NeCo: Neural Coordinate-based Internal Learning for GIF Interframe Synthesis

Di Huang(502308), Zihao Zou(502540), Di Huang(488194), Boxiang Hou(490049), Zhengji Wu(500379)

Abstract

Computer version and graphics has always been one of the more popular research directions in the field of computing. But the traditional MLP model has a fatal drawback, which it will have a poor performance on the high frequency content present in image when the input is low dimension. However, depending on a new feature expansion technique-Fourier Feature Mapping (FFM), it will be feasible for MLP to learn high frequency functions in low dimensional problem domains. Thus, we decide to conduct this research by adopting FFM on MLP in order to let our GIF's motion more smoothness. For the result, under different range of frequencies (from 5 to 20), we all achieve a very low PSNR (about 30) after enough epoch, which indicates FFM truly help to overcome the drawback of the standard MLP model. We propose a novel deep-learning-based approach to synthesize intermediate frames between two adjacent frames in a GIF. We call it Neural Coordinate-based Internal Learning for GIF Interframe Synthesis(NeCo). The mainly advance of our algorithm is that we can generate arbitrary interframes with low cost after training, and the result can still retain the spatial-temporal information if the input images do not vary too much.Experimental results show that we approach can achieve fairly good performance on some test images.

1 Introduction

Video frame interpolation, as shown in 1, is a technique to smooth the motion in a sequence of frames by synthesizing new frames between two adjacent frames. This technique has been developed for several decades. It was first implemented with analytic methods, and with the recent explosion in computational ability, it quickly improved with the aid of a deep neural network. Our goal in this paper aims to solve a problem that people frequently meet in real life–low-quality GIFs.

GIF, which is Graphics Interchange Format, is a bitmap image format. Like the video, it can be referred to as a loop of the animated image set. Further, GIFs can be decomposed into a sequence of images in discrete time. Therefore, we define our goal as predicting the intermediate frames of the GIFs, which has the effect of smoothing the GIFs.

There is considerable interest in deep learning based solutions to intermediate frame synthesis. We begin by reviewing existing video interpolation approaches to demonstrate the significance of the problem. We then adopt an existing approach in the image reconstruction domain and translate its concept into the video interpolation scope. We are dedicated to contributing a new force to the computer vision community.

We adopt an image inversion approach-Coordinate-based Internal Learning (CoIL). Sun et al. (2021)proposed CoIL as a deep learning model for the continuous representation of measurements. It utilizes a multilayer perceptron (MLP) that maps the input coordinate to the corresponding sensor response. Traditional deep learning models such as Convolution Neural Network (CNN) and Res-Net utilize convolution kernels to encode the low-dimensional image input to higher dimensional content. CoIL alternatively utilizes a FFM transformation prior to training in MLP. The FFM methods are both introduced by Tancik et al. (2020) and Mildenhall et al. (2020). FFM is a feature expansion method

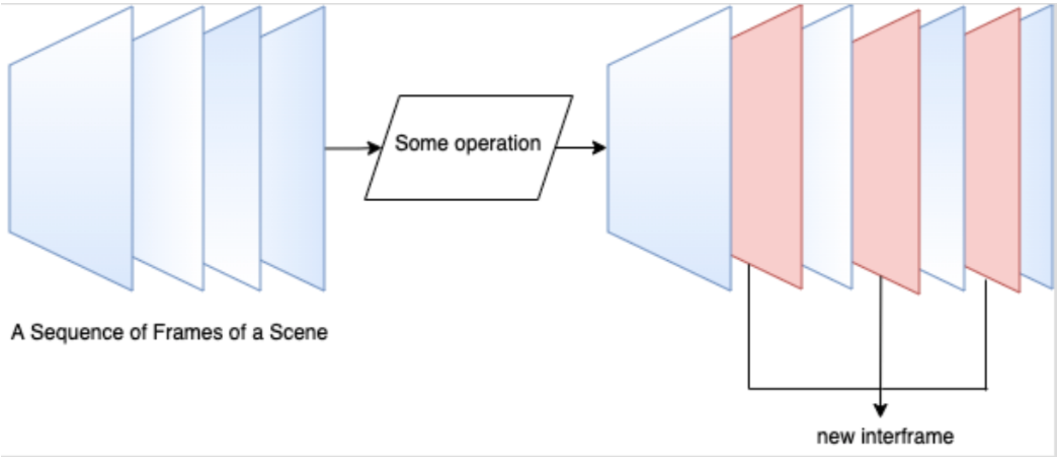

Figure 1: Video Frame Interpolation

that maps the low-frequency image coordinate into the high-frequency domain. Our work defines the input as 3-dimensional, and we utilize FFM to transform it into a higher frequency content. We implement a Residual MLP (ResMLP) to overcome the problem of gradient vanishing and expansion. We conduct several experiments to determine the suitable frequency range. We obtain the pixel value as the output. Our model NeCo can achieve 30.92 PSNR for GIFs after 100 training epochs.

2 Related Works

In this section, we discuss the recent advance in intern learning and video frames interpolation. Moreover, we discuss CoIL, the reseach that we are inspired by.

2.1 Deep Internal Learning

Recent advance in computer vision has shown the feasibility of encoding an object or scene information into a deep neural network implicitly. Curless and Levoy (1996) introduced a method to reconstruct a surface with a sequence of aligned range images with neural network. CPPNsStanley (2007) is a neural network that encodes the DNA information into the weights. NeRF is a recent advance novel view synthesis. It employs a MLP network to represents a complex scene. The network can output corresponding view's color information based on different spatial coordinate input.

2.2 Video Frames Interpolation

Many approaches has been proposed in the area of video interpolation. These approaches can be categorize into two different methods: analytic methods and model-based methods. We briefly review these methods and discuss their impacts.

2.2.1 Analytic Methods

The classic approachHerbst et al. (2009) utilize the optical flow, and analyze the occlusion of the objects in the scene. This approach can generate arbitrary interframes between two adjacent frames. One thing challenges this method is that the object's boundary is still hard to intepolate since it is highly dynamic.

Mahajan et al. (2009) proposed an approach that traces an pixel movement along its path, and generate arbitrary intermediate frames. It utilizes a poisson equation to handles the occlusion of a pixel.

2.2.2 Model-Based Methods

Recent progress in deep learning enables the algorithm to analyze the statistic information of the scenes without explicitly expressing it. Several approaches are based on neural network. Jiang et al. (2018) adopted convolution neural network to generate the intermediate frames. They resolved the boundary problem by feeding the analytic result to a convolution neural network.

2.2.3 CoIL Methods

CoIL is a coordinate-based internal learning method. As the internal learning suggested, CoIL does not require separate training examples for training the model, which leverages the redundancy within the measurements of a single unknown image. CoIL consists of two core parts:

1. Fourier Feature Mapping (FFM), FFM is a feature expansion method that encodes the low-dimensional image coordinate into the high-frequency sine and cosine frequency domain.
2. Residual MLP. (ResMLP) CoIL utilizes the Residual MLP as the backbone for the training task. The Residual MLP in CoIL extends the standard MLP with additional concatenate layers, sometimes called the skip layers He et al. (2016). ResMLP has the benefit of solving the vanishing gradient problem. CoIL shows its outstanding performance in image reconstruction problems.

3 Implementation Details

Our approach defines a model consisting of two parts: 1) Fourier Feature Mapping (FFM), 2) Residual Mutlilayer Perceptron (ResMLP). The former one is defined before training, and the later one needs to be optimized. We implement the Fourier-Feature Mapping (FFM) that augments the low-frequency feature in a high-frequency domain. FFM is a feature augmentation strategy that aims to solve the natural difficulties underlying the intermediate frame generation. We then implement a Residual Mutli Layer Perceptron (ResMLP) that inputs the high-frequency features generated by FFM to generate the intermediate frames. We discuss the technical detail in the following sub-section.

3.1 Fourier Feature Mappings

The difficulty in generating arbitrary intermediate frames is generating high-resolution intermediate frames. Traditionally, model-based approaches can merely generate the frames but lose significant amount of image quality due to the low-frequency input. We consider a novel approach by encoding the low-frequency frames into a high-frequency domain into a MLP neural network. We then use the standard image quality metric-Peak-Signal-to-Noise (PSNR) to evaluate our proposed approach and demonstrate its superiority.

Consider the input feature:

$$(x, y, t) \tag{1}$$

where x, y represent the coordinate in two-dimensional space, and t represents the time-stamp. The three-dimensional input is considered to be a low-frequnecy input. We then implement FFM to augment the low-frequency input. In the article CoIL: Coordinate-based Internal Learning for Imaging Inverse Problems, Sun et al. (2021) has proposed that the internal learning could be boosted by three to four dB with the aid of FFM.

3.1.1 Basic FFM

as shown in Figure2, where the constant L defines the range of the frequency, and k is a step function, this method linearly map the original triples into frequency, which the range of frequency is controlled by the L parameters.

3.1.2 Gaussian FFM

Gaussian FFM is a variant of basic FFM. Every frequency in the basic FFM is added a factor B in Gaussian FFM, where B is sampled from $N(0, \sigma^2)$, and σ is user-defined. This method employs a random variable to generate a more dynamic frequency.

$$\begin{pmatrix} sin(k_1\pi x), & cos(k_1\pi x), & sin(k_1\pi y), cos(k_1\pi y), & sin(k_1\pi t), cos(k_1\pi t) \\ sin(k_L\pi x), & cos(k_L\pi x), & sin(k_L\pi y), cos(k_L\pi y), & sin(k_L\pi t), cos(k_L\pi t) \end{pmatrix}$$

Figure 2: Fouier Feature Mappings

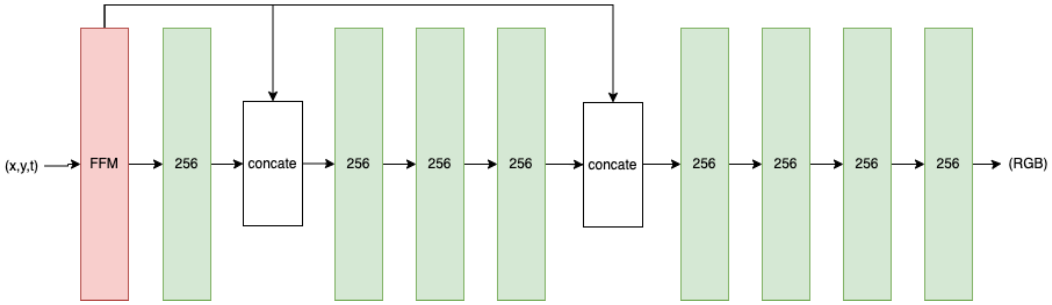

Figure 3: MLP architecture

After comparing results from both methods, we empirically found that the basic FFM can better fit our job. We encode the input to a higher frequency domain controlled by L. We conclude that in brief, ResMLP learning from FFM encoding generates higher PSNR intermediate frames than stand-alone ResMLP. We perform multiply testing on several GIFs and recommend to the choose L = 10, which gives the best PSNR results. We will discuss the experimental detail about choosing proper L, basic FFM vs. Gaussian FFM, and MLP with FFM and without FFM in the discussion section.

3.2 Residual Mutlilayer Perceptron

We implement an 8 layers ResMLP in Neco, containing 3 concatenation layers. All layers are fully connected and have 256 hidden neurons, and have the LeakyReLU activation except the last layer. We implement 2 concatenation layers before the second and fifth hidden layers. The concatenation layer is also fully connected but with increased input size. The input size was enlarge by FFM to (#input*2*L). We optimize the model by Adam and use weight decay to optimize the learning rate. Neco trains a separate MLP to generate the intermediate frames of a given GIF. Therefore, there is no testing set and no model's generalization ability.

3.3 Training Workflow

Given a GIF, we first extracted the GIF into a sequence of images. Based on the pixel's value, its position and the order in the sequence, we generate a sequence of sex-element tuples:

$$(x, y, t) \rightarrow (R, G, B) \tag{2}$$

all values are normalized to between 0 and 1 for stability. We then feed (x, y, t) to the network. the corresponding outputs are compared to (R, G, B) with L2 loss. Then we should the loss to optimize our network. The graph4 illustrate this procedure.

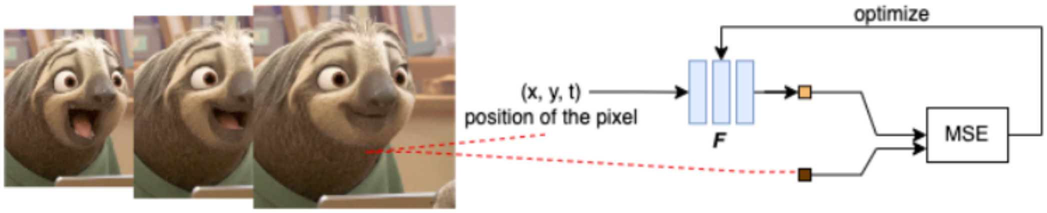

Figure 4: Training workflow

4 Experiments

We conduct experiments to demonstrate the superiority of Neco. First, we test on the effect of L in FFM to learn the proper frequency range. Second, we compare the Neco with stand-alone MLP to testify the effectiveness of FFM layers. Third, we compare the Neco with basic FFM layers and Gaussian FFM layers to examine the variation of FFM. We conduct the experiments by downsampling the GIF and training models to generate intermediate frames. In addition to generating existing frames from, we perform the slow motion experiments on the GIFs, which predicts the frame that does not exist in the original frame. We run the experiments under the same configuration. We train the model using Nivida RTX 2080. We choose L = 10 in the FFM layers and train each model for 100 epochs with learning learning rate $10^{-}3$ and weight decay $10^{-}8$. We use Sloth GIF as our first and main experiment object. The reason why we use Sloth is that its motion is slow and continuous, which makes it an ideal object to visualize the quality of reconstructed GIF. There are 95 frames in the ground truth Sloth, and we keep only 19 frames in our training data. Hence, our model will predict the missing 76 frames.

4.1 Number of Epochs

At the beginning, we train the model for 300 epochs under our chosen configuration. After analyzing the training log, we find out that the validation PSNR remains around 30 and there is few improvement after 100 epochs. Since it takes 6 hours to train our model for 300 epochs in one experiment, which is twice the time of training for 100 epochs, we decide to set the number of epochs = 100 in the following experiment.

5 Discussion

5.1 Different Ranges of Frequency

L (different ranges of frequency) is a critical hyperparameter in FFM since it decides how big our embedding space is and so the input size of MLP module. To find out the effects of parameter L on model performance, we train the model with our chosen configuration and collect the validation PSNR when L equals 5, 10, 15, 20 respectively. The result is in Fig 6. As shown in 6, the model performs best when $L = 10$. There is not too much difference among other three L settings. Specifically, only using $L = 5$ reduces the performance and increasing L from 10 to 20 doesn't improve performance. So we set $L = 10$ in the following experiments.

5.2 Input Mapping

As we have mentioned, input mapping strategies can significantly boost the model perfomance. So in this experiment, to validate the benefits of using FFM for our model, we train the model with basic FFM, without FFM, and with Gaussian FFM under our chosen configuration. The result is shown in Fig 5, in which from left to right is the ground truth, without FFM, with basic FFM, and with Gaussian FFM. The validation PSNR for the reconstructed three is 28.35dB, 30.4dB, 30.91dB respectively1. As shown in fig 5, the quality of the reconstructed Sloth without FFM is the worst. We cannot clearly see the details of Sloth such as its fur between its two eyes (marked by the red

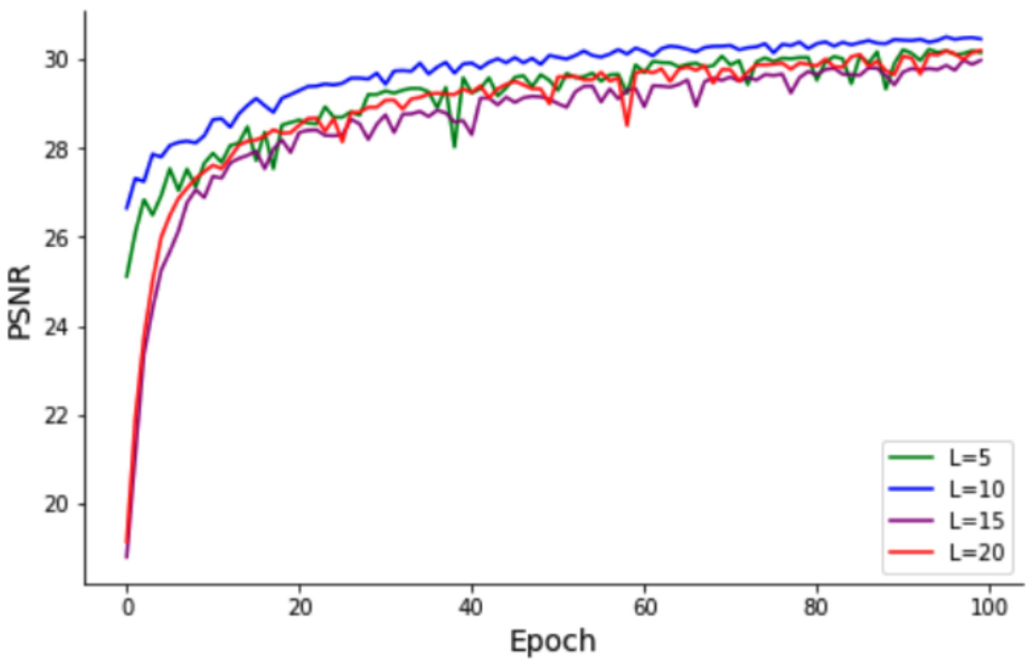

Figure 6: PSNR comparison under different L value

rectangle). Comparing the reconstructed Sloth with basic FFM and the one with Gaussian FFM, the latter is slightly better.

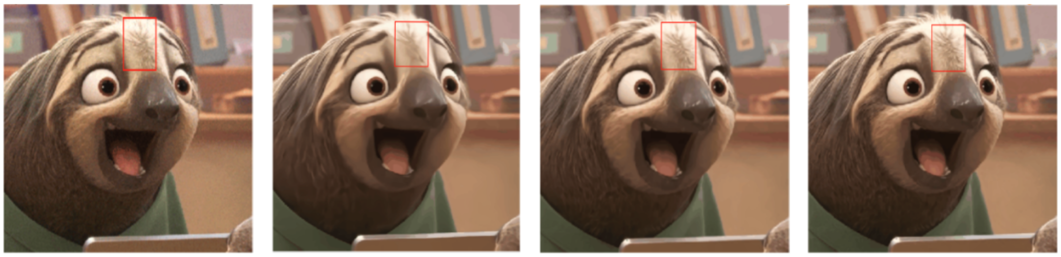

Figure 5: Sloth result

	MLP	MLP+Basic FFM	MLP+Gaussian FFM
PSNR(dB)	28.35	30.4	30.91

Table 1: PSNR comparison of the Sloth.gif

5.3 Inferring Inexistent Frames

To exploit our model's capability of inferring frames that don't exist, we don't reduce the number of frames in the ground truth. Hence, in this experiment we are inserting frames and GIFs with slower motion than the ground truth are created. We pick three GIFs, which are Rocket 7, Spongebob Squarepants 8, and Girl 9. There aren't lots of colors in Rocket and only its hands is waving and its body is moving upward and downward. Compared with Rocket, Spongebob Squarepants and Gril are more colorful and their movements are much more complicalted.

Like what we do in the experiment on Sloth, we train our model for the three GIFs respectively with 100 epochs. The GIFs constructed by our model are 10, 11, 12.

After inserting frames, only the quality of the reconstructed Rocket is acceptable. We can clearly see that the waving speed of its hand is slowed. For the reconstructed Spongebob Squarepants, although the shaking speed of its body is slowed down, the whole GIF is blurred. Both arms disappear and the colors in-between motions becomes black. For the reconstructed Gril, our model only extends each motion in the training frame a little bit longer without really capturing the moving trend.

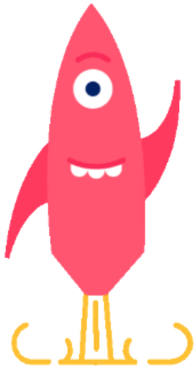

Figure 7: Rocket

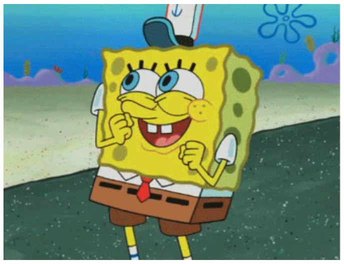

Figure 8: Spongebob squarepants

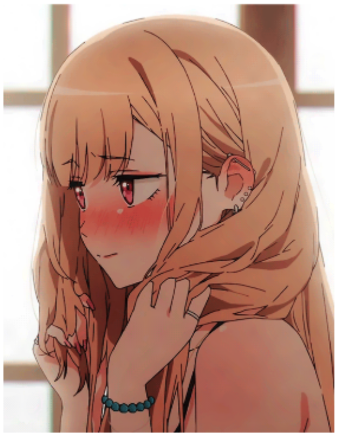

Figure 9: Girl

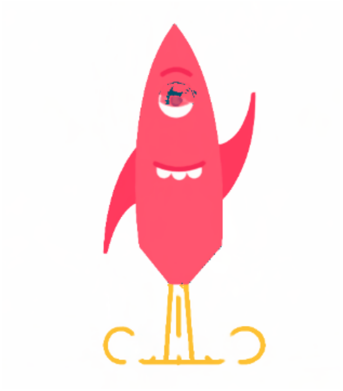

Figure 10: Rocket reconstruction

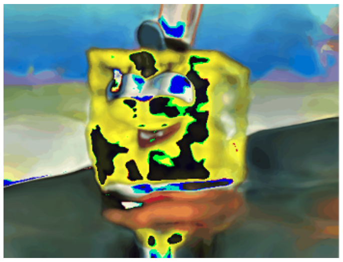

Figure 11: Spongebob squarepants reconstruction

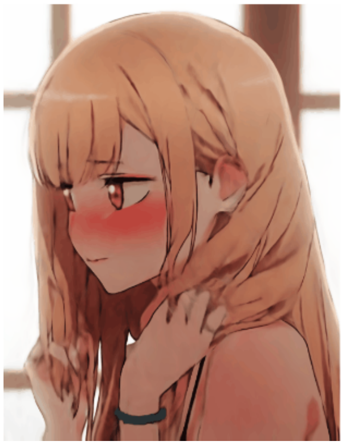

Figure 12: Girl reconstruction

6 Future Works

We conduct experiments on 4 GIFs-sloth, rocket, Spongebob, and anime girl. We observe some downsides of our proposed approach. First, the time complexity of NeCo is relatively high, although it requires no separate training examples. Our training platform is RTX 2080 with AMD 5800X. We spend nearly 3 hours training 100 epochs for each GIF. The expected training time can be long because we adopt MLP as the network framework with 256 hidden neurons. In addition, we implement the model in Python, which is a script language. The compile-time is relatively long. Therefore, we may need to consider implementing the computational model in a light-speed computer language. Second, the overall sloth's GIF performance is decent. We can achieve 30 PSNR as a result. However, the state of art method can generate PSNR over 35. As a comparison, we observe that the detailed fur synthesis in similar areas is relatively insufficient, such as the fur below the mouth. Therefore, we may consider adopting advanced deep learning models such as CNN as the backbone framework and preserve FFM as the feature expansion technique. This integrated method sheds light on an outstanding performance. We observed unsatisfactory performance on the slow-motion synthesis of rocket GIF, Spongebob GIF, and anime Girl GIF. For these three experiments, the synthesized interframes are blurry and even distorted. There is a common characteristic of these three GIFs: the character's movement speed is fast. Considering these factors, our model may not perform robustly in the GIFs with high movement speed. There are two potential improvements:

1. We can further adjust the input content. Besides FFM, we may consider extracting more information from the original image input and overcoming the high-speed motion problem in GIF synthesis.
2. We may need to conduct experiments on several deep learning models such as CNN and compare them with the proposed MLP model.

References

Sun, Y.; Liu, J.; Xie, M.; Wohlberg, B.; Kamilov, U. S. *arXiv preprint arXiv:2102.05181* **2021**,

Tancik, M.; Srinivasan, P.; Mildenhall, B.; Fridovich-Keil, S.; Raghavan, N.; Singhal, U.; Ramamoorthi, R.; Barron, J.; Ng, R. *Advances in Neural Information Processing Systems* **2020**, *33*, 7537–7547.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; Ng, R. Nerf: Representing scenes as neural radiance fields for view synthesis. European conference on computer vision. 2020; pp 405–421.

Curless, B.; Levoy, M. A volumetric method for building complex models from range images. Proceedings of the 23rd annual conference on Computer graphics and interactive techniques. 1996; pp 303–312.

Stanley, K. O. *Genetic programming and evolvable machines* **2007**, *8*, 131–162.

Herbst, E. V.; Seitz, S. M.; Baker, S. Occlusion Reasoning for Temporal Interpolation using Optical Flow. 2009.

Mahajan, D.; Huang, F.-C.; Matusik, W.; Ramamoorthi, R.; Belhumeur, P. *ACM Transactions on Graphics (TOG)* **2009**, *28*, 1–11.

Jiang, H.; Sun, D.; Jampani, V.; Yang, M.-H.; Learned-Miller, E.; Kautz, J. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. Proceedings of the IEEE conference on computer vision and pattern recognition. 2018; pp 9000–9008.

He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016; pp 770–778.