Frame Interpolation in Animation Processing

Loc Tran
Department of Electrical and Computer Engineering
New York University Tandon School of Engineering
Brooklyn, United States
lat9357@nyu.edu

Runyu Wang
Department of Electrical and Computer Engineering
New York University Tandon School of Engineering
Brooklyn, United States
rw3299@nyu.edu

***Abstract*—This project aims to explore and develop motion interpolation techniques for animation processing. Motion interpolation is a technique used to create artificial frames between two existing frames to smooth out motion in a video. The project will focus on exploring and developing an effective and efficient algorithm that can accurately predict the motion between two frames and generate new in-between frames. We aim to accomplish a animation interpolation algorithm that can be used under various types of animation.**

***Index Terms*—Image Processing, Computer Vision**

I. Introduction

After major success worldwide of Studio Ghibli with many animation series or Marvel Studios with their superheroes franchises, the demands for animation movies is higher than ever. Production companies have to keep up with the demands with constant new releases all seasons. However, the animation production pipeline has some drawbacks. Most available animations on the market has the frame rate of 24 to 30 frames per second (fps), but not all of them are key frames. The studio first create 8 to 10 hand-drawn key frames. After that, they draw the intermediate frames between the key frames by in-betweening, a process in animation production. The manual work is labor intensive and costly hence many studios are interested in speed up and reduce the cost of the in-betweening process by using automatic frame interpolation. In this paper, we examine this problem by exploring the current approach by Dr. Siyao [1] in 2021, Dr. Ronneberger in 2015 [2] , Dr. Li in 2020 [3] and Kevin in 2020 [4].

II. Background

A. ATD-12K Dataset

ATD-12K Dataset is an animation triplet dataset created by Siyao and his team in 2021 [1]. The dataset consists of 25+ hours of animation scenes divided to 101 clips in two resolutions 1920×1080 and 1280×720. Some triplets from the data sets can be seen below (from both train and test sets).

Fig. 1. Triplet from Aladdin (Walt Disney Pictures 1992)

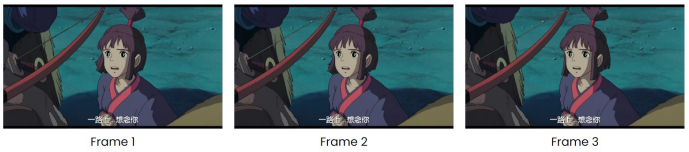

Fig. 2. Triplet from Princess Mononoke (Studio Ghibli 1997)

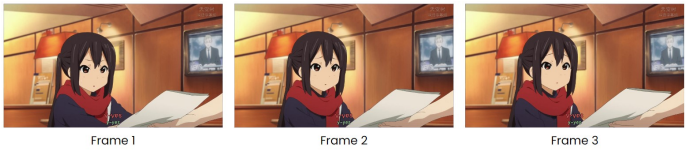

Fig. 3. Triplet from K-On! (Kyoto Animation 2009)

From figure 2 and 3, we can see that some of the triplets have watermark/subtitle or black border on them, which might be an issues that need to be addressed. However, the dataset still has a board range of animation styles: Disney 2-D animation (different from the 3-D like graphics of newer Disney movie like Toy Story (Pixar 1995), which might require more computing power to process), Japanese Anime (Including Studio Ghibli's detail rich hand-drawn style). This enables us to have a wide range of data to train and test our network architecture.

B. U-Net

The UNet architecture is a deep learning model specifically designed for image segmentation tasks. Figure 4 [2] is an example U-net architecture for 32x32 pixels in the lowest resolution.

The "U-Net" architecture consists of 2 parts:

1) Contracting Path (left side): a Convolutional Neural Network which scans the image, extract patterns from it, and combine them into features using a sequence of convolutional and max pooling layers.
2) Expansive Path (right side): The network upsamples the feature maps from the contracting path and combine them with the features from the input image to produce the final segmentation map. This is done using a sequence of convolutional and upsampling layers.

By having 2 parts with skip connection in between, the architecture enables the decoder to utilize all features from

the encoder. Thus, The U-Net doesn't need multiple runs to perform image segmentation and can learn with very few labeled images. Lighter architectures means we can run experiments, which require high processing power, on a more limited system like what we are currently dealing with.

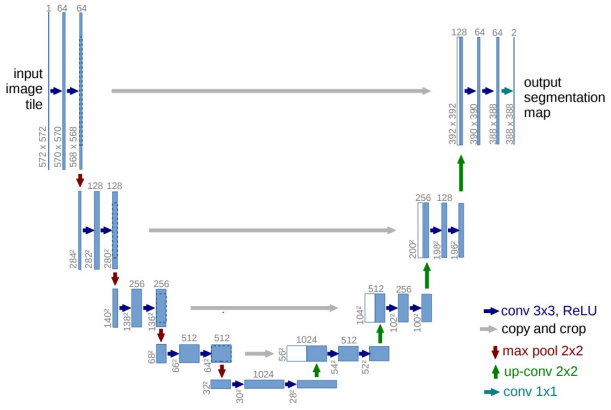

Fig. 4. U-Net Architecture [2]

C. RRIN

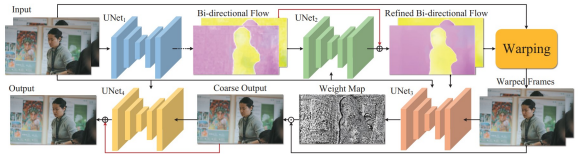

Fig. 5. Orignal RRIN Model [3]

Optical Flow is used to compute the motion of the pixels between two images sequence. It provides point to point pixel correspondence between that image, including direction and magnitude in a vector.

Introduced in 2020 [3], RRIN is a deep neural network that uses residue learning and adaptive weight map used for video frame interpolation. Residue refinement is used both on optical flow and features generation (wraped frames), resulting in improved visual quality. The method also requires less computing power to execute, due to U-Net implemented submodules with less depth as explained in the above section.

To interpolate an in-between frame in a video using RRIN, the steps are the following: [3]:

1) Calculate bi-directional optical flow between two consecutive provided frames.
2) Improve arbitrary-time optical flow using residue learning.
3) Create two approximations of the in-between frames by using the refined flow to warp the input frames in both directions
4) Combine the resulted frames linearly.
5) Use residue learning to refine the approximate, resulting in the final output.

III. Experiment

A. Revised Model

One limitation of RRIN is its dependency on computationally demanding of U-Nets to achieve residue refinements. Both UNet2 and UNet4 in RRIN consist of 5.47 million parameters each. Consequently, RRIN's parameter count reaches 19.19 million. This will effect both runtime and space consumption to the model.

As former work [5] concluded, it is possible to do modifications on networks to have better performance on both run time and model size on image processing field. From a similar interpolation work done by RRIN, it is noticeable that there is potential increase of efficiency by introducing generative adversarial network (GAN) [4]. To maintain high-quality outputs and decrease the runtime of RRIN, the approach by substituting the U-Nets for residue refinement in RRIN with a discriminator from Pix2Pix discriminator GAN is selected. Note that to align with the original RRIN model and accommodate the use of small batch sizes, batch normalization layers from the PatchGAN discriminator are removed, because it requires large batch size to generate accurate mini-batch statistics.

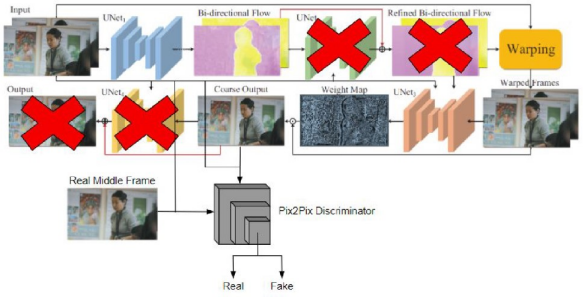

Fig. 6. Modified RRIN Model [4]

TABLE I
Number of Pramters and Runtime [6] [7] [3]

	RRIN	DAIN	MEMC-Net
Number of Parameters (in millions)	19.19	24.02	70.31
Runtime (in seconds)	0.08	0.13	0.12

B. Dataset

As introduced in background section, ATD-12K is applied in our experiment. Since the dataset is originally developed for interpolation, no additional processing is required as the video clips were already prepared as 3-frame sequences. However, as the computing power of the author's device is relatively limited, only 1200 triplets of the whole dataset are selected for the experiment, with a 80:10:10 train:val:test split. To decrease the effect of the narrow scope of dataset, a series of data augmentation techniques are applied to enhance the training capacity of train set. In the experiment, mirror flipping, cropping and random rotating are applied first, then each image will be resized to 448x256 for model input.

C. Parameters

In the PatchGAN discriminator, the Adam optimizer is deployed, and beta values from the Pix2Pix model's suggestions are used, which are $\beta_1 = 0.5$ and $\beta_2 = 0.999$. The weight between the custom RRIN's mean squared error (MSE) loss and the discriminator's adversarial loss is a hyperparameter that needs to be determined. While the original Pix2Pix model utilizes mean absolute error (MAE) loss instead of MSE loss which is practically used in the project, The ratio between the MSE loss and the adversarial loss from the original work, which are 0.99 and 0.01 respectively, are kept unchanged. As for training hyperparameters, since the memory capacity is constrained and original RRIN model prefers a small batch size as mentioned in background section, and only batch size 4 is tested with 20 epochs.

D. Evaluation Metrics

1) MSE: It is a widely used metric for measuring image quality. It calculates the average of the squared errors, which are the pixel-wise differences between the original and generated images.

$$MSE(y_i, g_i) = \sum_{i=1}^{D} (y_i - g_i)^2$$

In the formula, y_i represent ground truth images, while g_i are generated instances.

2) PSNR: It is a metric that measures the pixel-wise comparison between the original and generated images, similar to MSE. A higher PSNR value signifies better quality in the reconstructed image, and it is a valuable tool for approximating human perception of reconstruction quality. Hence, PSNR plays a crucial role in evaluating interpolated frames.

$$PSNR = 10 \log 10 \frac{R^2}{MSE}$$

3) SSIM: Although metrics like MSE and PSNR are commonly used for assessing image quality due to their simplicity and clear interpretation, they may not excel in discerning visual quality and structural content in images. Therefore, we additionally employed SSIM (Structural Similarity Index), a metric that considers the human visual system's characteristics. SSIM models the loss of correlation, luminance distortion, and contrast distortion, enabling it to estimate the perceived quality of images more effectively [8].

$$SSIM(y_i, g_i) = \frac{(2\mu_{y_i}\mu_{g_i} + C_1) + (2\sigma_{y_i g_i} + C_2)}{(\mu_{y_i}^2 + \mu_{g_i}^2 + C_1)(\sigma_{y_i}^2 + \sigma_{g_i}^2 + C_2)}$$

IV. Results

A. Train Result

The figure presents the performance results, where the blue lines labeled as "Custom RRIN" correspond to our custom RRIN's loss. The orange lines labeled as "D - real" represent the discriminator's loss on real images, while the green lines labeled as "D - fake" indicate the discriminator's loss on images generated by the custom RRIN. To facilitate visual comparison, the MSE loss for the custom RRIN and BCE loss for the PatchGAN discriminator are displayed on the graphs. For better visualization and scaling purposes, the MSE loss is multiplied by 100 in the graphs.

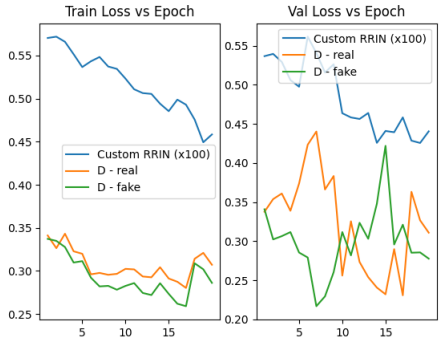

Fig. 7. Train Result

B. Evaluation Result

Compared to result of very mature model on larger and more variant datasets, the model obtains a satisfying result in PSNR and SSIM values.

TABLE II
PSNR AND SSIM METRICS [6] [7] [3]

	Our Model	RRIN	DAIN	MEMC-Net
PSNR	30.37	35.22	34.32	34.4
SSIM	0.962	0.964	0.941	0.962

C. Generated Samples

Aside from numerical results, a variety of interpolations are done on both images from or outside the dataset to better illustrate and visualize the result. The samples have overall great performance although there are occasionally blurs and distortions.

Fig. 8. Generated Interpolation for Aladdin

V. Conclusion

The project successfully applied a modified RRIN model on the task of animation interpolation and achieved outcomes that fitted the author's expectation on both numeric results and real interpolation generation. As satisfying results have been achieved as shown in former sections, it is certain that there are plenty of potential improvements to the project.

Fig. 9. Generated Interpolation for Sword Art Online

Fig. 10. Generated Interpolation for K-ON!

Fig. 11. Interpolation for Honda Cub (Outside of ATD-12K)

Fig. 12. Interpolation for Honda Cub (Outside of ATD-12K)

Since the project is executed on a platform with very limited resources like GPU power and memory so a small batch size for the ease of graphic memory and limited epoch numbers for reasonable training time are applied in the project though they may affect overall performance. Also, we have a relatively small dataset while GAN typically needs a large dataset to fully function, which is one of the reasons for dropping normalization layers from the PatchGAN discriminator. There are points of improvement which can be refined in further work.

REFERENCES

[1] L. Siyao, S. Zhao, W. Yu, W. Sun, D. N. Metaxas, C. C. Loy, and Z. Liu, "Deep animation video interpolation in the wild," 2021.

[2] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *CoRR*, vol. abs/1505.04597, 2015. [Online]. Available: http://arxiv.org/abs/1505.04597

[3] H. Li, Y. Yuan, and Q. Wang, "Video frame interpolation via residue refinement," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 2613–2617.

[4] Y. Kevin, C. Nikos, Y. Samson, and H. Tan, "Artificial intelligence project report," 2020.

[5] A. Sevastopolsky, "Optic disc and cup segmentation methods for glaucoma detection with modification of u-net convolutional neural network," *Pattern Recognition and Image Analysis*, vol. 27, pp. 618–624, 2017.

[6] W. Bao, W.-S. Lai, C. Ma, X. Zhang, Z. Gao, and M.-H. Yang, "Depth-aware video frame interpolation," 2019.

[7] W. Bao, W.-S. Lai, X. Zhang, Z. Gao, and M.-H. Yang, "Memc-net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 3, pp. 933–948, 2021.

[8] U. Sara, M. Akter, and M. S. Uddin, "Image quality assessment through fsim, ssim, mse and psnr—a comparative study," *Journal of Computer and Communications*, vol. 07, pp. 8–18, 01 2019.

APPENDIX

A. *Repository Link*

https://github.com/Ryw-cloud/IVP_project/tree/main