Deep Animation Video Interpolation with Global Motion Aggregation and Style Loss

LIANG KONG (HILBERT)
SID: 480043180

Supervisor: A/Prof. Zhiyong Wang

This thesis is submitted in partial fulfillment of
the requirements for the degree of
Bachelor of Advanced Computing (Honours)

School of Computer Science
The University of Sydney
Australia

29 May 2022

Student Plagiarism: Compliance Statement

I certify that:

I have read and understood the University of Sydney Student Plagiarism: Coursework Policy and Procedure;

I understand that failure to comply with the Student Plagiarism: Coursework Policy and Procedure can lead to the University commencing proceedings against me for potential student misconduct under Chapter 8 of the University of Sydney By-Law 1999 (as amended);

This Work is substantially my own, and to the extent that any part of this Work is not my own, I have indicated that it is not my own by Acknowledging the Source of that part or those parts of the Work.

Name: Liang Kong (Hilbert)

Signature: Liang Kong **Date**: May 29, 2022

Abstract

Traditional 2D cartoons are usually produced under a low frame rate because drawing keyframes by hand is labour-intensive. It would be very helpful if new frames could be interpolated automatically.

Although Video Frame Interpolation (VFI) attempts have been made by both researchers and artists, however, existing video interpolation methods perform poorly on animation videos because they are designed to analyse natural video data. Unlike natural videos, animation videos consist of only outlines and textureless colour blocks. Moreover, due to the low FPS nature, the displacements between each frame are even larger. There is very little research on this topic and the state-of-the-art method still suffers from many issues, such as Blurry interpolated results, Ineffective occlusion handling mechanism in motion estimation, and Colour bleeding results.

We propose a novel deep network by integrating a Global Motion Aggregation (GMA) module along with the current state-of-the-art animation video interpolation model AnimeInterp. We aim to tackle the occlusion handling problem since GMA can propagate the motion information of non-occluded points to the occluded pixels. In addition, we also reset the training goal from $\mathcal{L}_1$ loss to a Style loss which makes the trained model generate sharp and perceptually pleasant predictions.

We train and evaluate all proposed models on the ATD-12K dataset. The AnimeInterp + GMA model trained on L1 loss outperforms the original AnimeInterp model by 0.001 in SSIM but possesses a difference of 0.062 dB in PSNR. In addition, we also introduce LPIPS to emphasise the perceptual quality of the interpolated results. The AnimeInterp + GMA model trained on the Style loss surpasses the AnimeInterp trained on the same loss function by 1.417 in LPIPS.

Acknowledgements

I would like to express my sincere gratitude to my thesis supervisor, Associate Professor Zhiyong Wang for guidance throughout the research period and for spending precious time in weekly meetings with me to track my progress.

Also, I would like to thank Dr. Kun Hu for assisting A/Prof. Wang in terms of meeting arrangements as well as offering technical support such as server maintenance for the practical aspect of the project.

Finally, special thanks to Sujin Cho from Chung-Ang University, South Korea for sharing her thoughts and providing valuable feedback on GitHub as well as Wenjin Deng (Winston) from Xiamen University, China for offering various help on Pytorch and Deep learning practice in general.

Contents

Student Plagiarism: Compliance Statement **ii**

Abstract **iii**

Acknowledgements **iv**

List of Figures **viii**

List of Tables **x**

Chapter 1 Introduction **1**

1.1 Motivation 1

1.2 Contributions 2

1.3 Thesis Structure 3

Chapter 2 Background and Related Work **4**

2.1 Perceptual Quality Metrics 4

2.2 Optical Flow Estimation 6

2.2.1 Global Motion Aggregation 7

2.3 Video Frame Interpolation 8

2.3.1 Animation Frame Interpolation (AFI) 8

2.3.2 Frame Interpolation for Large Motion (FILM) 10

Chapter 3 Method **12**

3.1 Segment-Guided Matching (SGM) 12

3.1.1 Colour Piece Segmentation 13

3.1.2 Feature Extraction 14

3.1.3 Colour Piece Matching 14

3.1.4 Flow Generation 15

3.2 Recurrent Flow Refinement and Global Motion Aggregation Network (RFR+GMA) 16

3.2.1 Global Motion Aggregation Module 16

3.2.2 Recurrent Flow Refinement Module 18
3.2.3 Overall Workflow 18
3.3 Frame Warping and Synthesis 19
3.4 Loss Functions 19
3.4.1 L1 Loss 19
3.4.2 VGG Loss 20
3.4.3 Gram Matrix Loss 20
3.4.4 Style Loss 20

Chapter 4 Experiments **22**
4.1 Project Constraint 22
4.2 Dataset 22
4.3 Evaluation Metrics 23
4.4 Experiments Setup 24
4.4.1 AnimeInterp - $\mathcal{L}_1$ 24
4.4.2 AnimeInterp - $\mathcal{L}_s$ 24
4.4.3 AnimeInterp + GMA - $\mathcal{L}_1$ 25
4.4.4 AnimeInterp + GMA - $\mathcal{L}_s$ 25

Chapter 5 Results **26**
5.1 Preliminary Experiments 26
5.1.1 AnimeInterp Reproduction 26
5.1.2 AnimeInterp vs. FILM 27
5.2 Quantitative Evaluation 28
5.3 Qualitative Evaluation 30
5.3.1 Blurry Boundary in Estimated Flows 30
5.3.2 Out-of-Frame Motion 30
5.3.3 Sharpness 31
5.3.4 Detail Inpainting 31
5.4 Limitations 33

Chapter 6 Conclusion and Future Work **35**
6.1 Conclusion 35
6.2 Future Work 35

References **37**

List of Figures

2.1 Mean Squared Error (hence of PSNR) Counter Example [1] 4

2.2 Comparison between metrics in different image processing tasks [2] 5

2.3 **Pyramidal Processing:** By down-sampling, a large motion becomes smaller. 6

2.4 A typical occlusion scenario. [3] 7

2.5 (a) The overall pipeline of AnimeInterp. (b) The inner structure of the SGM module. (c) The workflow of the RFR module. [4] 9

2.6 FILM Architecture Overview [5] 10

3.1 The overall pipeline of AnimeInterp+GMA. 12

3.2 The inner structure of the SGM module. 13

3.3 The workflow of the RFR+GMA module. 16

3.4 The inner structure of the GMA module. 17

4.1 ATD-12K samples. 22

5.1 **Out-of-frame Movements:** Japan_v2_2_033283_s3 (The lowest PSNR among 2,000 triplets). 26

5.2 **Blur:** Disney_v4_20_043850_s1. 27

5.3 **Colour Bleeding:** Disney_v4_14_002999_s1 (The lowest SSIM among 2,000 triplets). 27

5.4 Min Δ PSNR and Min Δ SSIM 28

5.5 **Interpolation Error** of Evaluation on ATD-12K Test Set for AnimeInterp + GMA $\mathcal{L}_1$. 29

5.6 **PSNR** of Evaluation on ATD-12K Test Set for AnimeInterp + GMA $\mathcal{L}_1$. 30

5.7 Visual Comparison between Refined Flows output from AnimeInterp and AnimeInterp + GMA 31

5.8 Visual Comparison along with Ground-truth on Japan_v2_0_104298_s2 32

5.9 Visual Comparison along with Ground-truth on Japan_v2_3_129363_s3 and Japan_v2_2_008563_s1 32

5.10 Visual Comparison along with Ground-truth on Japan_v2_3_162765_s2 33

5.11 Visual Comparison along with Ground-truth on Disney_v4_21_041221_s2 34

List of Tables

5.1 **AnimeInterp vs. FILM on ATD-12K**. Note that the LPIPS computed is multiplied by 100 for readability. 27

5.2 Δ PSNR and Δ SSIM between FILM and AnimeInterp 27

5.3 Quantitative results on the test set of ATD-12K. The best and runner-up values are bold and underlined, respectively. 29

CHAPTER 1

Introduction

Traditionally, an animated series consists of thousands of illustrations, which are typically painted by hand. With a growing interest in traditional-styled animations from a global audience, studios and animators are under extreme pressure due to the incredibly labour-intensive process. As a result, in practice, it is common to see producers replicate one drawing two or three times to ease the burden (at the cost of reducing the actual frame rate) and reuse similar frames to avoid another hand-drawing process. Many research attempts have been made to aid the animators: Sykora *et al.* [6] propose an interactive tool which simplifies the sketch colorization process by filling the color within a region. Whited *et al.* [7] present the BetweenIT system for the user-guided automation of tight inbetweening. However, those tools and systems only solve a proportion of the process such as colourisation or in-between frames productions. There are still many aspects that require manual labour. Is it possible to automatically generate without any hand drawings? With the aid of recent advancements in Computer Vision, it is possible. More specifically, Video Frame Interpolation (VFI), a method of generating intermediate frames directly from a video sequence, can be the solution to reduce animators' burden.

1.1 Motivation

Different from real-life videos, cartoon frames are occupied with contours and colour blocks. There are not any existing fine textures and this leads to the denial of a variety of VFI models which significantly depend on the textures to perform motion estimation. This type of characteristic has already been identified in [4].

We would like to go one step deeper to discuss the primary reason why separate approaches toward VFI and Animation Frame Interpolation (AFI) should be applied. It is strictly related to the idea of ground truths. Natural videos are a recording of the moment that happens at a time point and location. If we have another person holding the same camera with the same angle at the same time and exact location, he can easily re-produce a video recording. The two video clips recorded should be identical and the pixels collected should be the same if we fix all the variables. Hence, natural videos contain the concept of ground truth.

While for animations, cartoon frames are not observations of the one and only universe. They can vary. If you ask an animator to re-produce a keyframe of his previous work, it is likely to end up with a slightly different frame. However, if the animator substitutes the old frame with a new frame. You probably will not notice since the animation is still perceptually smooth compared to the previous version. Therefore, we believe that for AFI, instead of a single ground truth, there exists a group of correct answers and those frames share common characteristics which we define as **Style**. So if an interpolated frame matches the perceptual similarity among the Style group, the output is acceptable.

In the aspect of applications, VFI can help in tasks such as Video Restoration which is essentially critical to the information generated in the interpolated frames. For AFI, it is more likely to be applied to post-process animation videos with the purpose of providing entertainment in which case, the truthness of synthesised results is less significant or the ground truth ,in this case, is subjective. Even the original creators cannot decide what the ground truth of the mid-frame "is", but what the ground truth is expected to be "like".

Therefore, without strictly enforcing the pixel-wise difference, instead, we give the model a certain degree of freedom but guide it with perceptual quality metrics. We believe this method will still yield promising results and in later sections, we will also evaluate and compare both approaches.

1.2 Contributions

The contributions of this work can be summarized as follows:

(1) First to justify the different expectation between Video Frame Interpolation and Animation Frame Interpolation.
(2) Integrate the Global Motion Aggregation module [3] into Recurrent Flow Refinement Module of AnimeInterp [4].
(3) First to introduce the Style loss in Animation Frame Interpolation.

1.3 Thesis Structure

The overall structure of the thesis is organised as follows:

- Chapter 2 reviews the literature in the related field of studies ranging from Video Frame Interpolation to Sketch-based approaches.
- Chapter 3 describes the methodology of the entire computational model.
- Chapter 4 presents the experiments setup.
- In Chapter 5, we evaluate the results both quantitatively and qualitatively.
- We conclude and suggest directions for future work in Chapter 6.

CHAPTER 2

Background and Related Work

2.1 Perceptual Quality Metrics

Though humans are relatively proficient at comparing two images perceptually, the underlying processes are thought to be quite complicated. Even this seemingly straightforward task of comparing visual patterns poses a wide-open problem in computer vision. Not only because visual patterns are high-dimensional and highly correlated, but the notion of visual similarity is also often very subjective.

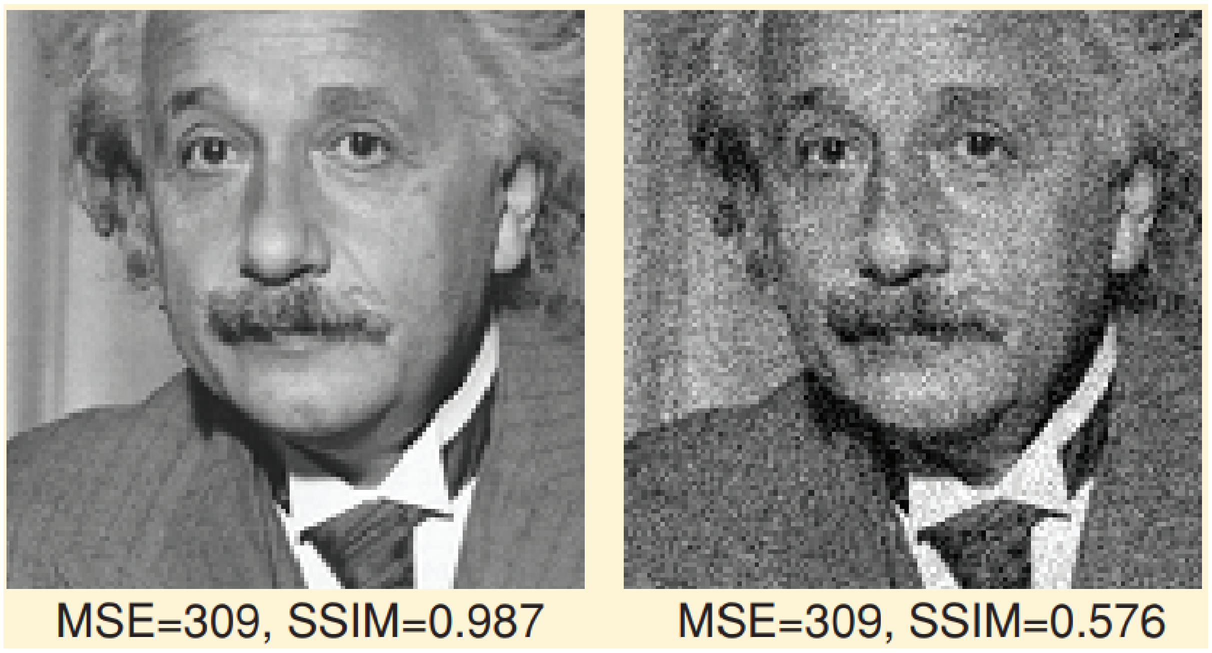


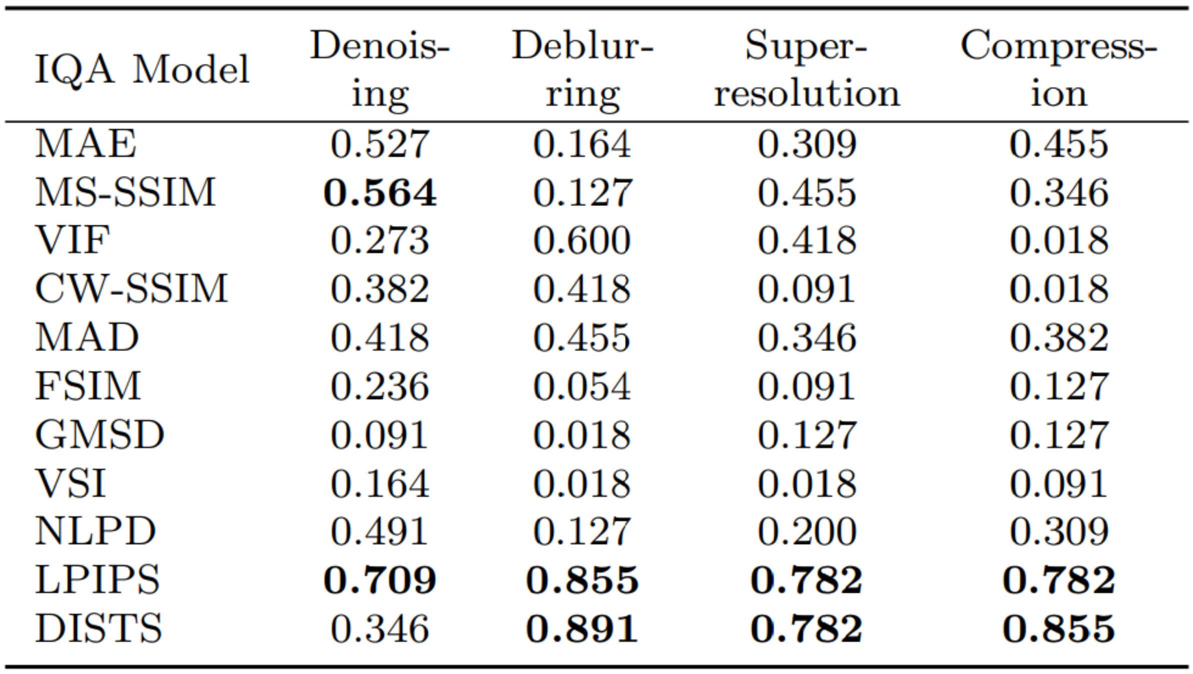
FIGURE 2.1: Mean Squared Error (hence of PSNR) Counter Example [1]

Classical pixel-wise measures such as Peak Signal-to-Noise Ratio (PSNR) are insufficient for structured outputs such as images due to the per-pixel independence. One typical example is Figure 2.1. What we need is to determine the "perceptual distance," which measures how similar two images are in a way that coincides with human judgment. Structural Similarity Index (SSIM) [8] is then proposed based on three aspects: brightness, contrast, and structure. However, [9] demonstrates the mathematical properties of SSIM and show that it is not adhering to properties of the human visual system. The inapplicability of PSNR and SSIM for the field of Image Super Resolution has also been discussed in other studies [10, 11].

According to [12], human judgments of similarity consists of three properties:

(1) Depend on high-order image structure
(2) Are context-dependent
(3) may not actually constitute a distance metric

With many direct approaches failing to generalise, Kim and Lee [13] use a CNN to predict visual similarity by training on low-level differences. The computer vision community has discovered that the internal activations of deep convolutional networks trained on image classification task, are often correspond to human perceptual judgments. Within [12], **Learned Perceptual Image Patch Similarity** (LPIPS) metric is also proposed and it has been widely used in various tasks such as Frame interpolation, Video deblurring, Super-resolution. According to comparison of various Image Quality Assessment (IQA) metrics [2] shown in Figure 2.2, LPIPS outperforms all the other methods.



IQA Model	Denois-ing	Deblur-ring	Super-resolution	Compress-ion
MAE	0.527	0.164	0.309	0.455
MS-SSIM	**0.564**	0.127	0.455	0.346
VIF	0.273	0.600	0.418	0.018
CW-SSIM	0.382	0.418	0.091	0.018
MAD	0.418	0.455	0.346	0.382
FSIM	0.236	0.054	0.091	0.127
GMSD	0.091	0.018	0.127	0.127
VSI	0.164	0.018	0.018	0.091
NLPD	0.491	0.127	0.200	0.309
LPIPS	**0.709**	**0.855**	**0.782**	**0.782**
DISTS	0.346	**0.891**	**0.782**	**0.855**

FIGURE 2.2: Comparison between metrics in different image processing tasks [2]

2.2 Optical Flow Estimation

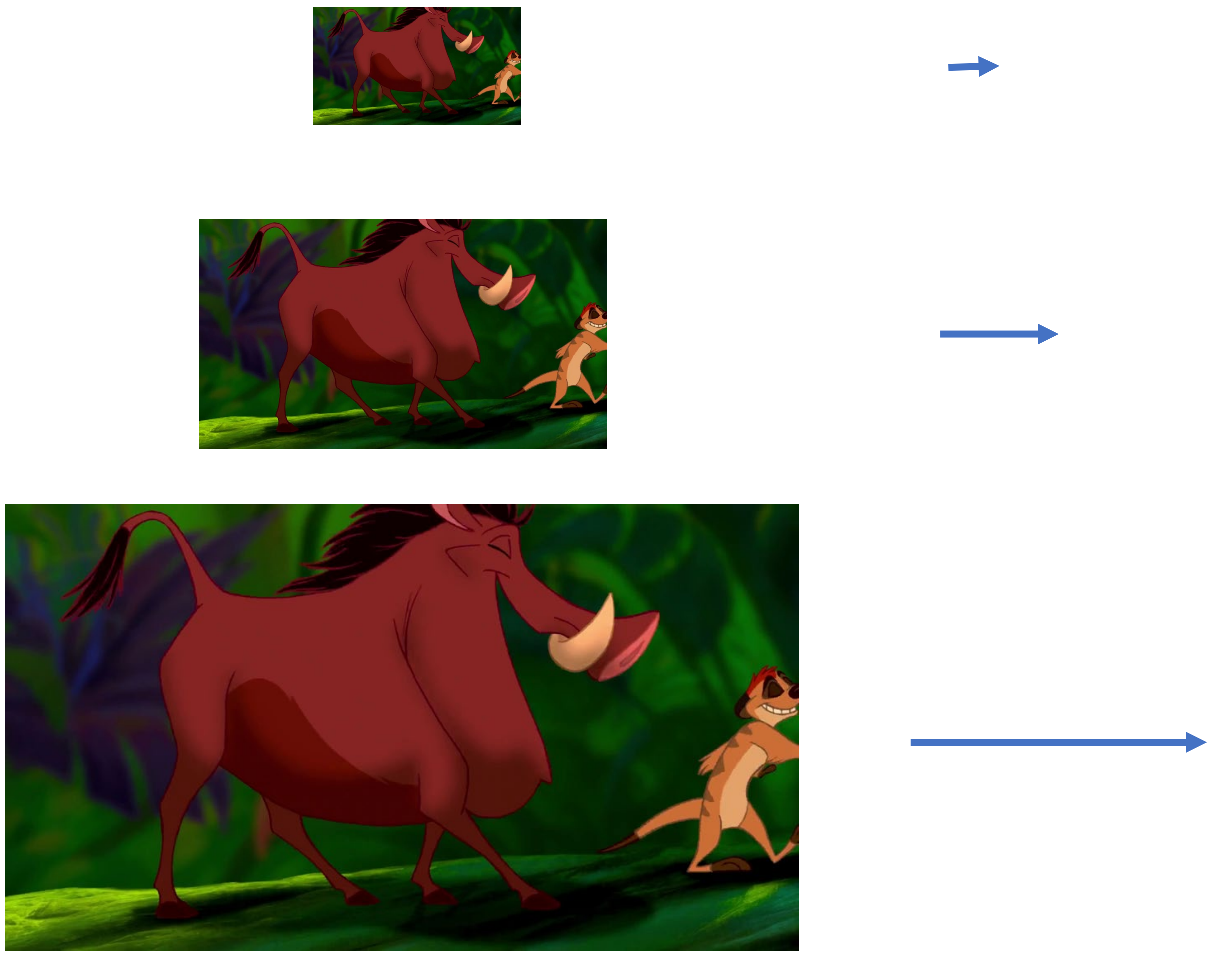


FIGURE 2.3: **Pyramidal Processing:** By down-sampling, a large motion becomes smaller.

Optical Flow Estimation is a fundamental procedure for various image and video processing tasks such as autonomous driving [14] and multi-view reconstruction [15]. Optical flow is the motion of objects between consecutive frames of the sequence, caused by the relative movement between both the object and camera. With the brightness constancy, a typical computation of optical flow is attained by estimating a dense motion field indicating the displacement of each pixel in consecutive images. And the reliability of the process becomes one of the main challenges in computer vision. Most top-performing methods adopt the energy minimisation approach introduced by the seminal work of Horn and Schunck [16]. However, operating on a complex energy function is computationally expensive for real-time applications. Inspired by the success of convolutional neural network (CNN), FlowNetS and FlowNetC are proposed [17] and show the possibility of direct optical flow estimation based on raw images. Followed by SpyNet [18] which utilises pyramidal processing to handle large motions in flow estimation. Pyramidal approach is an influential method to tackle large motion problem until today. (A sample illustration is shown in Figure 2.3) In 2018, PWC-Net [19] outperforms all previous models in both performance and runtime by revisiting the fundamental principles and utilising a harmonious combination of **P**yramidal processing, **W**arping and **C**ost volume. Teed and Deng proposed RAFT [20], an end-to-end differentiable model that leverages 4D cost volume and an iterative refinement unit with weights sharing, further improving the performance in major optical flow datasets.

2.2.1 Global Motion Aggregation

There are many factors that can make optical flow prediction a hard problem, including strong reflection, large motions, defocus blur, and texture less regions. Among these challenges, occlusion is one of the most difficult and under-explored. Similar to other challenges, occlusion violate the brightness constancy constraint [16]. Various solutions are proposed to compensate for the problem, while among all, the Global Motion Aggregation (GMA) [3] method has shown strong potential in motion estimation of animation frames.

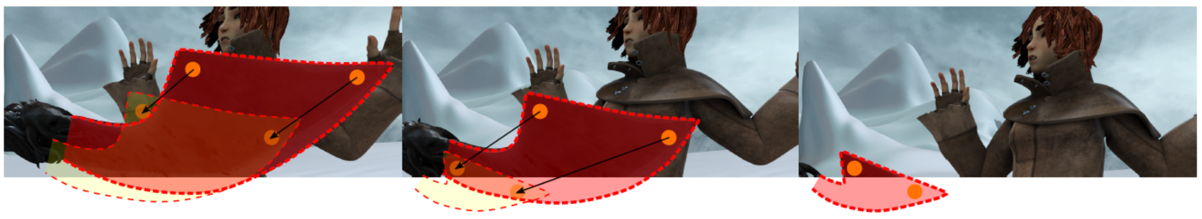


FIGURE 2.4: A typical occlusion scenario. [3]

When the local matching information is absent, the motion information has to be propagated from other pixels. While traditional approaches use convolutions to propagate the information, this method suffers from the drawback of the limited size of the receptive field as convolution is a local operation.

They propose a non-local approach based on the assumption that object motion is generally homogeneous across frames. Knowing what pixels are related to a pixel, or what object it belongs to, can be used to compensate for occlusions because the motion information of the non-occluded self-similar points can be propagated to the occluded points.

2.3 Video Frame Interpolation

Video Frame Interpolation (VFI) is an active area of research in computer vision with applications such as video post-processing and video restoration tasks. It aims to increase the frame rate of a video sequence by interpolating the intermediate frames between consecutive input frames.

A variety of recent work has been published regarding general video interpolation. Broadly, these works fall into following categories. Phase-based [21, 22] methods utilise phase information to learn the motion relationship for multiple video frame interpolation. Kernel-based [23, 24] methods formulate video frame interpolation as a spatially adaptive convolution whose kernel is generated using a CNN given the input frames but suffers from relatively large motion due to limited kernel size. CAIN [25] is an efficient flow-free method that employs the PixelShuffle operator and channel attention to capture the motion information implicitly. The most recent state-of-the-art has gravitated more towards flow-based methods, following corresponding advancements in optical flow estimation summarised in Section 2.2. DAIN [26] purpose a depth-aware video frame interpolation method that tightly integrates optical flow, local kernels, depth maps, and learnable hierarchical features for high-quality frame synthesis. Jiang *et al.* [27] propose SuperSlomo using the linear combination of the two bi-directional flows as an initial approximation of the intermediate flows and then refine them using U-Net. Xu *et al.* [28] propose QVI to exploit four consecutive frames and flow reversal filter to get the intermediate flows. Huang *et al.* [29] build a lightweight pipeline that achieves state-of-the-art performance while maintaining the conciseness of direct intermediate flow estimation. Flow-based methods can be further classified by forward [30] and backward [31] warping.

2.3.1 Animation Frame Interpolation (AFI)

Animation frame interpolation, as a sub-domain of VFI, is only recently defined in AnimeInterp [4], which is the only available model in this research area. According to AnimeInterp, the difference between animation videos and real-life videos is related to two unique characteristics:

- Cartoons comprise lines and smooth colour pieces. The smooth areas lacks textures and make it difficult to estimate accurate motions on animation videos.
- Cartoons express stories via exaggeration. Some of the motions are non-linear and extremely large.

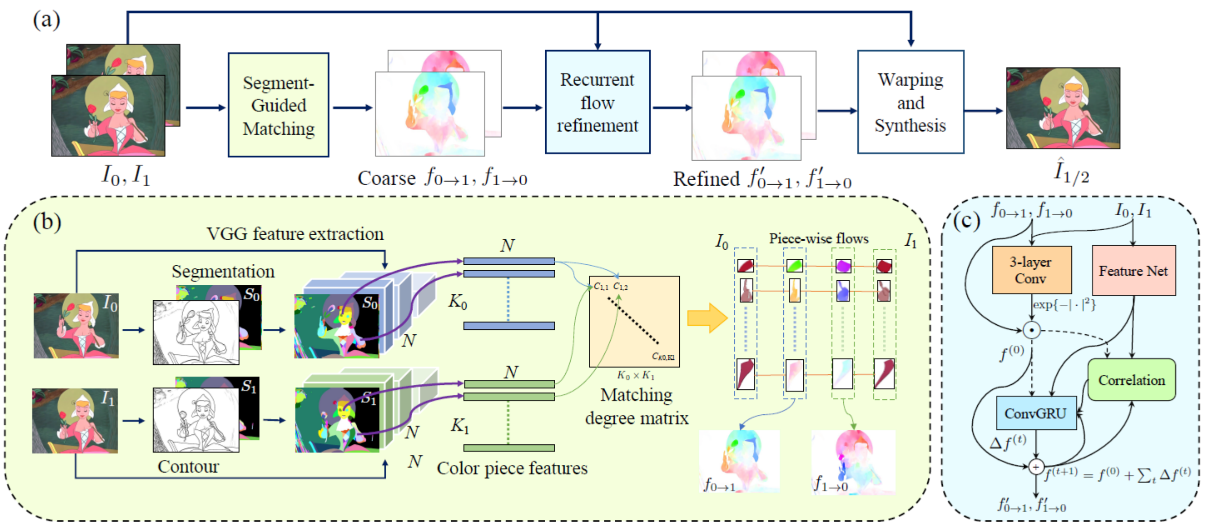


FIGURE 2.5: (a) The overall pipeline of AnimeInterp. (b) The inner structure of the SGM module. (c) The workflow of the RFR module. [4]

As shown in Figure 2.5, two dedicated modules are introduced to resolve aforementioned difficulties:

(1) **Segment-Guided Matching** resolves the "lack of textures" challenge by exploiting global matching among colour pieces that are piece-wise coherent.

(2) **Recurrent Flow Refinement** resolves the "non-linear and extremely large motion" challenge by recurrently refining the predictions using a transformer-like architecture.

Comprehensive experiments demonstrate the effectiveness of this model in interpolating animation data but there are still shortcomings. Mentioned by [32], limitations of the work consist of:

- Use of unscalable methods in ATD-12K data collection.
- Focus on pixel-perfect performance while having a loss and architecture resulting in blurriness and artifacts.
- Significantly slow due to inefficient segment-matching module in inference time.
- Perceptual quality is well evaluated quantitatively and qualitatively.

We re-produce AnimeInterp's results as a preliminary experiment for further quantitative and qualitative analysis refer to Section 5.1.1.

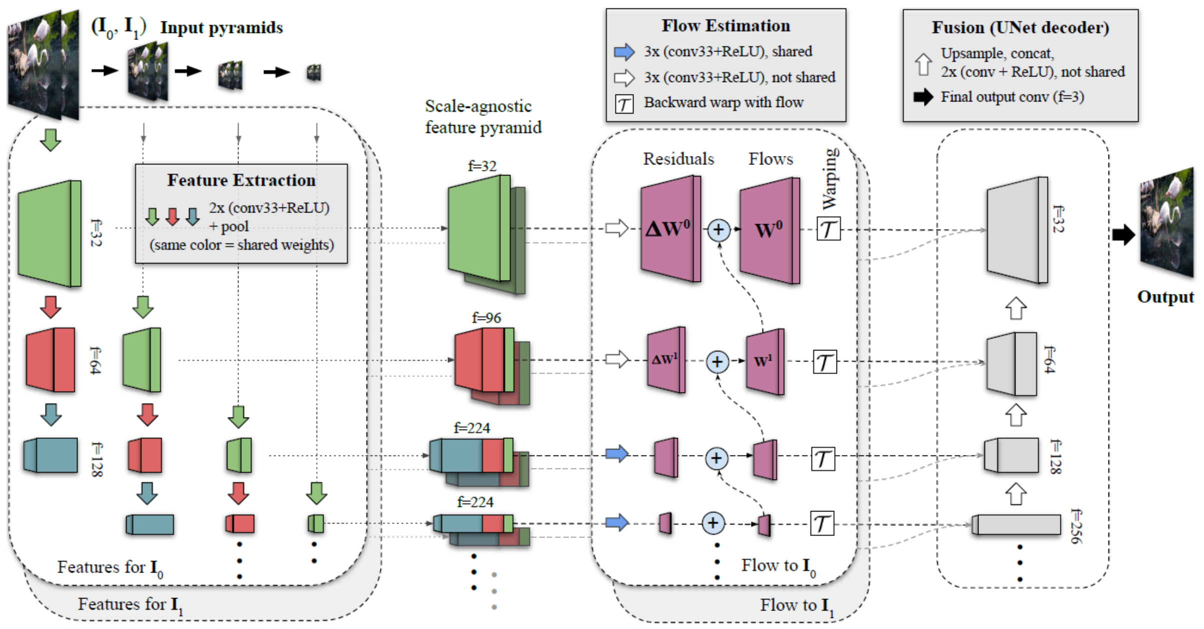


FIGURE 2.6: FILM Architecture Overview [5]

2.3.2 Frame Interpolation for Large Motion (FILM)

FILM is an elegant method that effectively tackle large motions. See in Figure 2.6, the contribution of the work can be summarised into:

- **End-to-end trainable.** Recent VFI methods use multiple networks to estimate optical flow or depth [26] and a separate network that is dedicated to frame synthesis. This is often complex and requires scarce optical flow or depth ground truth. FILM proposed **a unified, single-stage model for large motion frame interpolation**. It can be directly trained from frame triplets and doesn't require additional optical flow.
- **Effective feature extraction with weights sharing.** FILM has a scale-agnostic bi-directional motion estimation module with a multi-scale feature extractor [33] that shares the same weights at each pyramid level.
- **Well-designed Loss Function.** The gram matrix loss [34, 35, 36] is used to generated crisp and pleasing intermediate frames.

As a result, this is the first work that utilises shared feature extraction and Gram matrix loss for frame interpolation. FILM outperforms other methods and handles large motion well. However, as a limitation, FILM produces unnatural deformations when the in-between motion is extreme. While the resulting videos are still appealing, the subtle movements may not look natural. The single-model, weight-shared, and Gram matrix loss methods can all be used for interpolating animation frames. We also run some preliminary experiments on FILM refer to Section 5.1.2.

CHAPTER 3

Method

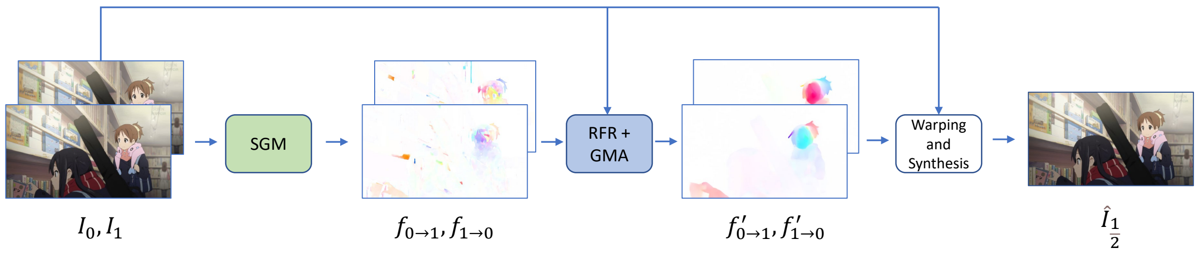


FIGURE 3.1: The overall pipeline of AnimeInterp+GMA.

The overview of the proposed network AnimeInterp+GMA is shown in Figure 3.1. Given two input frames (I_0, I_1), the coarse flows $f_{0\to1}$ and $f_{1\to0}$ between I_0 and I_1 in both directions are estimated through the AnimeInterp's SGM module [4] in Section 3.1. Then the coarse flows are set as the initialisation and fed into the RFR + GMA module. The flows are gradually refined and finally produced as the fine flows $f'_{0\to1}$ and $f'_{1\to0}$ in Section 3.2. Lastly, based on $f'_{0\to1}$ and $f'_{1\to0}$, the network trained with a ground-truth I_t warps I_0 and I_1 to synthesise a mid-frame $\hat{I}_t$ with time $t \in (0, 1)$. During training and evaluation, We set $t = 0.5$ to comply with the training triplets but can predict more in-between frames by recursively invoking the model.

3.1 Segment-Guided Matching (SGM)

For classical 2D animations, scenes and objects are mostly contoured with explicit outlines while each enclosed segment is filled with one single colour. Despite the large motion of moving objects, the colours are likely to remain stable from one frame to another. In contrast to natural video frames where the objects contain complex patterns of texture and this generally will not work well to establish a segment-to-segment matching, but in animation frames, those can be strong clues to find appropriate semantic matching for the colour pieces. SGM module leverages the clues to estimate the piece-wise motions and then generates coarse optical flows for later handling procedures. The procedure is illustrated in Figure 3.2.

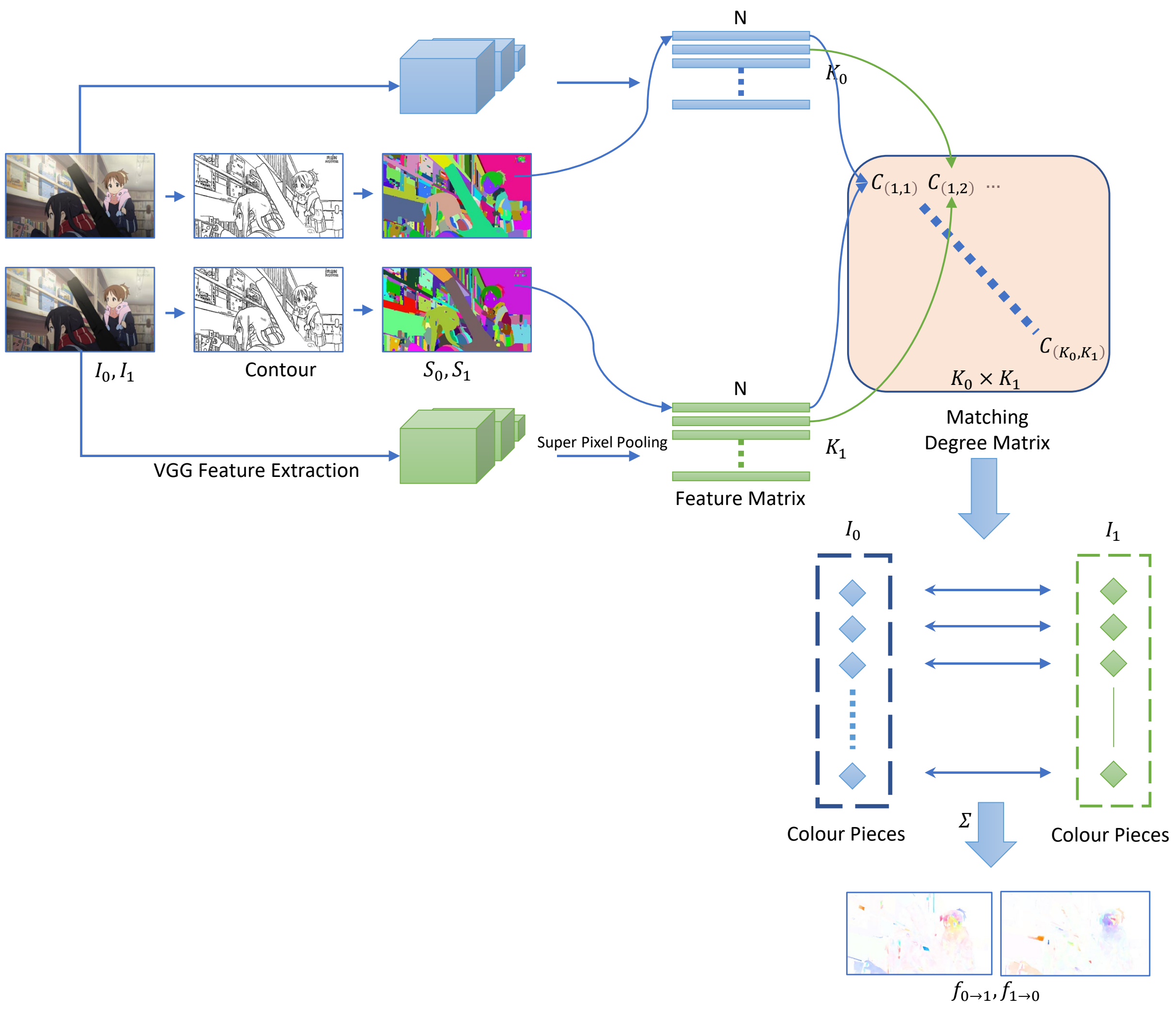


FIGURE 3.2: The inner structure of the SGM module.

3.1.1 Colour Piece Segmentation

Following Zhang *et al.*'s work [37], the Laplacian filter is adopted to contour the input frames. Then the "trapped-ball" algorithm is applied to fill the contours and generate colour pieces. Up to this stage, a segmentation map $S \in \mathbb{N}^{H \times W}$ is obtained, where pixels of each colour piece is labeled by an identity number. We notate the segmentation map of I_0 and I_1 as S_0 and S_1, while $S_0(i)$ represents the pixels in the i^{th} colour piece of I_0 and similar for $S_1(i)$.

3.1.2 Feature Extraction

The input frames I_0 and I_1 are also fed into a pre-trained VGG-19 model [38]. The features located at *relu1_2, relu2_2, relu3_4, relu4_4* layers are extracted and assembled per segment using super-pixel pooling proposed in [39]. After the pooling, we can attain a $K \times N$ feature matrix, where K is the number of colour pieces and N is the dimension of the feature vector that is correspond to a colour piece. We use F_0 and F_1 to denote the feature matrices for I_0 and I_1.

3.1.3 Colour Piece Matching

At this stage, we can use F_0 and F_1 to estimate a consistent mapping between colour pieces from frame I_0 and I_1. A forward map $\mathcal{M}_{0\to 1}$ and a backward map $\mathcal{M}_{1\to 0}$ can then be predicted where a map $\mathcal{M}_{i\to j}(n)$ indicates the maximum likelihood for the n^{th} colour piece in frame i correspond to the colour piece in frame j. To measure the likelihood between two colour pieces i, j ($i \in S_0, j \in S_1$), an affinity metric $\mathcal{A}$ is computed using F_0 and F_1:

$$\mathcal{A}(i,j) = \sum_{n}^{N} \min\left(\tilde{F}_0(i,n), \tilde{F}_1(j,n)\right) \tag{3.1}$$

where $\tilde{F}_0(i,n) = \frac{F_0(i,n)}{\sum_n F_0(i,n)}$ is the normalised feature of F_0. The same also applies to $\tilde{F}_1(i,n)$. This affinity metric measures the similarities of all pairs (i,j) globally.

To avoid potential outliers, two constraint penalties namely the *distance penalty* and the *size penalty* are exploited.

Firstly, it is assumed that **the displacement between two matching colour pieces is not overly large**. Hence, the *distance penalty* is defined as the ratio of the distance between the centroids of two colour pieces and the diagonal length of the image:

$$\mathcal{L}_{dist}(i,j) = \frac{\mid P_0(i) - P_1(j) \mid}{\sqrt{H^2 + W^2}} \tag{3.2}$$

where $P_0(i)$ and $P_1(j)$ represent the centroids of $S_0(i)$ and $S_1(j)$, $|\cdot|$ denotes the distance between two centroids. This penalty is only applied to the matching with the displacement larger than **15% of the diagonal length of the image**.

Secondly, it is assumed that **the sizes of matched pieces should be similar**. The *size penalty* is defined as:

$$\mathcal{L}_{size}(i,j) = \left| \frac{|S_0(i)| - |S_1(j)|}{H \times W} \right| \tag{3.3}$$

where $|\cdot|$ denotes the number of pixels in a cluster.

Combining all the items above, a matching degree matrix $\mathcal{C}$ can be computed as:

$$\mathcal{C} = \mathcal{A} - \lambda_{dist}\mathcal{L}_{dist} - \lambda_{size}\mathcal{L}_{size} \tag{3.4}$$

where λ_{dist} and λ_{size} are set to **0.2** and **0.05** in the implementation.

For each pair $(i \in S_0, j \in S_1)$, the matching degree matrix $\mathcal{C}(i,j)$ indicates the likelihood. The forward map $\mathcal{M}_{0\to1}$ and backward map $\mathcal{M}_{1\to0}$ can then be found as:

$$\mathcal{M}_{0\to1}(i) = \arg\max_j(\mathcal{C}(i,j)), \mathcal{M}_{1\to0}(j) = \arg\max_i(\mathcal{C}(i,j)) \tag{3.5}$$

where for the i^{th} colour piece in S_0, the most likely matching piece in S_1 is the one with maximum matching degree, and vice versa.

3.1.4 Flow Generation

After attaining $\mathcal{M}_{0\to1}$ and $\mathcal{M}_{1\to0}$, we can use them to predict dense bidirectional optical flows $f_{0\to1}$ and $f_{1\to0}$. Here we only describe the procedure to compute $f_{0\to1}$ since $f_{1\to0}$ can be computed by reversing the mapping order.

For each matched pair (i, j) where $j = \mathcal{M}_{0\to1}(i)$, we first compute the shift base (displacement between the centroids):

$$f_c^i = P_1(j) - P_0(i) \tag{3.6}$$

Then we compute the local deformation $f_d^i(u,v)$ by variational optimisation:

$$E(f_d^i(x)) = \int |I_1^j(x + f_c^i(x) + f_d^i(x)) - I_0^i(x)|dx + \int (|\nabla u(x)|^2 + \nabla |v(x)|^2)dx \tag{3.7}$$

where I_0^i represents a masked I_0 where pixels not belonging to i^{th} colour piece are set to 0. The same also applies to I_1^j.

The optical flow for pixels in i^{th} colour piece is then $f^i_{0\to1} = f^i_d + f^i_c$. And the final flow for the whole image can be computed by adding up all piece-wise flows together:

$$f_{0\to1} = \sum_i f^i_{0\to1} \tag{3.8}$$

To further avoid outliers, we abandon the flow of i^{th} piece (set $f^i_{0\to1}$ to zero) if it does not satisfy the mutual consistency $\mathcal{M}_{1\to0}(\mathcal{M}_{0\to1}(i)) \neq i$. This will prevent the subsequent flow refinement step to be misled by the low-confidence matching.

3.2 Recurrent Flow Refinement and Global Motion Aggregation Network (RFR+GMA)

In this section, we refine the coarse optical flows $f_{0\to1}$ and $f_{1\to0}$ computed from the SGM into finer views $f'_{0\to1}$ and $f'_{1\to0}$ through a deep recurrent refinement network. It is modified from AnimeInterp's RFR network [4] by embedding the Global Motion Aggregation module [3]. The workflow is shown in Figure 3.3.

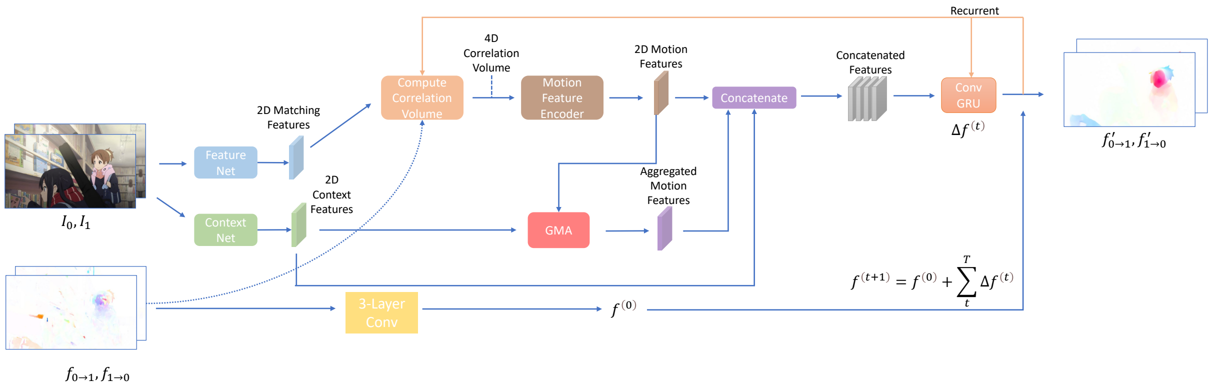


FIGURE 3.3: The workflow of the RFR+GMA module.

3.2.1 Global Motion Aggregation Module

For the GMA module, we utilise the standard implementation without using a 2D relative positional embedding vector, which achieved the best result according to [3]. The structure can be viewed in Figure 3.4.

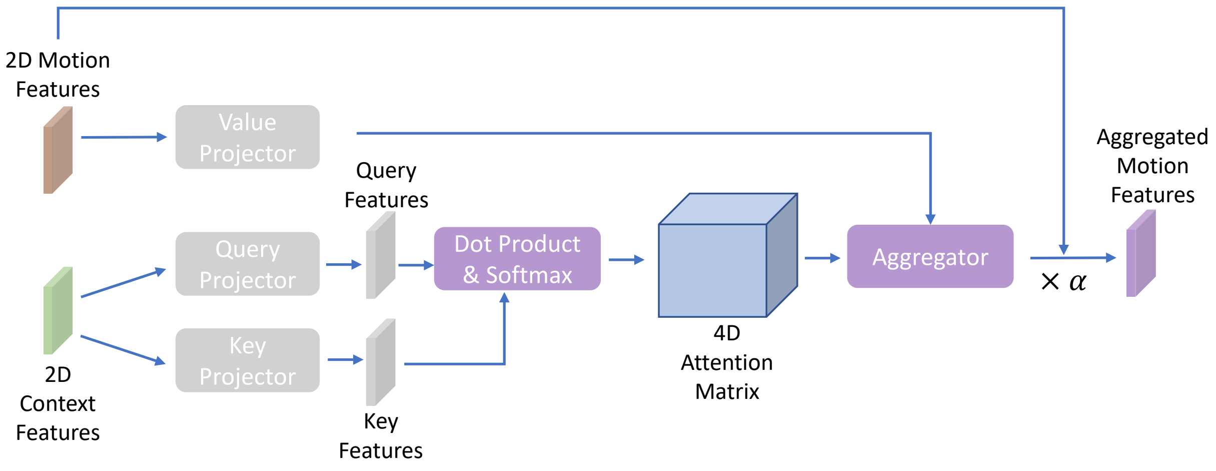


FIGURE 3.4: The inner structure of the GMA module.

Let $x \in \mathbb{R}^{N \times D_c}$ denotes the context (appearance) features and $y \in \mathbb{R}^{N \times D_m}$ denote the motion features, where H, W, D are the height,width, channel dimension of the feature map, and $N = HW$. The i^{th} feature vector is denoted as $x_i \in \mathbb{R}^{D_c}$.

The GMA module computes the feature vector update as an attention-weighted sum of the projected motion features. The aggregated motion features are given by:

$$\hat{y}_i = y_i + \alpha \sum_{j=1}^{N} f(\theta(x_i), \phi(x_j))\sigma(y_j) \tag{3.9}$$

where α is a learned scalar parameter initialised to 0. θ, ϕ and σ are the projection functions for the query, key and value vectors given by:

$$\theta(x_i) = W_{qry} x_i, \tag{3.10}$$

$$\phi(x_i) = W_{key} x_i, \tag{3.11}$$

$$\sigma(x_i) = W_{val} y_i, \tag{3.12}$$

where W_{qry}, $W_{key} \in \mathbb{R}^{D_c \times D_c}$ and $W_{val} \in \mathbb{R}^{D_m \times D_m}$. All of them are learnable parameters.

The final output is a concatenation of three feature maps $[y|\hat{y}|x]$ which will then be passed to a convolutional GRU [40] to decode and obtain the corresponding residual flow.

f is a similarity attention function given by:

$$f(a_i, b_i) = \frac{\exp\,(a_i^T b_j/\sqrt{D})}{\sum_{j=1}^{N} \exp\,(a_i^T b_j/\sqrt{D})} \tag{3.13}$$

3.2.2 Recurrent Flow Refinement Module

According to [4], there are two main motivations for introducing the RFR module:

(1) In Section 3.1, we abandon non-robust colour-piece pairs, which leaves null flow values in some locations. RFR is capable of remedying those with valid flows.

(2) The SGM module is beneficial for large displacements but less effective to predict precise deformation especially for the non-linear and exaggerated motions in animation frames. Hence, a recurrent refining approach can be the complement to the coarse piece-wise approach.

Moreover, the original AnimeInterp's RFR architecture is also modified from RAFT [41]. Since both RFR and GMA are extended from RAFT, it is possible that we integrate GMA into RFR to improve its robustness towards occlusion problem.

3.2.3 Overall Workflow

For the sake of brevity, we will demonstrate the process to attain $f'_{0\to1}$ from $f_{0\to1}$ since $f'_{1\to0}$ can be attained in similar fashion.

For initialisation, a **pixel-wise** confidence map g is learned to mask the unreliable values away from the coarse flow $f_{0\to1}$ via a three-layer CNN (3 layer Conv). The base flow $f^{(0)}_{0\to1}$ is now attained by multiplying $f_{0\to1}$ with $exp\{-g^2\}$.

For recurrent residues $\Delta f^{(t)}_{0\to1}$, they are learned via the GRU [40]:

$$\Delta f^{(t)}_{0\to1} = ConvGRU(f^{(t)}_{0\to1}, \mathcal{F}_0, corr(\mathcal{F}_0, \mathcal{F}^{(t)}_{1\to0}), \mathcal{C}_0) \tag{3.14}$$

where $\mathcal{F}_0$ and $\mathcal{F}_1$ are frame features extracted by the Feature Net in Figure 3.3, $\mathcal{F}^{(t)}_{1\to0}$ is bilinearly sampled from $\mathcal{F}_1$ with optical flow $f^{(t)}_{0\to1}$, $corr(\cdot,\cdot)$ computes the correlation between two tensors, $\mathcal{C}$, is the context features extracted by the Context Net in Figure 3.3. In addition, within $ConvGRU()$ operation, note that $corr(\mathcal{F}_0, \mathcal{F}^{(t)}_{1\to0})$ produces a 4D Correlation Volume and it is passed to the Motion Feature Decoder to generate the motion features y mentioned in Section 3.2.1. Then the context features $\mathcal{C}$ as x and y are passed to the GMA module to generate $[y|\hat{y}|x]$ that will eventually passed back to the GRU to aggregate with other inputs and form $\Delta f^{(t)}_{0\to1}$.

Learned residues are recurrently accumulated to update the base flow. The optical flow refined after T iterations is shown as:

$$f_{0\to1}^{(T)} = f_{0\to1}^{(0)} + \sum_{t=0}^{T-1} \Delta f_{0\to1}^{(t)} \tag{3.15}$$

while the fine flow $f'_{0\to1}$ is the output of the last iteration.

3.3 Frame Warping and Synthesis

To generate the intermediate frame with refined flow $f'_{0\to1}$ and $f'_{1\to0}$, the splatting and synthesis strategy of SoftSplat [30] is adopted. Through a multi-scale CNN, features are extracted from I_0 and I_1 and then splatted via forward warping to the center position.

For instance, I_0 is splatted to $I_{0\to\frac{1}{2}}$ as:

$$I_{0\to\frac{1}{2}}(x + \frac{f_{0\to1}(x)}{2}) = I_0(x) \tag{3.16}$$

Finally, all warped frames and features are fed into a GridNet [42] with three scale levels to synthesise the target frame $\hat{I}_{\frac{1}{2}}$.

3.4 Loss Functions

We use only image synthesis losses to supervise final outputs of the network and it consists of a combination of three terms. The inspiration comes from FILM [5].

3.4.1 L1 Loss

L1 reconstruction loss is used to measure the **pixel-wise RGB difference** between the predicted frame $\hat{I}_t$ and the ground-truth I_t. It is defined as:

$$\mathcal{L}_1 = |\hat{I}_t - I_t|_1 \tag{3.17}$$

Training through $\mathcal{L}_1$ loss can yield interpolation results that score well on benchmarks but typically produce blurry outputs [5, 32]. This part will be further discussed in Section 5.1.

3.4.2 VGG Loss

To enhance the image details, a perceptual loss using the L1 norm of the VGG-19 high-level feature representation [38] is given by:

$$\mathcal{L}_{VGG} = \frac{1}{L}\sum_{l=1}^{L} \alpha_l |\Psi_l(\hat{I}_t) - \Psi_l(I_t)|_1 \tag{3.18}$$

where $\Psi_l(I_t) \in \mathbb{R}^{H\times W\times C}$ is the feature map from the l^{th} layer of a pre-trained ImageNet VGG-19 network for a frame $I_t \in \mathbb{R}^{H\times W\times 3}$. L is the number of the finer layers considered and α_l is an importance weight of the i^{th} layer.

$\mathcal{L}_{VGG}$ enforces structural similarity over a nearby region around each output pixel due to the receptive fields of each VGG layer, and it is proven to ease the blurry artifacts in many image synthesis tasks [43, 34, 44, 23, 45].

3.4.3 Gram Matrix Loss

Gram Matrix Loss is firstly discussed in [46] and further explained in [47]. It is the L2 norm of the auto-correlation of the VGG-19 high-level feature representations [38]:

$$\mathcal{L}_{Gram} = \frac{1}{L}\sum_{l=1}^{L} \alpha_l |M_l(\hat{I}_t) - M_l(I_t)|_2 \tag{3.19}$$

where $M_l(I_t) \in \mathbb{R}^{C\times C}$ is the Gram matrix of the interpolated frame at the l^{th} layer and it is defined as:

$$M_l(I_t) = (\Psi_l(I_t))^T(\Psi_l(I_t)) \tag{3.20}$$

According to [5], Gram Matrix Loss is **effective in boosting image sharpness and preserving details when inpainting disocclusions in sequences with large motion**. During the experiment, we followed similar setups as FILM, which uses five activations from VGG-19 namely *relu1_2, relu2_2, relu3_2, relu4_2, relu5_2* with corresponding weights of 0.3846, 0.2083, 0.2703, 0.1786, 6.6667, respectively.

3.4.4 Style Loss

The overall Style Loss is a combination of $\mathcal{L}_1$, $\mathcal{L}_{VGG}$, and $\mathcal{L}_{Gram}$:

$$\mathcal{L}_S = w_l\mathcal{L}_1 + w_{VGG}\mathcal{L}_{VGG} + w_{Gram}\mathcal{L}_{Gram} \tag{3.21}$$

with the weights $(w_l, w_{VGG}, w_{Gram}) = (1.0, 0.25, 40.0)$ which was used in training the last 1.5M epochs of FILM.

CHAPTER 4

Experiments

4.1 Project Constraint

Video Frame Interpolation requires an enormous amount of computing power as well as storage. Moreover, the training is likely to take more time than most the other deep learning tasks. Due to the limited computing resources, we have access to and the project timeframe, we have to call off the experiments when hitting a reasonable margin. Hence, the outputs mentioned in the below sections do not indicate the best performance and we will provide sufficient evidence to support that the model can perform better.

Also, as a result, the models are trained in a pause-resume manner to accumulate enough epochs, which made it significantly difficult to time the entire process. Hence, we are unable to provide precise training time information.

4.2 Dataset

Since Animation Video Interpolation is still at a very early stage, currently the only available dataset is **ATD-12K** [4] and this is the major dataset we use throughout the experiments.

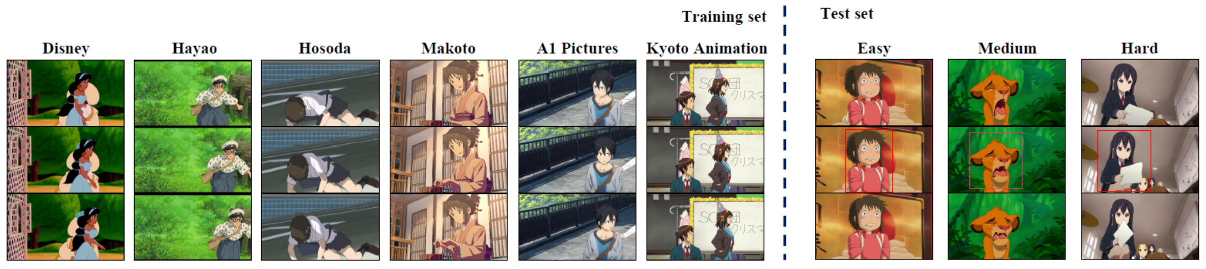


FIGURE 4.1: ATD-12K samples.

The set contains 12,000 animation frame triplets divided by 10,000 as the training set and the remaining 2,000 are for evaluation purpose. The cartoon clips are extracted from works by various producers including Disney, Hayao, Makoto, Hosoda, Kyoto Animation and A1 pictures. It provides triplets of resolution 1920×1080 and 1280×720 but all our models are trained and tested at 960×540.

In addition, the test set of ATD-12K comes with rich annotations including Difficulty Levels, Motion RoIs and Motion Tags.

(1) **Difficulty Levels.** The test set is divided into three difficulty levels namely **Easy**, **Medium**, and **Hard**. The criteria are based on the average magnitude of motions and the area of occlusions in each triplet.

(2) **Motion RoIs.** A bounding box is provided for the second image in each triplet to locate the foreground moving object. It is more suitable to examine the performance of interpolation methods on those Room of Interests as the motions of those regions have more impact on the visual quality.

(3) **Motion Tags.** For every triplet of the test set, the major motion is classified into two attributes General Motion Type (Camera Movement) and (Character) Behavior.

    General Motion Type includes translation, rotation, scaling, and deformation.

    Behavior includes speaking, walking, eating, sporting, etc.

In our experiment, we focus more on the overall quality of the whole interpolated frames but we will provide statistics related to difficulty levels and motion RoIs. Motion tags are not considered in the evaluation.

4.3 Evaluation Metrics

We use frame1 and frame3 of each triplet as input frames to produce the interpolated middle frame and then compare it against the original frame2 as the ground truth. We adopt common quantitative metrics: Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Image Metric (SSIM) [8], while higher PSNR and SSIM indicate better quality.

We also introduced LPIPS [12] to indicate the perceptual quality reflected by the interpolated frames because PSNR and SSIM, are simple, shallow functions, and fail to account for many nuances of human perception. To measure the LPIPS, we use the official LPIPS library with the AlexNet [48] backbone.

4.4 Experiments Setup

According to Table 5.3, we have overall four models to compare.

4.4.1 AnimeInterp - $\mathcal{L}_1$

This model requires no training and it is the original AnimeInterp model from [4]. We directly loaded the pre-trained weights to compare with other models, while the training by the authors consists of three phrases:

- They first pre-train the RFR network following [41] including:
  (1) FlyingChairs [49] dataset with Batch Size 8, Learning Rate 0.00025, Epochs 120000, Input Size 368x496, Optimiser Adam with Weight Decay 0.0001.
  (2) FlyingThings [50] dataset with Batch Size 5, Learning Rate 0.0001, Epochs 120000, Input Size 400x720, Optimiser Adam with Weight Decay 0.0001.
  (3) Sintel [51] dataset with Batch Size 5, Learning Rate 0.0001, Epochs 120000, Input Size 368x768, Optimiser Adam with Weight Decay 0.0001.
- They then fix the weights of RFR and train the rest parts of the network on QVI-960 [28] for 200 epochs using Adam without weight decay. Also, SGM flows are not input at this phase. The learning rate is initialised as 10^{-4} and decreases with a factor of 0.1 at the 100^{th} and 150^{th} epochs.
- Finally, they fine-tune the whole network for 50 epochs on the training set of ATD-12K with constant learning rate 10^{-6}. During fine-tuning, all the input frames are re-scaled into 960×540 and then randomly cropped into 380×380 with batch size 16. Data augmentation of stochastically flipping the images and reversing the triplet order are applied as well.

4.4.2 AnimeInterp - $\mathcal{L}_s$

This model is for comparison purposes and is initialised using the pre-trained weights from 4.4.1. We use the Style Loss function mentioned in 3.4.4 to fine-tune the model with Batch Size 8, Learning Rate 3×10^{-6} and Optimiser Adam [52] without weight decay. Data augmentation of random cropping, flipping, and order-reversing are also applied. Due to 4.1, we can only train for 35 epochs and the results in Table 5.3 are evaluated.

4.4.3 AnimeInterp + GMA - $\mathcal{L}_1$

Refer to 4.4.1, it is impossible for us to train the AnimeInterp + GMA model completely from scratch since training on the optical flow estimation network with tens of thousands of epochs requires a significant amount of time. The workaround we perform is by loading the pre-trained weights to initialise the model. We combine AnimeInterp's pre-trained weights along with GMA's weights pre-trained on Sintel [3] as the initialisation for our training. We will discuss the effect brought by this decision in later sections.

Then following 4.4.1, we freeze the weights of RFR + GMA module to train the rest parts of the network on QVI-960 [28] for 110 epochs using Adam without weight decay. SGM flows are not used. The learning rate is initialised as 10^{-4} and decreases with a factor of 0.1 at the 45^{th} and 90^{th} epochs.

Finally, we fine-tune the whole network for 195 epochs on ATD-12K's training set using $\mathcal{L}_1$ loss function with initial learning rate at 10^{-6} and increase to 3×10^{-6} at 161^{st} epoch due to lack of time. Fortunately, the loss still manages to decrease. Aforementioned data augmentation is also applied in the training with the aid of Adam optimiser without weight decay.

4.4.4 AnimeInterp + GMA - $\mathcal{L}_s$

Recall from FILM's training setup[5], the authors first train the network using weights $(w_l, w_{VGG}, w_{Gram}) = (1.0, 1.0, 0.0)$ mentioned in 3.4.4 because enabling $\mathcal{L}_{Gram}$ will prevent the model from converging. And the weights $(w_l, w_{VGG}, w_{Gram}) = (1.0, 0.25, 40.0)$ are enabled after 1.5M iterations. By having $\mathcal{L}_1$ loss oriented, and then switching to the style loss allows FILM to finally converge and provide astonishing results.

Following similar approach, AnimeInterp + GMA - $\mathcal{L}_s$ is actually the second stage of the training. We load the weights attained after 195 epochs trained on $\mathcal{L}_1$ loss in 4.4.3 and continue on training with $\mathcal{L}_s$ style loss. We set the learning rate to 3×10^{-6} to save time and maintain all the training parameters and data augmentation identical to previous training. We also train this model for 35 epochs and then evaluate the results in Table 5.3.

CHAPTER 5

Results

5.1 Preliminary Experiments

Our objective is to improve the performance of the state-of-the-art model AnimeInterp. Hence, we run several preliminary experiments to summarise the shortage of AnimeInterp and hence to find where we can improve upon it.

5.1.1 AnimeInterp Reproduction

We first use the pre-trained weights provided by [4] to re-run the original implementation of AnimeInterp on ATD-12K. The quantitative result is summarised in Table 5.3 as AnimeInterp - $\mathcal{L}_1$ since the weights are trained fully on $\mathcal{L}_1$ distance according to S. Li.

We also conduct visual comparison and summarise the results into several categories:

(1) **Out-of-frame Movements.** The foreground object contains parts that are only available in either one of the frame, which significantly increases the difficulty of optical flow estimation since the matching colour piece (for SGM) or pixel (for RFR) is completely out of the frame.

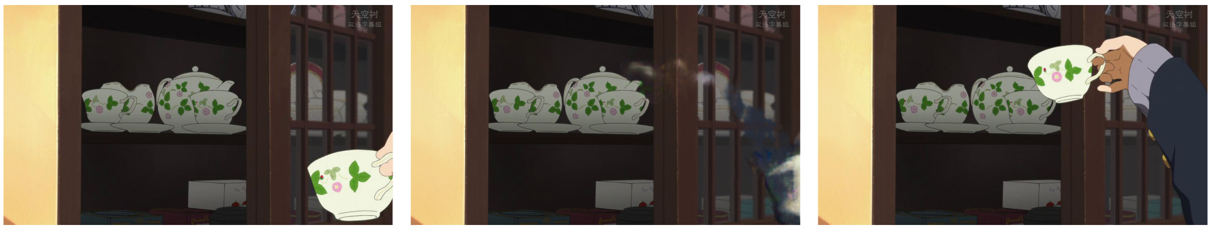


FIGURE 5.1: **Out-of-frame Movements:** Japan_v2_2_033283_s3 (The lowest PSNR among 2,000 triplets).

(2) **Blur.** The synthesised parts in the interpolated frame are blurry especially for the contours.

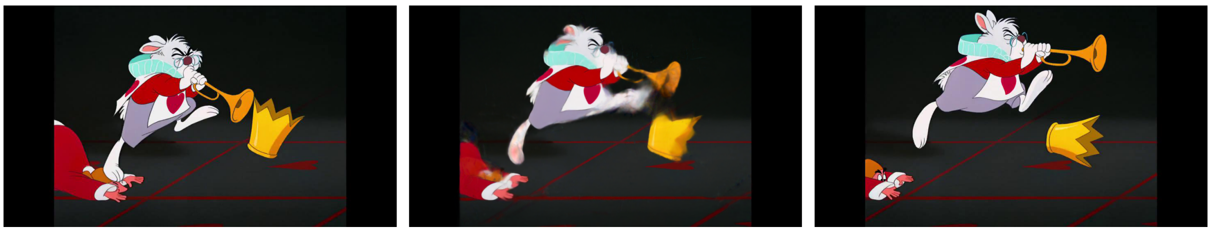


FIGURE 5.2: **Blur:** Disney_v4_20_043850_s1.

(3) **Colour Bleeding.** Weird colour stripes appear in regions filled with similar colours. It is more likely to occur when the region is relatively large (roughly larger than a quarter of a frame).

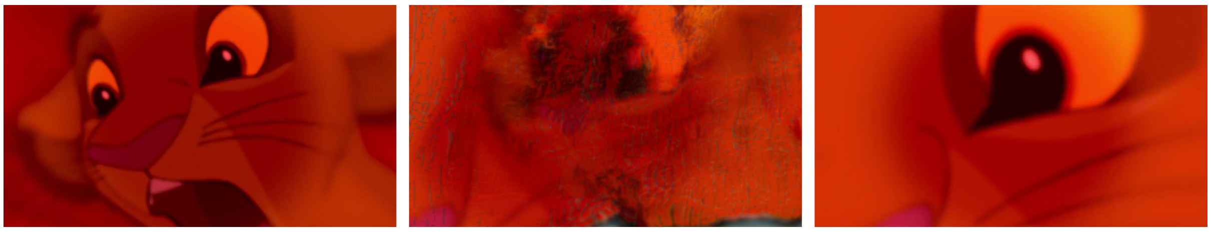


FIGURE 5.3: **Colour Bleeding:** Disney_v4_14_002999_s1 (The lowest SSIM among 2,000 triplets).

5.1.2 AnimeInterp vs. FILM

TABLE 5.1: **AnimeInterp vs. FILM on ATD-12K**. Note that the LPIPS computed is multiplied by 100 for readability.

Model	PSNR↑	SSIM↑	LPIPS↓
AnimeInterp	29.632	0.939	6.650
FILM	28.724	0.932	5.638

We use FILM [5] with pre-trained weights to interpolate the test set of ATD-12K to compare the results with outputs from Section 5.1.1. The quantitative summary is in Table 5.1.

TABLE 5.2: Δ PSNR and Δ SSIM between FILM and AnimeInterp

Maximum and Minimum	Values
Max Δ PSNR	11.868
Min Δ PSNR	-11.921
Max Δ SSIM	0.140
Min Δ SSIM	-0.114

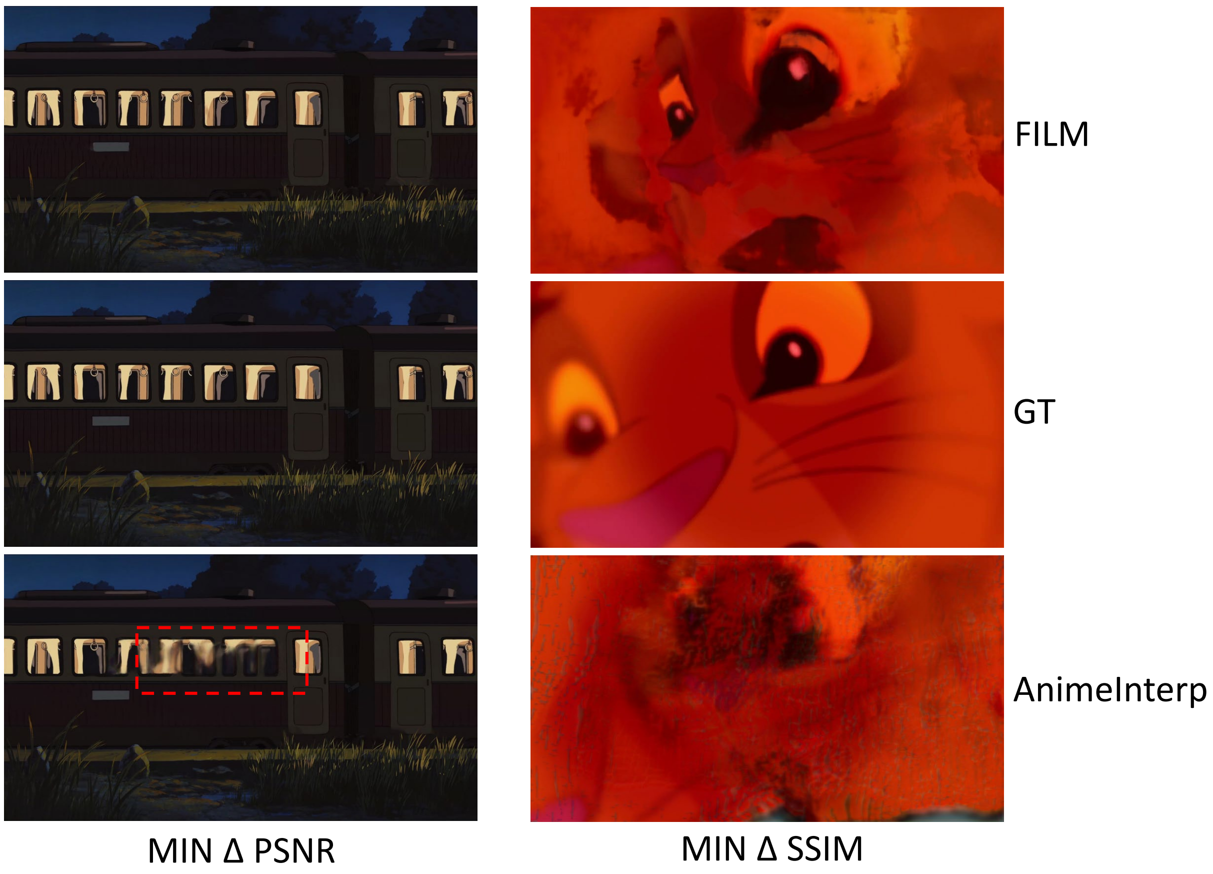


FIGURE 5.4: Min Δ PSNR and Min Δ SSIM

We introduce a measurement $\Delta = Value_{AnimeInterp} - Value_{FILM}$ to compare the results aiming to find samples which FILM outperforms AnimeInterp. The maximum and minimum are shown in Table 5.2. According to the result, the differences are very large which means there are frames that FILM tackles extremely well comparing to AnimeInterp (Figure 5.4). In the synthesised result of Min Δ PSNR, FILM has restored the train perfectly and with sharp outlines. Do note that the pre-trained weights of FILM are collected from models trained on natural video datasets [5] and it generalises so well on animation data.

5.2 Quantitative Evaluation

The quantitative results are shown in Table 5.3. Note that the LPIPS computed is multiplied by 100 for readability.

TABLE 5.3: Quantitative results on the test set of ATD-12K. The best and runner-up values are bold and underlined, respectively.

	Whole			RoI		Easy		Medium		Hard	
Method	LPIPS↓	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑	PSNR↑	SSIM↑
AnimeInterp - $\mathcal{L}_1$	6.650	**29.675**	$\underline{0.958}$	**26.265**	**0.910**	**31.864**	$\underline{0.971}$	**29.256**	$\underline{0.959}$	**27.068**	$\underline{0.939}$
AnimeInterp - $\mathcal{L}_s$	$\underline{5.836}$	28.446	0.949	25.107	0.892	30.887	0.967	28.102	0.951	25.400	0.922
AnimeInterp + GMA - $\mathcal{L}_1$	6.884	$\underline{29.613}$	**0.959**	$\underline{26.202}$	**0.910**	$\underline{31.838}$	**0.972**	$\underline{29.200}$	**0.960**	$\underline{26.949}$	**0.940**
AnimeInterp + GMA - $\mathcal{L}_s$	**5.233**	28.771	0.952	25.364	$\underline{0.895}$	31.024	0.968	28.327	0.953	26.101	0.928

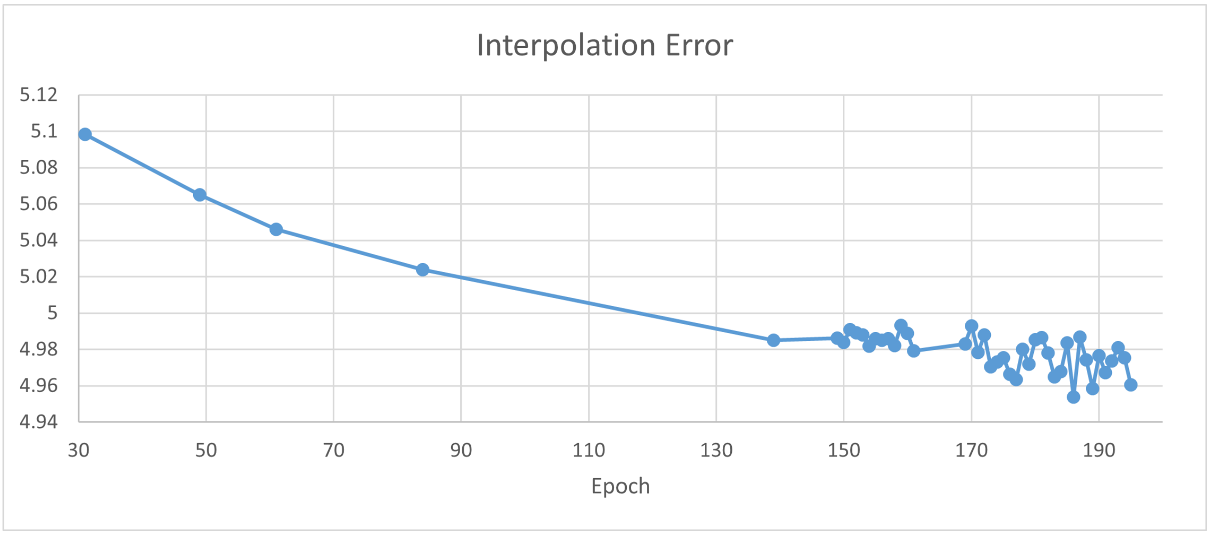


FIGURE 5.5: **Interpolation Error** of Evaluation on ATD-12K Test Set for AnimeInterp + GMA $\mathcal{L}_1$.

Based on PSNR, AnimeInterp - $\mathcal{L}_1$ outperforms all other methods in both Whole and RoI regions as well as all three difficulty levels. While for SSIM, AnimeInterp + GMA - $\mathcal{L}_1$ comes first in Whole, RoI and all three difficulty levels but the differences between the runner-up values are less noticeable. However, do note that AnimeInterp + GMA - $\mathcal{L}_1$ lacks of proper optical flow estimation training and it is not yet converged refer to Figure 5.5 where a decreasing trend in interpolation error and Figure 5.6 an increasing trend in PSNR can still be observed in 195^{th} epoch. We believe that if we are given sufficient time to train AnimeInterp + GMA - $\mathcal{L}_1$, it will eventually surpass AnimeInterp - $\mathcal{L}_1$. Moreover, if it is possible to train the full model from scratch starting from the flow estimation, the performance can be improved to a higher degree.

Similar to the results from FILM [5], after training using the Style loss, the model's performance on PSNR and SSIM tends to drop. AnimeInterp drops 1.229 dB in PSNR and 0.009 in SSIM but having LPIPS decreased by 0.814. AnimeInterp + GMA model drops 0.842 dB in PSNR and 0.007 in SSIM but having LPIPS decreased significantly by 1.661 which is nearly double the decrease of AnimeInterp.

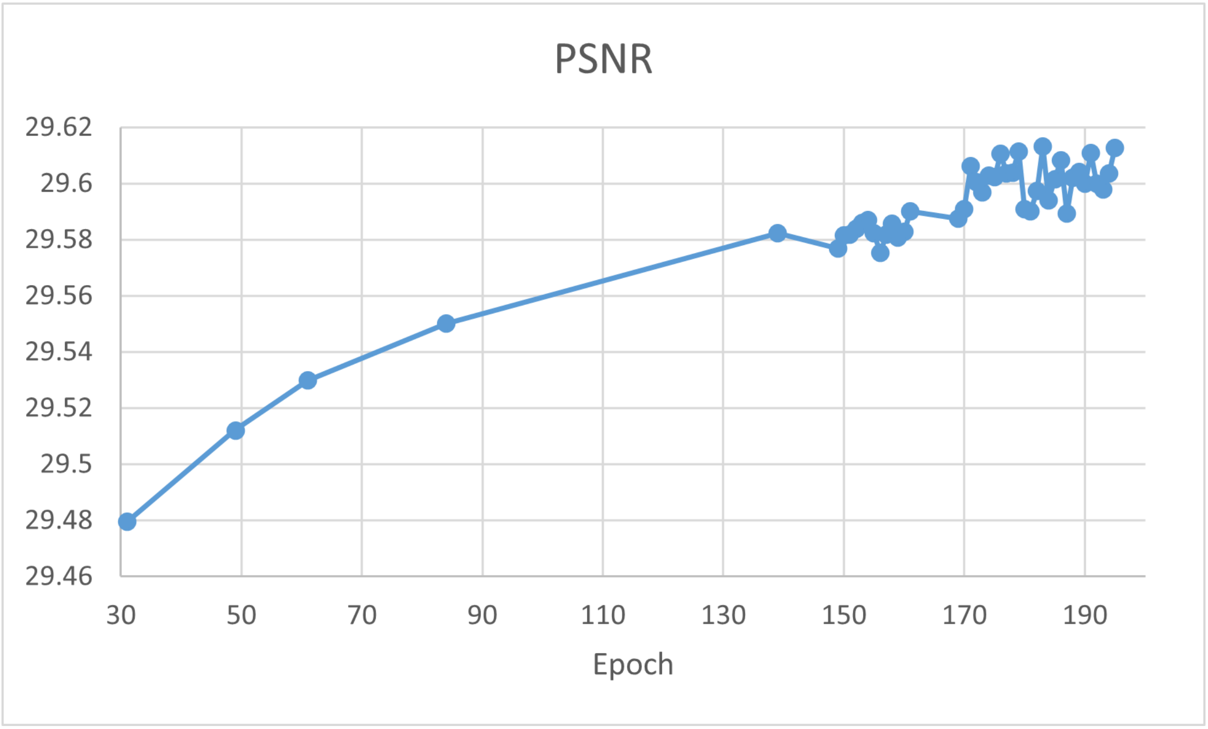


FIGURE 5.6: **PSNR** of Evaluation on ATD-12K Test Set for AnimeInterp + GMA $\mathcal{L}_1$.

Hence, AnimeInterp + GMA - $\mathcal{L}_s$ surpasses all the other methods in LPIPS. Note that both models complete their training on the style loss with exactly the same parameters and setup. It seems like AnimeInterp + GMA model is more compatible with the Style loss.

5.3 Qualitative Evaluation

5.3.1 Blurry Boundary in Estimated Flows

We have noticed that among most of the visualisation of refined flows synthesised by AnimeInterp + GMA, the boundaries are blurry and rough (see Figure 5.7). We believe this is due to lack of training in the optical flow estimation module (RFR + GMA).

5.3.2 Out-of-Frame Motion

We introduce the Global Motion Aggregation module into the AnimeInterp aiming to predict Out-of-frame movements. However, due to training constraints, the model performs less satisfactory quantitative results mentioned in 5.3. Figure 5.8 shows a typical case of Out-of-frame movements. The full

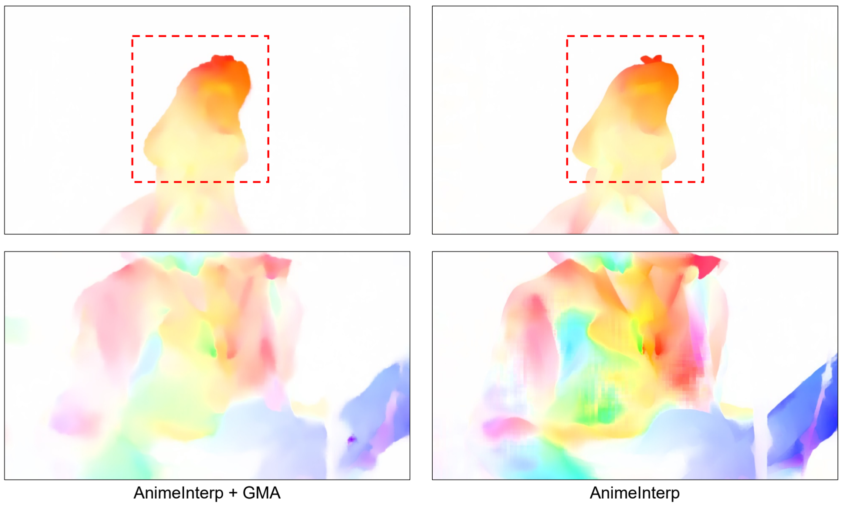


FIGURE 5.7: Visual Comparison between Refined Flows output from AnimeInterp and AnimeInterp + GMA

upper body of the left pedestrian is only available in F_3 and the right pedestrian's back of the head is also just available in F_3. AnimeInterp fails to re-create the left pedestrian in addition to the back of the head of the right pedestrian. However, by AnimeInterp + GMA, the major parts of both pedestrians are interpolated.

5.3.3 Sharpness

To evaluate the effectiveness of the Gram Matrix-based loss function in preserving image sharpness, we visually compare two models against its $\mathcal{L}_1$ version. As shown in Figure 5.9, models trained with $\mathcal{L}_s$ synthesises visually superior results, with crisp image details on the clothes and preserving the shapes of flowers located at the corners.

5.3.4 Detail Inpainting

We find that the interpolated frames produced by models trained with $\mathcal{L}_s$ are able to recover details that are totally ignored by models trained with $\mathcal{L}_1$. According to FILM [5], they have analysed the disocclusion inpainting. However, in Figure 5.10, the foreground dragon appears fully in all three frames, but we can still see the inpainting detail such that the dragon's claws are revealed by $\mathcal{L}_s$ models.

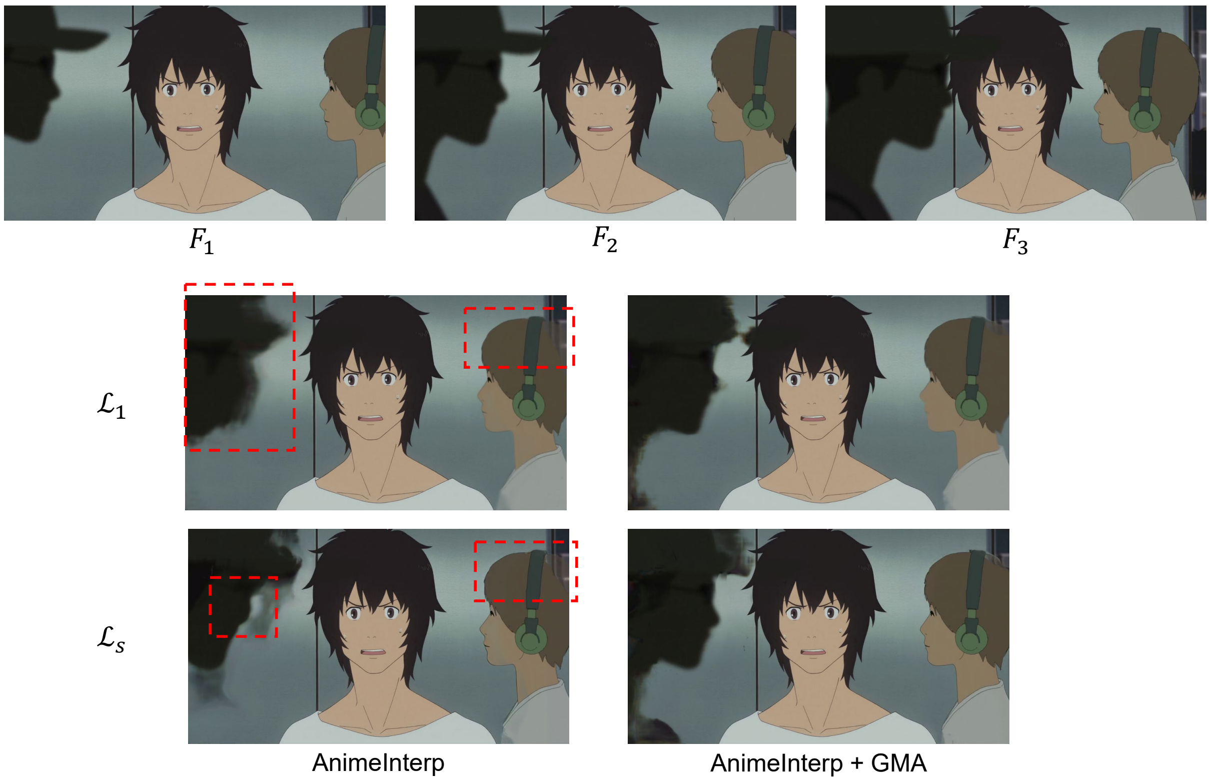


FIGURE 5.8: Visual Comparison along with Ground-truth on Japan_v2_0_104298_s2

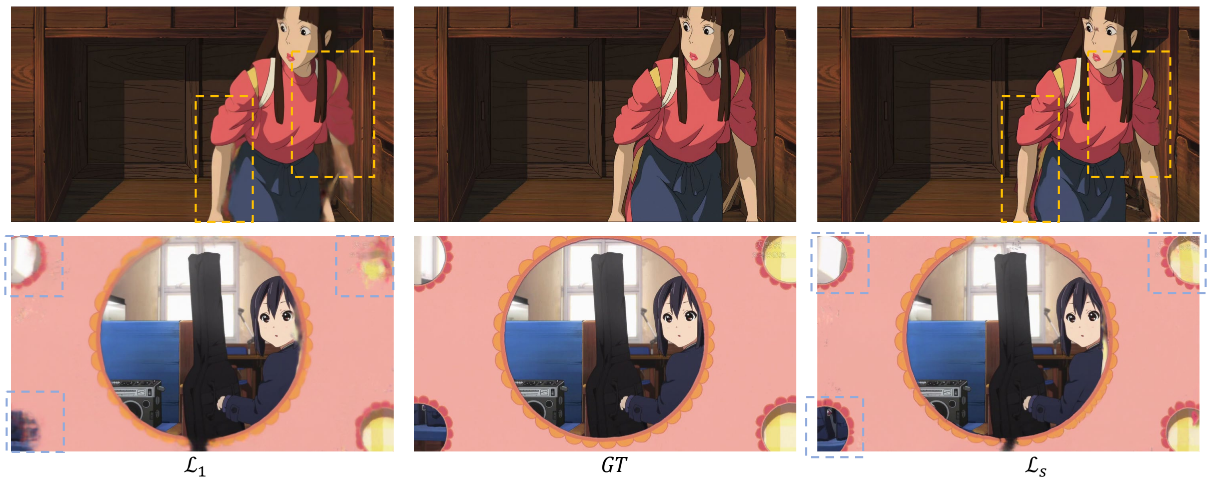


FIGURE 5.9: Visual Comparison along with Ground-truth on Japan_v2_3_129363_s3 and Japan_v2_2_008563_s1

We believe this is a new finding since we have not yet found any literature mentioning this behaviour and it can one of the key features of applying the Style loss in animation video frame interpolation. A more obvious example is shown in Figure 5.11.

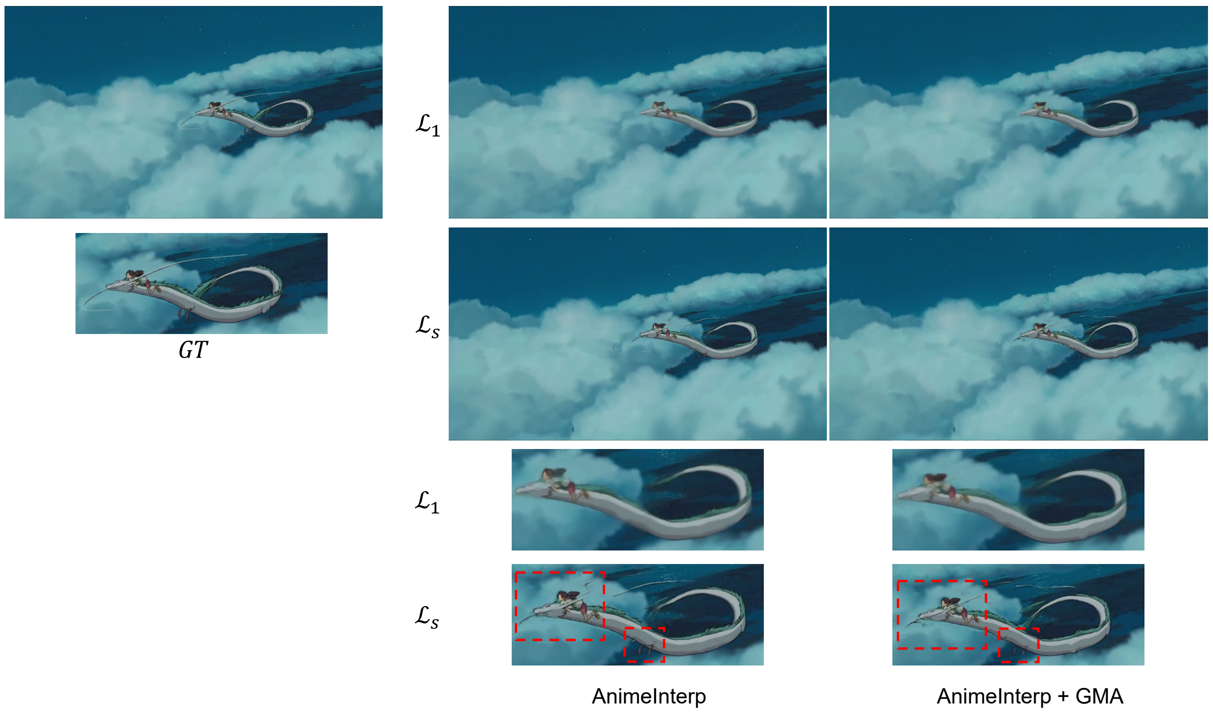


FIGURE 5.10: Visual Comparison along with Ground-truth on Japan_v2_3_162765_s2

5.4 Limitations

Apart from Section 4.1 which leads to an underfit optical flow estimation model, in some cases, the model will produce unnatural deformations. Moreover, the stability of the model is not guaranteed since the model is not trained thoroughly. It is possible to fail some relatively easy triplets.

SGM. Although we are not able to provide precise training time information, it is still necessary to mention the performance of the SGM module. To generate SGM flows for a triplet, on average it will take 3 minutes on our system. As a result of that, it is impossible to compute SGM in real-time and the only approach is through pre-computing. There are overall 10,000 training triplets in ATD-12K whose SGM flows are not provided. As SGM flows are necessary for training the model, we spend numerous amounts of time purely on pre-computing the SGM flows. According to the ablation study from AnimeInterp [4], disabling the SGM module results in a 0.14dB decrease in PSNR. We think it is a fair trade-off of removing the SGM module for future improvements. Moreover, the storage consumption of the SGM flows for the 10,000 training triplets is roughly 160 GB which is relatively larger compared to all the other public datasets.

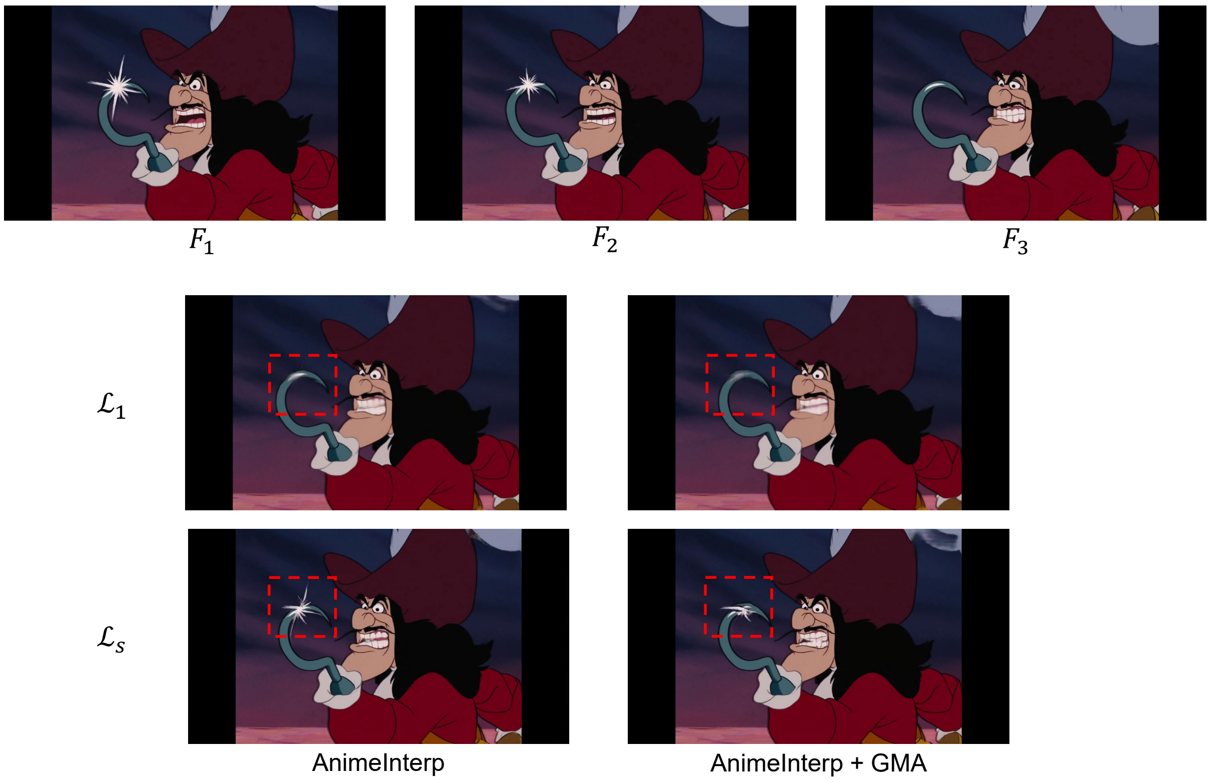


FIGURE 5.11: Visual Comparison along with Ground-truth on Disney_v4_21_041221_s2

CHAPTER 6

Conclusion and Future Work

6.1 Conclusion

This work proposes an improvement to the current style-of-the-art animation video frame interpolation model AnimeInterp [4] with the Global Motion Aggregation module [3] aiming to tackle occlusions. In addition, a different training objective (Style Loss) is applied to improve the perceptual quality of interpolated frames based on the justification of different intentions between Video Frame Interpolation and Animation Frame Interpolation.

Chapter 3 outlines the deep down methodology of the workflow from receiving the inputs to flow estimation, flow refinement and eventually frame synthesis. In Chapter 4, we describe the setup for the experiments including training and evaluation datasets, evaluation metrics, and training parameters. Finally, in Chapter 5, we discuss the results of both preliminary and final experiments. Our proposed model trained on L1 loss outperforms AnimeInterp in SSIM and has shown a potential increase in PSNR for future training. For the model trained with the Style loss, it surpasses AnimeInterp - $\mathcal{L}_s$ on PSNR, SSIM and LPIPS.

6.2 Future Work

Generative Adversarial Networks (GANs) for Animation Frame Interpolation. As GANs have achieved noticeable results in generating artistic content and style transfer, we believe the technique can also be applied to AFI. Moreover, we have shown the significant effect that a well-designed loss function can achieve while adversarial losses can also leverage. Also, GANs are proficient at generating pixels, which is possible to surpass the constraints that warped pixels from current models possess.

Further Distillation on ATD-12K Data. Visually, Western animation style (Videos are usually referred to as Cartoons) and Japanese animation style (Videos are usually referred to as Animes) are distinguishable. Since we have shown that Style Loss is a powerful aid, we believe that further designating models for a particular style can yield better results. As in this work, we train the model using data from both styles and it has to find a fit between Western and Japanese data.

The source code and pre-trained models are available at `https://github.com/Soooda/AFI`.

References

[1] Z. Wang and A. C. Bovik, "Mean squared error: Love it or leave it? a new look at signal fidelity measures," *IEEE Signal Processing Magazine*, vol. 26, no. 1, pp. 98–117, 2009.

[2] K. Ding, K. Ma, S. Wang, and E. P. Simoncelli, "Comparison of image quality models for optimization of image processing systems," *ArXiv*, vol. abs/2005.01338, 2020.

[3] S. Jiang, D. Campbell, Y. Lu, H. Li, and R. I. Hartley, "Learning to estimate hidden motions with global motion aggregation," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9752–9761, 2021.

[4] S. Li, S. Zhao, W. Yu, W. Sun, D. N. Metaxas, C. C. Loy, and Z. Liu, "Deep animation video interpolation in the wild," *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6583–6591, 2021.

[5] F. A. Reda, J. Kontkanen, E. Tabellion, D. Sun, C. Pantofaru, and B. Curless, "Film: Frame interpolation for large motion," *ArXiv*, vol. abs/2202.04901, 2022.

[6] D. Sýkora, J. Dingliana, and S. Collins, "LazyBrush: Flexible painting tool for hand-drawn cartoons," *Computer Graphics Forum*, vol. 28, no. 2, pp. 599–608, 2009.

[7] B. Whited, G. Noris, M. Simmons, R. W. Sumner, M. Gross, and J. Rossignac, "Betweenit: An interactive tool for tight inbetweening," *Computer Graphics Forum*, vol. 29, no. 2, pp. 605–614, 2010.

[8] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, pp. 600–612, 2004.

[9] J. Nilsson and T. Akenine-Möller, "Understanding ssim," *ArXiv*, vol. abs/2006.13846, 2020.

[10] A. R. Reibman, R. M. Bell, and S. Gray, "Quality assessment for super-resolution image enhancement," in *2006 International Conference on Image Processing*, pp. 2017–2020, 2006.

[11] A. R. Reibman and T. Schaper, "Subjective performance evaluation of super-resolution image enhancement,"

[12] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018.

[13] J. Kim and S. Lee, "Deep learning of human visual sensitivity in image quality assessment framework," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1969–1977, 2017.

[14] J. Janai, F. Güney, A. Behl, and A. Geiger, "Computer vision for autonomous vehicles: Problems, datasets and state-of-the-art," *Found. Trends Comput. Graph. Vis.*, vol. 12, pp. 1–308, 2020.

[15] C. Godard, P. Hedman, W. Li, and G. J. Brostow, "Multi-view reconstruction of highly specular surfaces in uncontrolled environments," *2015 International Conference on 3D Vision*, pp. 19–27, 2015.

[16] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artif. Intell.*, vol. 17, pp. 185–203, 1981.

[17] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 2758–2766, 2015.

[18] A. Ranjan and M. J. Black, "Optical flow estimation using a spatial pyramid network," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2720–2729, 2017.

[19] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8934–8943, 2018.

[20] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," *ArXiv*, vol. abs/2003.12039, 2020.

[21] S. Meyer, O. Wang, H. Zimmer, M. Grosse, and A. Sorkine-Hornung, "Phase-based frame interpolation for video," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1410–1418, 2015.

[22] S. Meyer, A. Djelouah, B. McWilliams, A. Sorkine-Hornung, M. H. Gross, and C. Schroers, "Phasenet for video frame interpolation," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 498–507, 2018.

[23] S. Niklaus, L. Mai, and F. Liu, "Video frame interpolation via adaptive convolution," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2270–2279, 2017.

[24] S. Niklaus, L. Mai, and F. Liu, "Video frame interpolation via adaptive separable convolution," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 261–270, 2017.

[25] M. Choi, H. Kim, B. Han, N. Xu, and K. M. Lee, "Channel attention is all you need for video frame interpolation," in *AAAI*, 2020.

[26] W. Bao, W.-S. Lai, C. Ma, X. Zhang, Z. Gao, and M.-H. Yang, "Depth-aware video frame interpolation," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3698–3707, 2019.

[27] H. Jiang, D. Sun, V. Jampani, M.-H. Yang, E. G. Learned-Miller, and J. Kautz, "Super slomo: High quality estimation of multiple intermediate frames for video interpolation," *2018 IEEE/CVF*

Conference on Computer Vision and Pattern Recognition, pp. 9000–9008, 2018.

[28] X. Xu, L. Siyao, W. Sun, Q. Yin, and M.-H. Yang, "Quadratic video interpolation," in *NeurIPS*, 2019.

[29] Z. Huang, T. Zhang, W. Heng, B. Shi, and S. Zhou, "Rife: Real-time intermediate flow estimation for video frame interpolation," *ArXiv*, vol. abs/2011.06294, 2020.

[30] S. Niklaus and F. Liu, "Softmax splatting for video frame interpolation," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5436–5445, 2020.

[31] J. ho Park, C. Lee, and C.-S. Kim, "Asymmetric bilateral motion estimation for video frame interpolation," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 14519–14528, 2021.

[32] S. Chen and M. Zwicker, "Improving the perceptual quality of 2d animation interpolation," *ArXiv*, vol. abs/2111.12792, 2021.

[33] M. C. Trinidad, R. Martin-Brualla, F. Kainz, and J. Kontkanen, "Multi-view image fusion," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4100–4109, 2019.

[34] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2414–2423, 2016.

[35] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," *ArXiv*, vol. abs/1804.07723, 2018.

[36] F. A. Reda, G. Liu, K. J. Shih, R. Kirby, J. Barker, D. Tarjan, A. Tao, and B. Catanzaro, "Sdc-net: Video prediction using spatially-displaced convolution," *ArXiv*, vol. abs/1811.00684, 2018.

[37] S.-H. Zhang, T. Chen, Y.-F. Zhang, S.-M. Hu, and R. R. Martin, "Vectorizing cartoon animations," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 4, pp. 618–629, 2009.

[38] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2015.

[39] F. Liu, C. Shen, G. Lin, and I. D. Reid, "Learning depth from single monocular images using deep convolutional neural fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, pp. 2024–2039, 2016.

[40] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," in *SSST@EMNLP*, 2014.

[41] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," *ArXiv*, vol. abs/2003.12039, 2020.

[42] D. Fourure, R. Emonet, É. Fromont, D. Muselet, A. Trémeau, and C. Wolf, "Residual conv-deconv grid network for semantic segmentation," *ArXiv*, vol. abs/1707.07958, 2017.

[43] A. Dosovitskiy and T. Brox, “Generating images with perceptual similarity metrics based on deep networks,” in *NIPS*, 2016.

[44] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, “Photo-realistic single image super-resolution using a generative adversarial network,” *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 105–114, 2017.

[45] M. S. M. Sajjadi, B. Schölkopf, and M. Hirsch, “Enhancenet: Single image super-resolution through automated texture synthesis,” *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 4501–4510, 2017.

[46] L. A. Gatys, A. S. Ecker, and M. Bethge, “A neural algorithm of artistic style,” *ArXiv*, vol. abs/1508.06576, 2015.

[47] Y. Li, N. Wang, J. Liu, and X. Hou, “Demystifying neural style transfer,” *ArXiv*, vol. abs/1701.01036, 2017.

[48] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Communications of the ACM*, vol. 60, pp. 84 – 90, 2012.

[49] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox, “Flownet: Learning optical flow with convolutional networks,” in *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[50] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, “A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation,” in *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. arXiv:1512.02134.

[51] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, “A naturalistic open source movie for optical flow evaluation,” in *European Conf. on Computer Vision (ECCV)* (A. Fitzgibbon et al. (Eds.), ed.), Part IV, LNCS 7577, pp. 611–625, Springer-Verlag, Oct. 2012.

[52] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *CoRR*, vol. abs/1412.6980, 2015.