lexica [3]Southeast University
{siyao002, henghui.ding, ziwei.liu, ccloy}@ntu.edu.sg, gutianpei@ucla.edu, 230189776@seu.edu.cn

In this supplementary file, we provide the detailed network structures mentioned in the main paper. Additionally, we present further comparisons to state-of-the-art raster frame interpolation methods. To better illustrate our motivation and results, we have included a supplementary video that presents our work in a more intuitive manner.

A. Network Structures

A.1. Vertex Geometric Embedding.

The vertex geometric embedding in Section 4.1 of the main paper involves three convolutional encoders: image context encoder $\mathcal{E}_I$, positional encoder $\mathcal{E}_P$ and topological encoder $\mathcal{E}_T$. Among of them, $\mathcal{E}_I$ is a 2D CNN, while $\mathcal{E}_P$ and $\mathcal{E}_T$ are 1D CNNs within the same network structures. The detailed networks are shown in Table 1. The input channels C_{in} of $\mathcal{E}_I$, $\mathcal{E}_P$ and $\mathcal{E}_T$, the input channel C_{in} are 3, 2 and 64, respectively. The output channel C_{out} is 128 and the intermediate output channel dimensions $C^{(1)}$ and $C^{(2)}$ are 32 and 64, respectively, for all encoders.

A.2. Vertex Correspondence Transformer.

The Vertex Correspondence Transformer consists of a cascade of alternating self-attention (SA) and cross-attention (CA) layers. The architecture of the attention layer is displayed in Figure 1. In our implementation, each attention layer comprises four heads ($H = 4$). The input channel is set to 128, as shown above. The model is composed of a total of 24 attention layers, with 12 for each.

A.3. Visibility Mask Predictor

In *AnimeInbet*, we adopt an MLP to predict the visibility of vertices in the intermediate vector graph G_t. The MLP is implemented as a three-layer 1D CNN with kernel size of 1 as shown in Table 1. Channels are set as $C_{in} = 128$, $C^{(1)} = 64$, $C^{(2)} = 32$ and $C_{out} = 1$.

✉ Corresponding author. *Work completed at UCLA.

Table 1: **Architectures of vertex encoders and visibility mask predictor (Section 4.1)**. "Conv1D" and "Conv2D" represent 1D and 2D convolutions, respectively, and their arguments represent the input channel number, the output channel number, the kernel size, the convolution stride, the padding size on both ends of input data, and the dilation number in turn. InstanceNorm denotes Instance Normalization [5]. C_{in} and C_{out} denotes the input and Output Channels.

	2D CNN (used for $\mathcal{E}_I$)
	Input: 0, **Argument:** C_{in}, $C^{(1)}$, $C^{(2)}$, C_{out}
1	Conv2D(C_{in}, $C^{(1)}$, 7, 3, 1, 1)
2	InstanceNorm($d = C^{(1)}$)
3	ReLU( )
4	Conv2D($C^{(1)}$, $C^{(2)}$, 3, 1, 1, 1)
5	InstanceNorm($d = C^{(2)}$)
6	ReLU( )
7	Conv2D($C^{(2)}$, C_{out}, 3, 1, 1, 1)
8	InstanceNorm($d = C_{out}$)
	Output: 8
	1D CNN (used for $\mathcal{E}_P$, $\mathcal{E}_T$ and mask predictor)
	Input: 0, **Argument:** C_{in}, $C^{(1)}$, $C^{(2)}$, C_{out}
1	Conv1D(C_{in}, $C^{(1)}$, 1, 0, 1, 1)
2	InstanceNorm($d = C^{(1)}$)
3	ReLU( )
4	Conv1D($C^{(1)}$, $C^{(2)}$, 1, 0, 1, 1)
5	InstanceNorm($d = C^{(2)}$)
6	ReLU( )
7	Conv1D($C^{(2)}$, C_{out}, 1, 0, 1, 1)
	Output: 7

B. More Comparison Results

We present more visual comparisons to state-of-the-art raster-image-based frame interpolation methods including VFIformer [3], EISAI [1], RIFE [2] and FILM [4] in Figure 2. Following the main paper, we assign these examples in an increasing difficulty from top to bottom. As illustrated in the main paper, compared methods usually produce

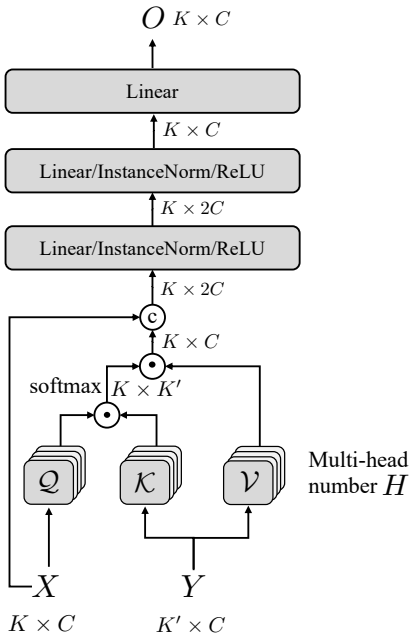

Figure 1: **Attention layer.** For self attention, the inputs X and Y are the same, while for cross attention, X and Y are different. $\mathcal{Q}$, $\mathcal{K}$ and $\mathcal{V}$ are linear projections with H heads. c denotes concatenation.

severe artifacts that makes the inbetweened frame invalid for further use in anime, while our method can keep a relatively complete, clean and concise structure of the line art.

References

[1] Shuhong Chen and Matthias Zwicker. Improving the perceptual quality of 2d animation interpolation. In *ECCV*, 2022. 1

[2] Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. Real-time intermediate flow estimation for video frame interpolation. In *ECCV*, 2022. 1

[3] Liying Lu, Ruizheng Wu, Huaijia Lin, Jiangbo Lu, and Jiaya Jia. Video frame interpolation with transformer. In *CVPR*, 2022. 1

[4] Fitsum Reda, Janne Kontkanen, Eric Tabellion, Deqing Sun, Caroline Pantofaru, and Brian Curless. Film: Frame interpolation for large motion. In *ECCV*, 2022. 1

[5] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 1

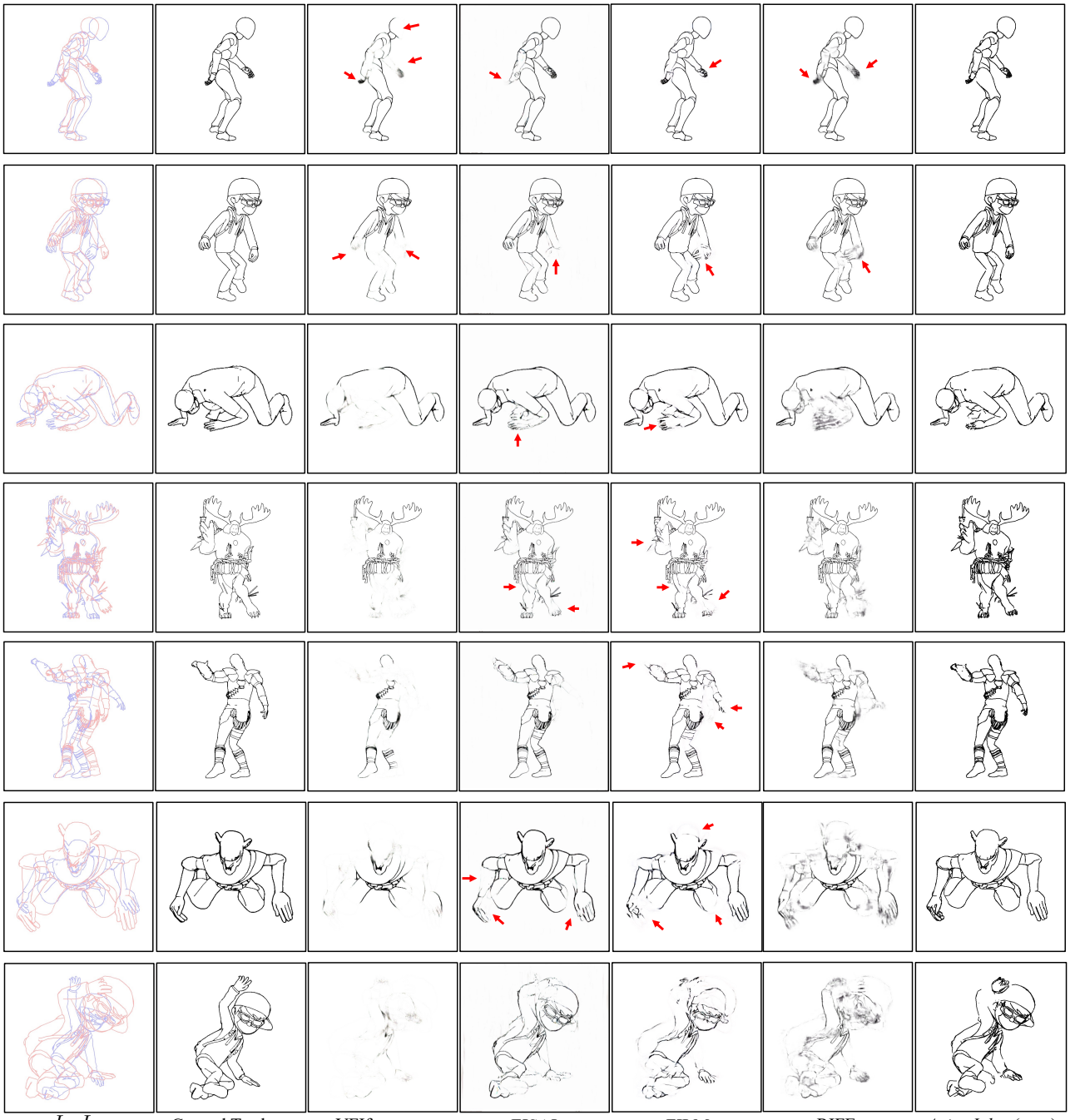

Figure 2: **Inbetweening results on *MixamoLine240* test set.** Examples are arranged from small (top) to large (bottom) motion magnitudes.