國立臺灣大學電機資訊學院資訊工程學系
碩士論文
Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Taiwan University
Master Thesis

基於循環畫面生成之視訊畫面內插
Deep Video Frame Interpolation using Cyclic Frame Generation

廖苡彤
Yi-Tung Liao

指導教授：莊永裕博士, 林彥宇博士
Advisor: Yung-Yu Chuang, Ph.D.
Yen-Yu Lin, Ph.D.

中華民國 107 年 7 月
July, 2018

doi:10.6342/NTU201802624

國立臺灣大學電機資訊學院資訊工程學系
碩士論文
Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Taiwan University
Master Thesis

基於循環畫面生成之視訊畫面內插
Deep Video Frame Interpolation using Cyclic Frame Generation

廖苡彤
Yi-Tung Liao

指導教授：莊永裕博士, 林彥宇博士
Advisor: Yung-Yu Chuang, Ph.D.
Yen-Yu Lin, Ph.D.

中華民國 107 年 7 月
July, 2018

doi:10.6342/NTU201802624

國立臺灣大學碩士學位論文
口試委員會審定書

基於循環畫面生成之視訊畫面內插

Deep Video Frame Interpolation using Cyclic Frame Generation

本論文係廖苡彤君（學號 R05922033）在國立臺灣大學資訊工程學系完成之碩士學位論文，於民國 107 年 7 月 30 日承下列考試委員審查通過及口試及格，特此證明

口試委員：莊永裕　林彥宇

（指導教授）

陳煥宗

系主任 莊永裕

誌謝

首先，我想感謝在最後的這半年能與劉育綸學長合作，因為合作而開始的新方向，讓我突破了一年多來研究的瓶頸；而過程中的種種挫折或原地踏步，也真的很幸運能跟你一起討論一起克服。

謝謝歪歪一開始就算不知道我是誰，還是願意收我當研究生，謝謝你這兩年來的支持鼓勵與嗆我愛吃東西；謝謝一開始舉辦的 Deep learning 讀書會，讓初初踏入這個領域的我獲益良多。謝謝林彥宇老師，帶著我學會在研究這條路上該怎麼走：碩一時的每週論文報告，讓我學會了快速抓住每篇論文的重點，逐漸引導出我的研究方向；每週的咪挺也讓我逐漸找到做研究的竅門，進而發展出這篇論文。

謝謝滿滿瘀青學學克萊兒曾瑀呈呈還有胖達萌萌百變怪奇奇漸層，這段日子一來一起努力一起咖啡廳唸書寫扣真的很開心，能有這樣彼此督促的日子真的很好。謝謝醜男。謝謝 CVLAB 的大家，特別是洸瑞學長、存毅學長和陽明學長，每次向你們討教的時候都有醍醐灌頂的感覺。謝謝工作站的 GPU，還有讓我得以搶 GPU 的這台 MAC。

謝謝最愛的潘，即使隔了一個太平洋也會聽我分享研究點滴，適時給我各種意見，真的有你真好。謝謝象象與若予，謝謝你們不時的約我出門，讓我在充電之後繼續充滿力量地往前。謝謝瑜珈，讓我雖然總是對著電腦，晚睡晚起，還是擁有一個舒展身心的上午。

謝謝我愛的家人，包容我在研究低谷時的碎碎念與壞脾氣。謝謝媽媽安心的擁抱、弟弟偶爾的幫我整理房間、爸爸拉我出去散步買甜點、阿嬤每次溫暖的飯菜。謝謝兩年前的我選擇踏入這個領域，及這兩年來的不斷努力與堅持，未來的我也要秉持這樣認真的心繼續努力！

doi:10.6342/NTU201802624

摘要

這篇論文針對三項在卷積類神經網路應用於視訊畫面內插的觀察，提出對應的解決辦法。首先，若能利用生成的畫面再進行影像內差並得到高品質的結果，此生成畫面便是更可靠的。第二，在短時間內的光流 (optical flow) 變化為線性的。第三，根據對過去研究成果的分析，影像畫面內插在梯度值愈大的部分，內插的效果愈不好。

我們提出了一個主要的損失函數 (loss term) – 循環一致性損失函數 (Cycle Consistency Loss)，與兩種應延伸應用。利用循環一致性損失函數，我們便能比過去的研究更有效地利用訓練資料集，使得生成的畫面品質更好，也能在訓練資料量下降時維持模型的品質。兩種延伸應用為運動線性特徵損失函數 (Motion Linearity Loss) 與邊緣引導訓練 (Edge-guided Training)，分別針對影片中光流 (optical flow) 的劇烈變化與大梯度值提出改進方法。這兩種損失函數與訓練方式可以被應用在任何可以由頭至尾訓練的視訊畫面內插方法中。在這篇論文裡，我們主要是應用在 Deep Voxel Flow [15] 這篇研究所提出的模型上，用來展示我們每個部件的效果。我們在三個資料集上驗證我們提出的方法：其中，在 UCF101 資料集以及高畫質影片 <See You Again>，我們可以得到最高的峰值信號雜訊比 (PSNR); 而在 Middlebury 基準上，我們在真實場景的影片中表現最好，對於普遍化此方法在視訊畫面內插上也是最有利的。

關鍵詞： 視訊畫面內差 、 循環一致性 、 邊緣引導訓練

doi:10.6342/NTU201802624

Abstract

This thesis addresses three observations in video frame interpolation using convolutional neural networks (CNNs). First, interpolated frames are more reliable if we could further utilize them to interpolate high-quality output frames. Second, the optical flow should be linear within short time interval. Third, according to the analysis on previous works, there would be lower performances in areas with larger gradients. We tackle the three issues by introducing the new loss term, the *cycle consistency loss*, accompanied with two extensions, the *motion linearity loss* and the *edge-guided training*. The *cycle consistency loss* could fully utilize the training data, not only to enhance the results of interpolation, but also to maintain the performance with less training data than previous works. The *motion linearity loss* and *edge-guided training* are mainly designed for dealing with the areas with large motion and great gradient, respectively. These loss terms and training method could be applied on any end-to-end trainable network for video frame interpolation, and we adopt the Deep Voxel Flow [15] model in the thesis to demonstrate the effects of them. We have conducted the experiments on three datasets, achieving the state-of-the-art in two of them, the UCF101 benchmark and the high-quality video: "See You Again". For Middlebury benchmark, we have the best performance in the real scene videos, which is more important for the generalization of the frame interpolation method.

Keywords: Video frame interpolation, cycle consistency, edge-guided training.

doi:10.6342/NTU201802624

Contents

論文口試委員審定書 **ii**

誌謝 **iii**

摘要 **iv**

Abstract **v**

1 Introduction **1**

2 Related work **4**

2.1 Video Frame Interpolation 4

2.1.1 Traditional methods 4

2.1.2 Methods based on convolutional neural networks 5

2.2 Cycle Constraint 6

3 The proposed approach **7**

3.1 Full Model 7

3.2 Cycle Consistency Loss L_c 8

3.3 Motion Linearity Loss L_m 10

3.4 Edge-guided Training 11

4 Implementation details **13**

4.1 Implementation of the base model 13

4.2 Training details 14

doi:10.6342/NTU201802624

5 Experimental results **16**

5.1 Dataset 16

5.1.1 UCF101 16

5.1.2 Middlebury 17

5.1.3 High-quality Video: “See You Again” from Wiz Khalifa 18

5.2 Experimental setup 18

5.3 Ablation Study 19

5.4 Comparison to state-of-the-art methods 23

5.5 Visual comparison 25

6 Conclusions **27**

Bibliography **28**

doi:10.6342/NTU201802624

List of Figures

1.1 The training procedure for the proposed cycle consistency loss. First(1st) Stage: train a single interpolation network. Second(2nd) Stage: do the training to ensure the consistency between the output frame of the cycle model and the original input frame. 3

3.1 Full model architecure. 7

3.2 Details in the right image preserve more, with lower PSNR. (Left: ground truth; middle: PSNR=23.38 db; right: PSNR=21.48 db) 8

3.3 Simple model (The modified model from Deep Voxel Flow[15]), where I_0 and I_2 refer to the first frame and the third frame in the triplets, $F_{0\to 2}$ refers to the voxel flow between the two input frames, I_1 refers to the ground truth, and $\widetilde{I}_1$ refers to the interpolated frame. 9

3.4 Our Second-stage model, including the "simple model" and the "cyclic model", to apply the Cycle Consistency Loss. 10

3.5 Motion Linearity Loss. The blue blocks represent estimated flow maps. . 11

3.6 Edge-guided training with HED [34]. 12

3.7 Analytical results on the UCF101 testing set. (Up: The number of pixels in each gradient bin; Lower right: The statistical result of DVF; Lower right: The statistical result of SepConv.) 12

4.1 Base model architecture: U-Net. 14

5.1 Total 13320 videos in 101 action categories of the UCF101 dataset. . . . 17

5.2 One frame from each of the 12 categories in the Middlebury dataset. . . . 18

doi:10.6342/NTU201802624

5.3 The performances of our approach in PSNR with different vales of parameter (a) λ_c and (b) λ_m. 19
5.4 Evaluation of the performance for models with different training data size. 21
5.5 Visual results for model w/o and w/ motion linearity loss. 21
5.6 Visual results for model w/o and w/ edge-guided training. 22
5.7 *Left*: The comparison of performance for baseline and edge-guided training. The statistics of mean square error (the lower the better) between interpolated frames and ground truth in areas with different gradient magnitude, with the greater value to the right. *Right*: The improvement with the edge-guided training. (The decreasing magnitude in MSE.) 22
5.8 Qualitative evaluation on UCF101. 23
5.9 Quantitative evaluation on Middlebury dataset. 24
5.10 Qualitative evaluation on UCF101. 25
5.11 Qualitative evaluation on Middlebury. 26

doi:10.6342/NTU201802624

List of Tables

5.1 Evaluation of the performance for models with different training data size. 20

5.2 Evaluation of the effect of each component in our model. 23

5.3 Evaluation of the effect of the effect of Cycle Consistency Loss 24

5.4 Evaluation of the effect of Cycle Consistency Loss 24

doi:10.6342/NTU201802624

Chapter 1

Introduction

Speaking to videos, people prefer high frame rate videos for the visual enjoyment. Although it is possible to use cell phones to take 240-fps (frame-per-second) videos, professional cameras, which are hard to get for ordinary people, might be required for higher frame rate. On the other hand, recording all the video at high frame rate is quite impractical, since it needs large memories and is power-intensive for mobile devices. Video frame interpolation thus become a quite important topic. It aims to upscale the frame rate of videos by generating middle frames from lower frame rate videos. With this technique, we could produce smoother view transitions with the same memory used as the low frame rate videos. It also inspires new applications in self-supervised learning, serving as a supervisory signal to learn optical flow from unlabeled videos [16, 15].

Traditional methods usually estimate motion vectors or optical flow between frames, and do the interpolation along these flow maps [2, 33, 37], which are computationally expensive and time-consuming awing to the dense correspondence. Phased-based methods [30, 18] tried to deal with the computational issue by encoding small motions in the phase shifts of pixel's color change, but getting poor performance for frames with larger motions. *Convolutional neural networks* (CNNs)-based methods are thus introduced for more efficient and effective models, and have shown out some promising results. In spite of methods doing optical flow prediction with CNNs [1, 5, 6, 9, 27, 28, 32], there is recently a trend to directly generate the interpolated frames [20, 21, 15], combining motion prediction and pixel generation into a single step and avoiding the requirement for the

doi:10.6342/NTU201802624

hard-to-get, dense correspondence ground-truth. They could tackle with the challenging part for frame interpolation, including the complicated set of phenomena (the rapid movement of objects, the occlusion, the change of scene lighting, and the movement of cameras) in real scenes, with the regional consideration of the previous and next frames. Despite the encouraging progress, video frame interpolation still remains challenging in the area with large gradients or motions, better solutions are in demand consequently.

In this work, we proposed to integrate frame interpolation with a classical concept *cycle consistency*, which is widely used in many computer vision and feature matching algorithms [39, 40]. We address that interpolated frames are more reliable if we could further utilize them to synthesize high-quality output frames. Note that this concept offers an additional auxiliary loss term besides traditional minimization of l_1 or l_2 norm between the synthesized frame and the ground truth. l_2 norm directly maximizes PSNR but results in blurry output, while l_1 produces sharper results. But none of these losses can facsimile human perception. While our cycle consistency loss forces networks to produce realistic and sharp images, they can further be used as inputs for frame interpolation. As pointed out in [41], by coupling the original mapping G with an inverse mapping F and using the cycle consistency loss to enforce F(G(X)) $\approx$ X, we would get a more convincing result image G(X). We extend this idea from the image to image transition topic to the frame interpolation field. As far as our knowledge, we are the first to utilize cycle consistency in frame interpolation methods.

The training would be separated into two stages, as shown in Figure 1.1. In the first stage, we would only train the simple model to use I_0 and I_2 to interpolate I_1. In the second stage, we construct the cyclic model, copying the model weight from the pre-trained model to each training block. In this cyclic model, we form a *cycle consistency loss* to confirm the consistency of the second-stage output frame and the original input frame, in additional to the original reconstruction loss for the interpolated frame. In the experiments, we illustrate the proposed *cycle consistency loss* by applying it to one of the current state-of-the-art models [15]. The resulting interpolated frames' PSNR on UCF101 dataset [24] improved from 35.89 db to 36.71 db, which is an enhancement of 0.82 db

doi:10.6342/NTU201802624

compared with [15], for the testing set provided by [15]. We also compare the results of our method on the high-quality video (960 × 540), "See You Again" from Wiz Khalifa, which is used in the experiment of the other state-of-the-art method [21], and improved the PSNR from 39.69 db to 40.56 db, which is an enhancement of 0.87 db compared with [15].

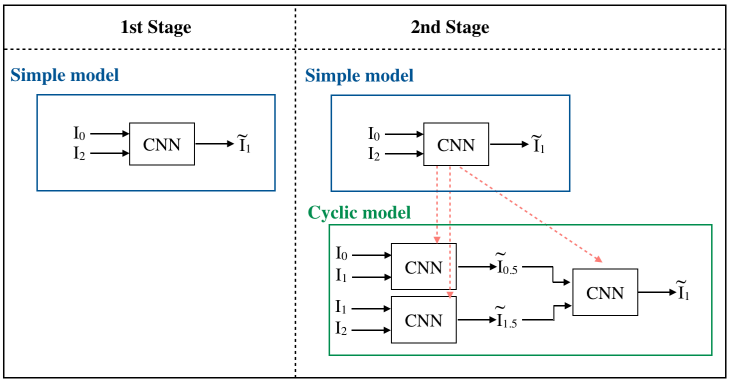


Figure 1.1: The training procedure for the proposed cycle consistency loss. First(1st) Stage: train a single interpolation network. Second(2nd) Stage: do the training to ensure the consistency between the output frame of the cycle model and the original input frame.

While applying our *cycle consistency loss* on the baseline model [15], we make an assumption that the flow magnitude should be half of the original if the interval between two consecutive frames becomes half of the original. According to this prior knowledge, we form another loss term to enforce that the flows generated by the three CNNs in the cyclic model would be half of the magnitude in the simple model. This loss term lead to a boost for our model in the PSNR from 36.71 db to 36.85 db on UCF101, and 40.56 db to 40.61 db on the high-quality video "See You Again".

To tackle with the aforementioned unpleasing results in image areas with great gradients, we add a more high-level information accompanied with input frames as guidance for the training. Recent study [19] has shown that the context information could contribute greatly on the performance of interpolation. In our work, we employ a pre-trained edge detection neural network [34], getting the edge maps derived from it, to guide the training of the network with *cycle consistency loss*. This step could further improve our model performance from 36.85 db to 37.00 db on UCF101 dataset, and from 40.61 db to 41.42 db on the high-quality video "See You Again".

doi:10.6342/NTU201802624

Chapter 2

Related work

Video frame interpolation is one of the classic problems in computer vision and video processing. In this chapter, we review a few relevant works, including traditional methods and methods based on convolutional neural networks. We would also review some papers related to the cycle constraint since our method is inspired by the cycle consistency concept.

2.1 Video Frame Interpolation

2.1.1 Traditional methods

Traditional methods regarding to this topic can be classified into 2 main types, which are motion-based methods and phase-based methods. Motion-based methods first estimate dense motion correspondences between two consecutive input frames using methods such as stereo matching or optical flow algorithms. Then, they interpolate one or more middle frames according to the estimated correspondences [2, 33, 37]. The performance of these methods often depends on the generating optical flow and the technical skill to do flow interpolation. However, this approach requires the optimization of a complex objective function and is often computationally expensive. Learning is often limited to a few parameters [14, 23, 25]. With today’s trend that people tend to see and make videos with higher resolution, the main challenge for interpolations would become the efficiency.

doi:10.6342/NTU201802624

Phase-based methods mainly deal with the efficiency issue. They encode small motions in the phase shift of an individual pixel's color to reduce the complexity for interpolation [30]. Meyer *et al.* extended this concept to accommodate large motion by propagating phase information across oriented multi-scale pyramid levels using a bounded shift correction strategy [18] and showed out good performances. However, these methods could not be widely used since the spatial displacement which can be encoded in the phase information is highly limited, that is, high-frequency details might be lost when facing video with large inter-frame changes.

2.1.2 Methods based on convolutional neural networks

Convolutional neural networks (CNNs) have gotten much success in the computer vision field these days. To deal with the interpolation task with CNNs, we can separate the methods into two types: one is using CNNs to do the prediction of optical flow; the other is directly generating the predicted middle frames.

There are lots of work using CNNs to do the prediction of optical flow, including both supervised [1, 5, 6, 9, 27, 28, 32] and unsupervised methods. Supervised methods use the CNNs to do the prediction of optical flows between two consecutive frames. They usually do the training with dense correspondence ground-truth, which is hard to obtain in the real world. An unsupervised approach [16] uses a CNN to predict optical flow by synthesizing interpolated frames, and then inverting the CNN. However, since their end-goal is to generate optical flow, they do not numerically evaluate the interpolated frames. The interpolated frames come out to be quite blurry.

Another branch in video frame interpolation on CNNs is to directly generate the frames. These methods do not need the dense correspondence as ground-truth. There are a number of papers that use CNNs to directly generate images [8] and videos [29, 36]. However, blur is often a problem for these generative techniques, since natural images follow a multimodal distribution, while the loss functions used often assume a Gaussian distribution. Ziwei at al. [15] tried to solve this blurring problem by copying coherent regions of pixels from existing frames with the use of an optical flow layer in the network. They could

doi:10.6342/NTU201802624

achieve relatively clear interpolated frames and get the flow unsupervisedly, but there's space for improvement in the quantitative result. Our method is inspired by their method. Other methods [20, 21] combine motion estimation and frame synthesis into a single convolution step by estimating spatially-varying kernels for each output pixel and convolving them with input frames to generate a new frame. However, these methods need large kernels to handle large motion, they cannot generate all pixels in a high-resolution frame since there would be memory limitation.

2.2 Cycle Constraint

The idea of using cycle constraint as a way to regularize structured data has a long history. In the language domain, using "back translation and reconsiliation" to verify and improve the accuracy for translations is a technique used by human translators [3]. For visual tracking, there is also a trick to get better result with enforcing simple forward-backward consistency [26]. More recently, higher-order cycle consistency has been used in many other vision tasks, including structure from motion [38], 3D shape matching [10], co-segmentation [31], dense semantic alignment [39, 40], depth estimation [7] and image-to-image translation [41].

Of all the above work, Zhou et al. [40], Godard et al. [41] and Zhu et al. [41] are most similar to our work, since they use the concept "cycle consistent" to regularize supervised CNN training. For the work of Zhu et al. [41], they couple their initial mapping G : $X \rightarrow Y$, which is highly under-constrained, with an inverse mapping F, using the cycle consistency loss to enforce $F(G(X)) \approx X$ to get a more promising mapping of G. The difference between our approach and theirs is that we are not enforcing the dual direction mapping between two mapping functions, but to use cycle consistency to ensure that we can gain high quality interpolations with interpolated frames as input into the same model.

doi:10.6342/NTU201802624

Chapter 3

The proposed approach

Our approach is introduced in this chapter. Given a target CNN architecture, our main goal is to utilize the simple concept that the interpolated frames are more reliable if we could synthesize good enough results from them to enhance the performance of the model for frame interpolation.

We would first give a quick sketch for the whole model architecture. Then, we describe how each component works in our model, including how to apply the special loss term - Cycle Consistency Loss, how edge-guided training works and how to formulate the motion linearity loss.

3.1 Full Model

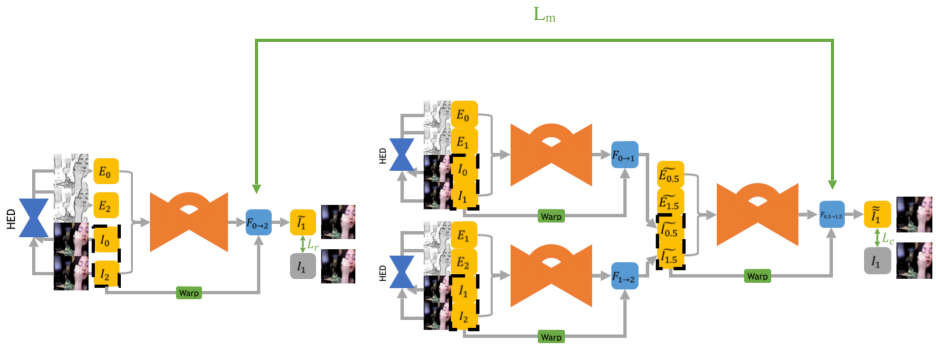


Figure 3.1: Full model architecure.

The full architecture of our model is shown in Fig3.1. There are three crucial parts in

doi:10.6342/NTU201802624

our model: the cycle consistency loss, motion linearity loss and edge-guided training. The cycle consistency loss is inspired by the idea that the interpolated frames are more reliable if we could further synthesize good enough results from them. The motion linearity loss is to ensure the flow linearity when the flow between frames are pretty small. There would be a further improvement in the performance of the model with edge-guided training, inspiring by the different performances in different gradient magnitudes. We would dig into the details for each part in the following sections.

The loss for our model is stated as below, where L_r means the reconstruction loss (l_1), L_c means the cycle consistency loss, and L_m means the motion linearity loss:

$$L_{total} = \lambda_r L_r + \lambda_c L_c + \lambda_m L_m$$

3.2 Cycle Consistency Loss L_c

According to our observations, traditional image quality assessment metrics such like *Peak Signal to Noise Ratio* (PSNR) alone could be hard to represent the effect of interpolation since some frames with lower PSNR might be clear and pleasant for audience than the frames with hight PSNR, as shown in Figure 3.2. Thus, we propose the Cycle Consistency Loss L_c —a loss for any end-to-end training CNN models to not only ensure the quantitative result of interpolated frames, but also improve visual quality of video frame synthesis. This loss could be easily applied to any CNN model, especially easier for those directly generating the middle frames. The training data is the same as the target model we'd like to improve, usually the triplets of consecutive video frames.

Figure 3.2: Details in the right image preserve more, with lower PSNR. (Left: ground truth; middle: PSNR=23.38 db; right: PSNR=21.48 db)

doi:10.6342/NTU201802624

The training process needs to be split into two stages, which is to pre-train the "simple model" in the first stage and to jointly fine-tune the "simple model" and "cyclic model" in the second stage.

The first stage training. During the first-stage training, we construct the "simple model", which is a modified model from Deep Voxel Flow (DVF) [15], as shown in the Figure3.3. For this model, two frames in the triplets of consecutive video frames are provided as inputs and the remaining frame is used as the reconstruction target. With the simple L_1 loss (L_r, for which r stands for "reconstruction"), the model learns to synthesize the in-between frame from the input frames, by warping the input frames with the generated voxel flow and mask in the intermediate stage.

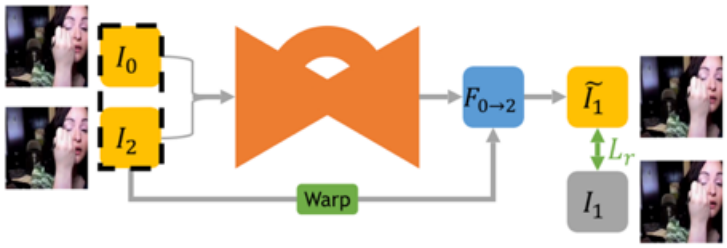


Figure 3.3: Simple model (The modified model from Deep Voxel Flow[15]), where I_0 and I_2 refer to the first frame and the third frame in the triplets, $F_{0\to2}$ refers to the voxel flow between the two input frames, I_1 refers to the ground truth, and $\widetilde{I}_1$ refers to the interpolated frame.

The second stage training. After pre-training the "simple model" to the convergent state, we construct our second-stage model (Figure3.4), the "cyclic model", by copying the weight of pre-trained "simple model" three times, to apply the Cycle Consistency Loss. The Cycle Consistency Loss is to measure the difference between the second-stage interpolated frame $\widetilde{\widetilde{I}}_1$, which is synthesized from our first-stage interpolated frames $\widetilde{I_{0.5}}$ and $\widetilde{I_{1.5}}$, with the ground truth I_1. The four CNN blocks (the orange blocks shown in Figure3.4), are sharing the same weight during the second-stage training, with the two loss terms, L_r and L_c, considered jointly. The two loss terms are shown as below,

$$L_r = ||\widetilde{I}_1 - I_1||_1, L_c = ||\widetilde{\widetilde{I}}_1 - I_1||_1$$

doi:10.6342/NTU201802624

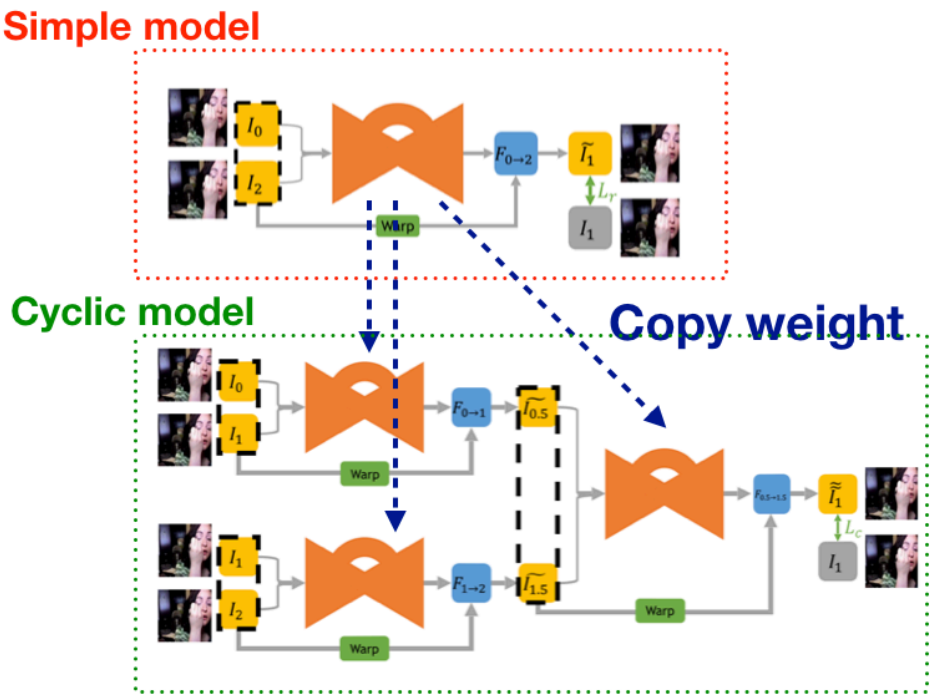


Figure 3.4: Our Second-stage model, including the "simple model" and the "cyclic model", to apply the Cycle Consistency Loss.

3.3 Motion Linearity Loss L_m

We assume that the motion between the two consecutive frames would be linear when the time interval between them are relatively small. In our aforementioned two-stage training procedure, the time intervals between the first-stage input frames are set to be twice of those between the second-stage input frames, in order to apply the cycle consistency loss L_c. Integrating the above two prior knowledge, the generated flow magnitude of the simple model should be twice of those generated by the cyclic model, as shown in Figure3.5 (The schematic flow maps are just for the demonstration of our idea, not the real generated flow maps). The motion linearity loss L_m could thus be formulated as below, where F means the flow map and the subscripts mean the flow is from which frame to which,

$$L_m = ||F_{0\to 2} - 2 \cdot F_{\widetilde{0.5}\to\widetilde{1.5}}||_2^2$$

doi:10.6342/NTU201802624

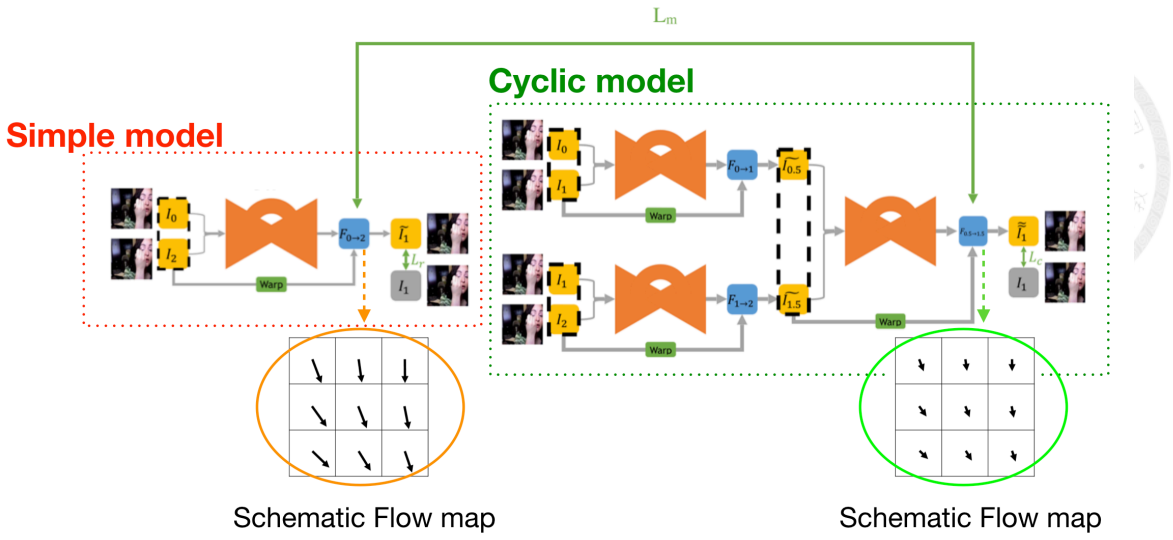


Figure 3.5: Motion Linearity Loss. The blue blocks represent estimated flow maps.

3.4 Edge-guided Training

To analyze the performance of past interpolation models and gain some insight for where we could further enhance, we have done some statistical analysis on those models. Shown as Figure3.7, we here use the DVF [15] model and the SepConvs [21] model as examples. We tested the two models on the UCF101 testing set, and divided the pixels of the frames in the testing set into ten groups according to the gradient magnitude of each pixel, getting the average performance (mean square error) for each gradient magnitude bin. We observe that the mean square error between the interpolated results and the ground truth turns out to be larger as the gradient magnitude of the pixel becomes larger. The edge-guided training idea is inspired by this observation. We believe that with the help of the edge information, the model would learn extra knowledge with only the color images themselves.

After several experiments for the generation of the edge information, including traditional methods [4, 12, 17] and CNN-based methods, we choose to use the CNN-based method, Holistically-Nested Edge Detection (HED) [34]. The benefit to use a CNN-based method is that we could combine the training of the network with our cycle consistency loss since it is fully differentiable. As shown in Fig3.6, we get the edge maps E_0 and E_2 for the two input frames I_0 and I_2, and then concatenate them with the color images to

doi:10.6342/NTU201802624

form the eight-channel final input, composed with 3(color image channel)x2(frames) and 1(edge map channel)x2(frames), for our edge-guided training procedure. The output and loss term for this training procedure is the same as the original, the predicted middle frame $\widetilde{I}_1$ and the L_r loss.

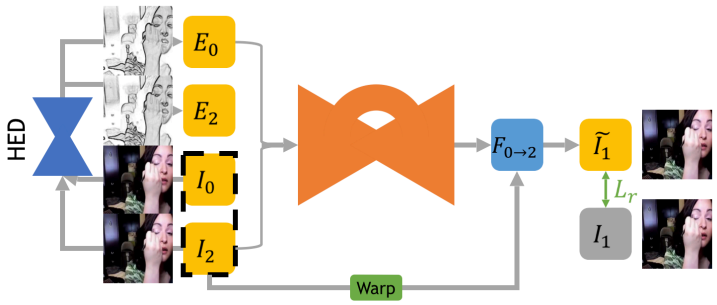


Figure 3.6: Edge-guided training with HED [34].

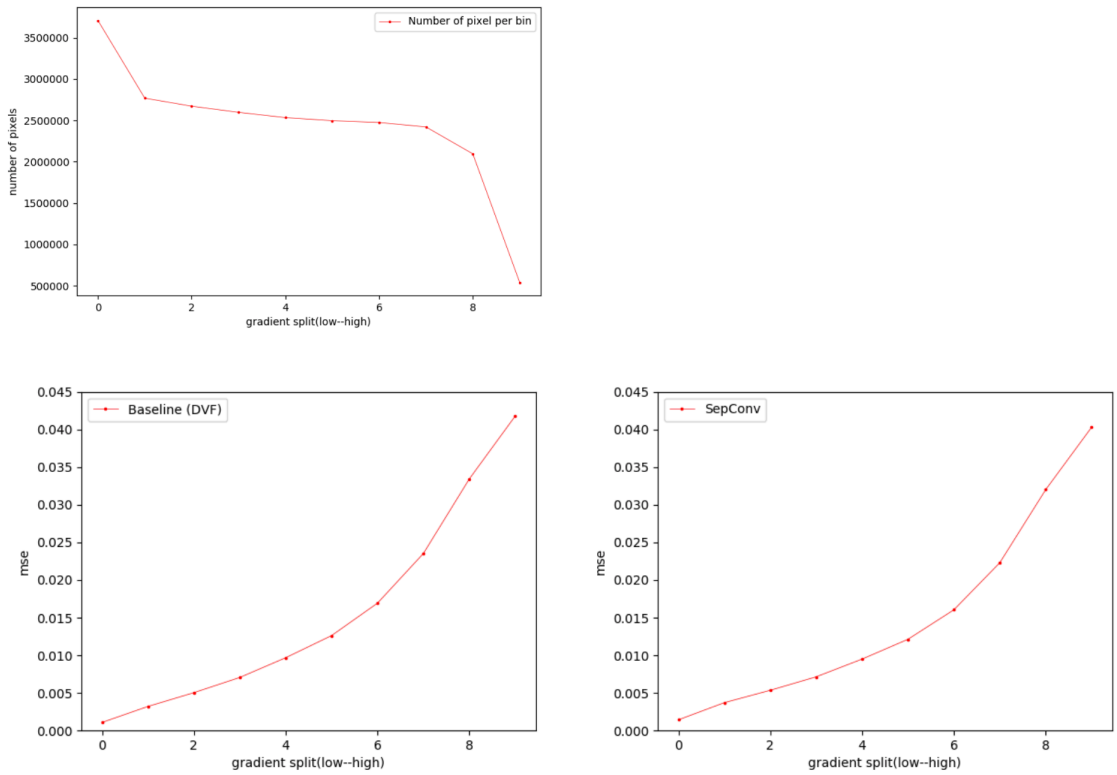


Figure 3.7: Analytical results on the UCF101 testing set. (Up: The number of pixels in each gradient bin; Lower right: The statistical result of DVF; Lower right: The statistical result of SepConv.)

doi:10.6342/NTU201802624

Chapter 4

Implementation details

The implementation and training details are demonstrated in this chapter. First, we would dig into the architecture of our base model (the orange block in all the previous figures), which is an extension from [15], for applying the *cycle consistency loss*. Then, we show the training environment of our approach.

4.1 Implementation of the base model

Our full model is constructed by four orange blocks, shown as Figure3.1, while each block refers to a base model, shown as Figure4.1. For the base model, we adopt the U-Net architecture [22], which is a fully convolutional neural network, consisting of an encoder and a decoder, with skip connections between the encoder and decoder features at the same spatial resolution. For both the encoder and decoder, we have 3 hierarchies. In the encoder, there are one convolutional and one ReLU layer in each hierarchy, and an average pooling layer with a stride of 2 at the end of each hierarchy except the last one, to decrease the spatial dimension. In the decoder part, a bilinear upsampling layer is used at the beginning of each hierarchy to increase the spatial dimension by a factor of 2, followed by one convolutional and one ReLU layer.

There would be an 3-channel intermediate output, which are the flow map of x and y direction and the blending mask respectively.. The final output, which is the interpolated frame, would be generated by the model intermediate outputs, with warping the two input

doi:10.6342/NTU201802624

frames with the generated flow maps and blending them with the mask. Since the network need to take the flow computation into account, it is important to use large filters in the first few layers of the encoder to capture long-range motion. We therefore use 5 x 5 kernels in the first two convoluional layers and 3 x 3 in the rest layers.

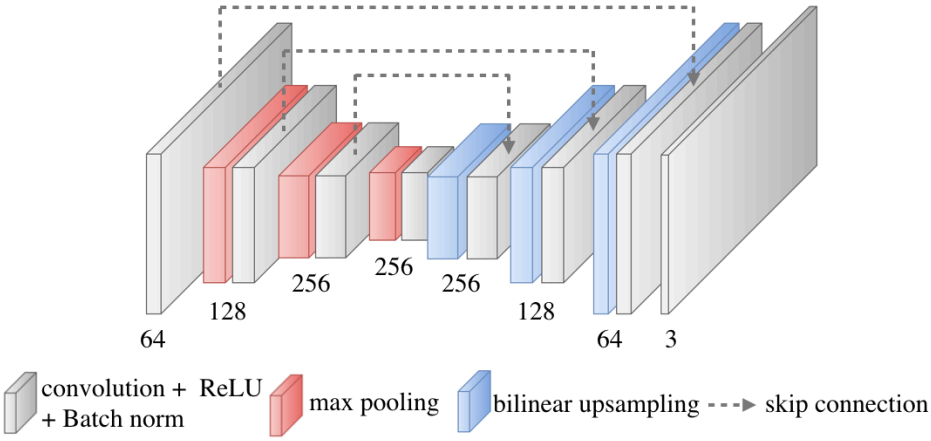


Figure 4.1: Base model architecture: U-Net.

4.2 Training details

Due to the two stage training for our full model, given input images I_0 and I_2, the intermediate frame I_1, our prediction of the intermediate frame $\widetilde{I}_1$, the prediction with synthesized input frames $\widetilde{\widetilde{I}}_1$, and the predicted flow $F_{n\to m}$, our loss functions are formulated as below:

$$L_{1^{st}-stage} = L_r \tag{4.1}$$

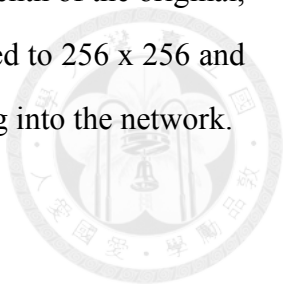
$$L_{2^{nd}-stage} = L_r + \lambda_c * L_c + \lambda_m * L_m \tag{4.2}$$

where $L_r = ||\widetilde{I}_1 - I_1||_1$, $L_c = ||\widetilde{\widetilde{I}}_1 - I_1||_1$, $L_m = ||F_{0\to 2} - 2 * F_{\widetilde{0.5}\to\widetilde{1.5}}||_2^2$. The weight has been set empirically using the validation set, and we will discuss this in the next chapter (the ablation study part). Every component of our network is differentiable, thus our model is end-to-end trainable.

Our network is trained using the Adam optimizer [13] with batch size 8 on videos from the UCF-101 training set [24]. The learning rate is initialized to be 0.0001 in the first-stage

doi:10.6342/NTU201802624

training. In the second-stage training, we set the learning rate to one tenth of the original, 0.00001. For the preprocessing of the input data, each frame is resized to 256 x 256 and the pixel values are normalized into the range of [1, -1] before feeding into the network.



doi:10.6342/NTU201802624

Chapter 5

Experimental results

The performance of the proposed cycle consistency loss is evaluated in this chapter. We also demonstrate the effect of combining the edge information and the prior knowledge of flow linearity into training. We first describe the adopted datasets, including UCF101 [24], Middlebury [2] and a high-quality video (960 × 540), "See You Again" from Wiz Khalifa. We mainly use UCF101 for the training, and for the performance measure, we use the test set of UCF101 released by [15] and the rest two aforementioned datasets. We would then give the details about the construction and initialization of our network. Afterwards, The quantitative results are reported and analyzed, including the performances of applying the cycle consistency loss, the edge-guided training, and the consideration of motion linearity individually or jointly, and the comparison of our approach to the existing approaches. Finally, we present the visual results of our interpolated frames in the three datasets mentioned previously, comparing with the results of state-of-the-arts.

5.1 Dataset

5.1.1 UCF101

This dataset is proposed originally for action recognition. It is formed of realistic action videos collected from YouTube, with 13320 videos in 101 action categories. The largest diversity in these videos, including camera motion, object appearance and pose, ob-

doi:10.6342/NTU201802624

ject scale, viewpoint, cluttered background, illumination conditions, etc, makes the dataset challenging not only for action recognition, but also the topic we want to deal with: video frame interpolation. Thus, it is suitable for us to use as training dataset. Figure5.1 shows one image frame from each category in the dataset.

Figure 5.1: Total 13320 videos in 101 action categories of the UCF101 dataset.

To form the training dataset, we first use ”ffmpeg” to extract frames from every video in the train_list1.txt offered by the UCF101 website at a frame rate of 30 fps. Since there might be consecutive frames with no or little motion on it, which might be meaningless to use them for the training, we need to do some preprocessing work to filter them out. We calculate the PSNR between two consecutive frames, while lower PSNR means lower similarity and thus more obvious motion between frames, and only keep the lower PSNR triplets to form the training set of approximately 280,000 triplets. Also, because of the training memory limitation, we resize all the frames to a resolution of 256 x 256 before using them to train our network.

5.1.2 Middlebury

This dataset is mainly for optical flow prediction. The testing sets consist of eight frames from each of the 12 categories, divided into four types, which are hidden texture, synthetic, stereo, high-speed camera videos. Since this dataset doesn't announce the ground truth for us to do the evaluation ourselves, we upload our testing results for the testing. Shown as Figure5.2, this dataset only accept eight categories for interpolation evaluation.

doi:10.6342/NTU201802624

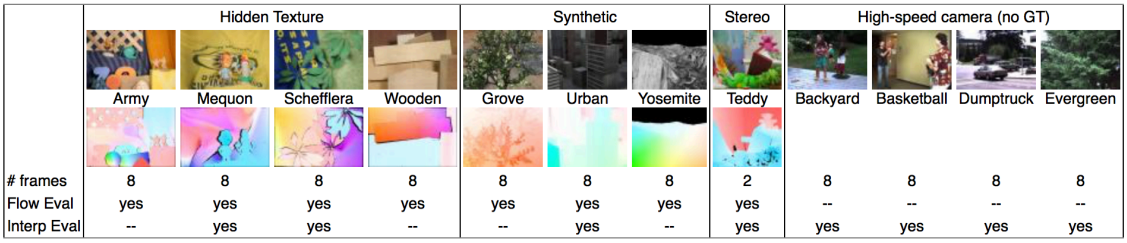


Figure 5.2: One frame from each of the 12 categories in the Middlebury dataset.

5.1.3 High-quality Video: "See You Again" from Wiz Khalifa

We also evaluate our model on the video with high resolution. The reason we choose this video is that the state-of-the-art method [21] have performed testing result on this video. We use the same testing setting as [21], sampled the video at the frame rate 23.98 fps to get 2846 frames with a resolution at 540 x 960 as input to generate 2845 interpolated frames without any fine-tuning.

5.2 Experimental setup

In our experiment, we pre-train our first-stage model using the UCF101 training dataset with simple L_1 (L_r) loss, with random initialization of the weight. Once the first-stage model converges, we copy the weight of the model three times to construct the second-stage model for applying the cycle consistency loss (L_c), as shown in Figure1.1. In the second stage, we fine-tune the model with L_r loss combining the cycle consistency loss L_c and motion linearity loss L_m, with learning rate on tenth of the original in the first stage.

For the performance evaluation, we test our model on three different datasets, including real scene videos, synthetic videos, and high-quality videos. For UCF101, in every triple of frames, the first and third ones are used as inputs to predict the second frame using 379 sequences provided by [15]. For Middlebury, we submit our video interpolation results of eight sequences to its evaluation server. For "See You Again", we directly test the performance without doing the fine-tuning.

doi:10.6342/NTU201802624

5.3 Ablation Study

We conduct the ablation study with the model trained on UCF101 training set, and test the performance on UCF101 testing set provided by [15]. We adopt our cycle consistency loss at [15], thus consider it as our baseline, and its performance (PSNR) on UCF101 is 35.89 db. We then compare the results of the baseline with the results of applying each component, including the cycle consistency loss, motion linearity loss, and edge-guided training, to demonstrate the effectiveness of each component in our training procedure.

Parameter tuning. We do the model analysis with the proposed loss function in equation 4.2, consisting of three loss functions, to decide the weighting among these three loss terms. Except for reconstruction loss L_r, the other two functions, L_c and L_m, are associated with leading parameters, i.e. λ_c and λ_m, respectively. We perform sensitivity analysis of the two parameters. First of all, the reconstruction loss L_r is employed with its leading parameter set to 1. We add the cycle consistency loss L_c for ensuring the predictions from interpolated frames to be as similar as the original input frames. The performance of our approach by varying the value of the corresponding parameter λ_c is shown in Figure5.3 (a). It can be observed that L_c is crucial, since the performance gain by changing λ_c from zero to a positive value is significant. We empirically set λ_c to 0.75. Then, the third loss is included for regularizing the areas with large motion. The performance of our approach with different values of parameter λ_m is similarly reported in Figure5.3 (b), and λ_m is fixed to 0.01 according to the experiment.

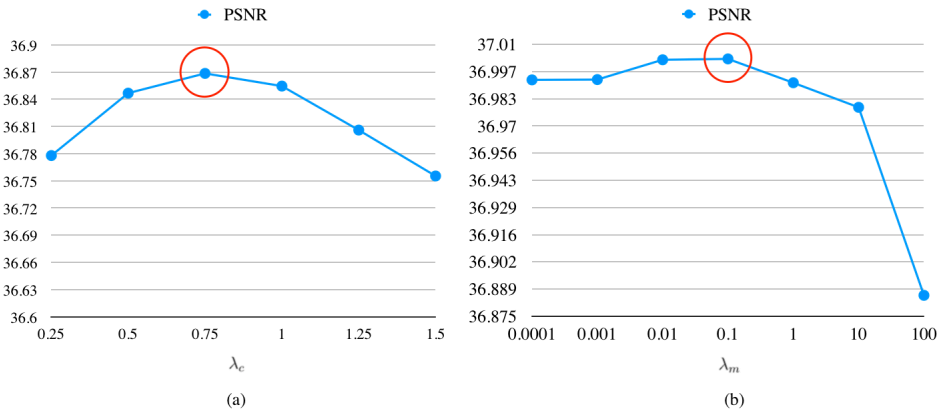


Figure 5.3: The performances of our approach in PSNR with different vales of parameter (a) λ_c and (b) λ_m.

doi:10.6342/NTU201802624

Effectiveness of Cycle Consistency Loss. We first test whether jointly learning the first stage and second stage interpolation improves the video interpolation results. Intuitively, ensuring the predictions from interpolated frames to be as similar as the original input frames might enforce the network to generate more solid results.

We compare the results of the model trained with or without the cycle consistency loss. Since we do the weight-sharing of all the CNN models in our two-stage training procedure, all the four models in our full architecture are the same model. Thus, in the testing stage, we only use 1 CNN model (the simple model) to do the testing. The first and second row in Table5.2 clearly demonstrate that the performance would be improved 0.82 db in the UCF101 testing set, from 35.89 db to 36.71 db. Column (b) and (c) in Figure5.8 demonstrate the visual differences for applying the cycle consistency loss, as some loss of content would be recovered with the regularization of this cycle consistency concept.

In additional to the better results qualitatively and quantitatively, there is another benefit accompanied with the cyclic frame generation. Since we reuse the training data in our two-stage training model, we could utilize the data in a more efficient way. As shown in Table5.1 and Figure5.4, we test the performance of two models, which are the model trained with simply L_1 loss and the model trained with our cycle consistency loss L_c, on two independent datasets, which are the UCF101 testing set provided by [15] and the high-quality video: "See You Again". The training data is randomly sampled from our full UCF101 training dataset (approximately 280,000 triplets) proportionally to the data size. We can see that we would maintain the performance of our model (trained with the cycle consistency loss) even with fewer training data, while the performance of the model trained without the cycle consistency loss would drop a lot when lack of training data.

Dataset	UCF101				Video: "See You Again" 2,841 samples at 960 × 540			
Size of Training Data	DVF	PSNR Drop	Ours (+Cycle +Flow)	PSNR Drop	DVF	PSNR Drop	Ours (+Cycle +Flow)	PSNR Drop
x 1	35.98	X	36.85	X	39.69	X	40.60	X
x 0.1	35.71	-0.27	36.83	-0.02	39.16	-0.53	40.47	-0.13
x 0.01	35.43	-0.55	36.70	-0.15	38.18	-1.51	39.88	-0.72
x 0.001	34.42	-1.56	36.10	-0.75	35.75	-3.94	38.13	-2.47

Table 5.1: Evaluation of the performance for models with different training data size. The reported values are the average PSNR (db) for the testing results. (*Cycle*: Cycle Consistency Loss; *Flow*: Motion Linearity Loss.)

doi:10.6342/NTU201802624

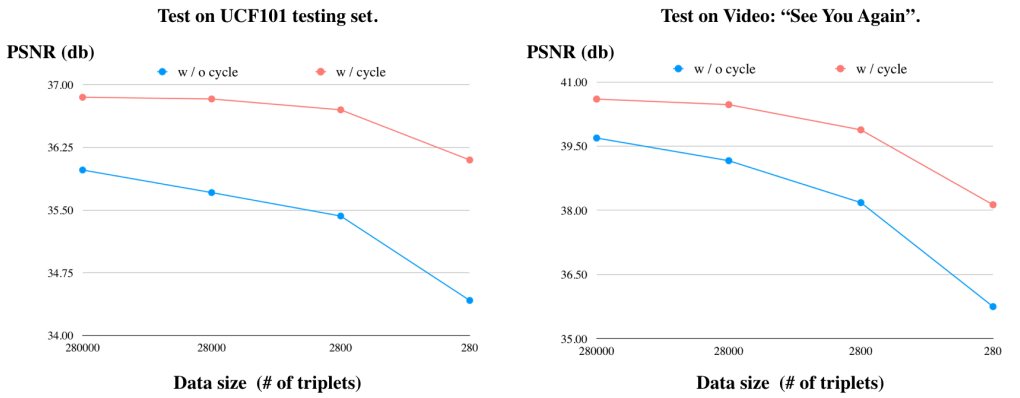


Figure 5.4: Evaluation of the performance for models with different training data size.

Effectiveness of Motion Linearity Loss l_M. Here we add another loss term into training. With the assumption that the optical flow between frames in short time interval could be approximated to be linear, the flow we got from the simple model should be twice the magnitude of those generated by the cyclic model since the time interval between the two input frames for the cyclic model is half of those for the simple model. By applying this loss term, we can ensure that the interpolation process for frames with larger motion would have a more simple target, the half size of the original motion.

Taking into account the motion linearity concept, our model performance become better than just applying cycle consistency loss alone, as shown in the second and third row in Table5.2, improving from 36.71 db to 36.85 db, which is an improvement of 0.14 db. The visual results are shown Figure5.5 and Figure5.8 by comparing column (c) and (d). We can see that the interpolated frames would have more accurate predicted positions for objects in the frame.

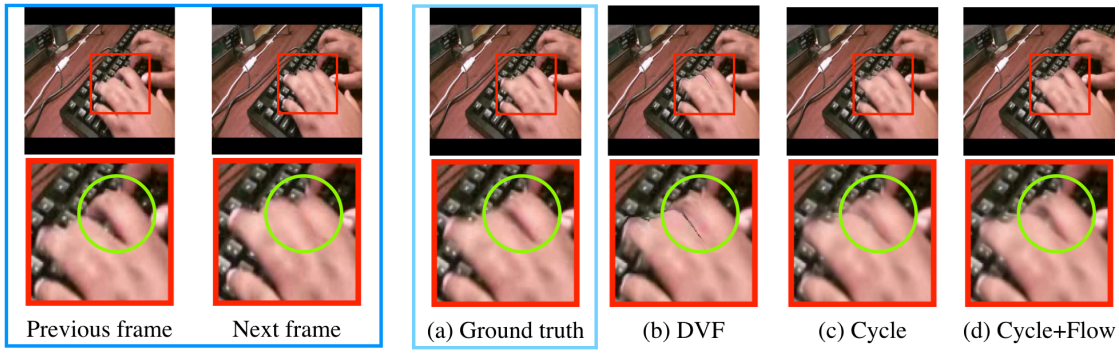


Figure 5.5: Visual results for model w/o and w/ motion linearity loss.

doi:10.6342/NTU201802624

Effectiveness of Edge-guided Training. According to our observations, as Fig3.7, frame interpolation usually have low performances in the strong gradient areas. When combining the input frames with the high-level information, the edge maps, they could serve as the guidance for the interpolating process, to boost the interpolations more when the gradient is larger, as Fig5.7. We could see in Table5.2, edge-guidance could further improve the performance, from 36.71 db to 36.86 db, which is an improvement of 0.15 db. Also, as shown in column (c) and (d) in Fig5.6 and in column (c) and (e) in Fig5.8, it would have sharper results than those without the edge-guided training.

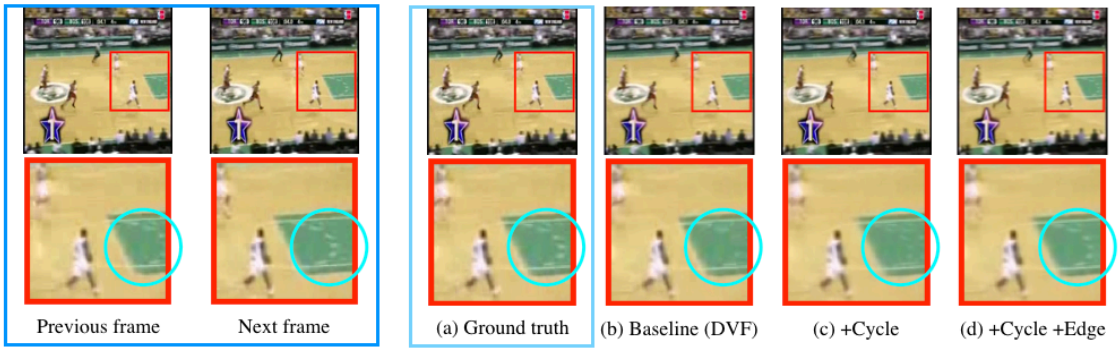


Figure 5.6: Visual results for model w/o and w/ edge-guided training.

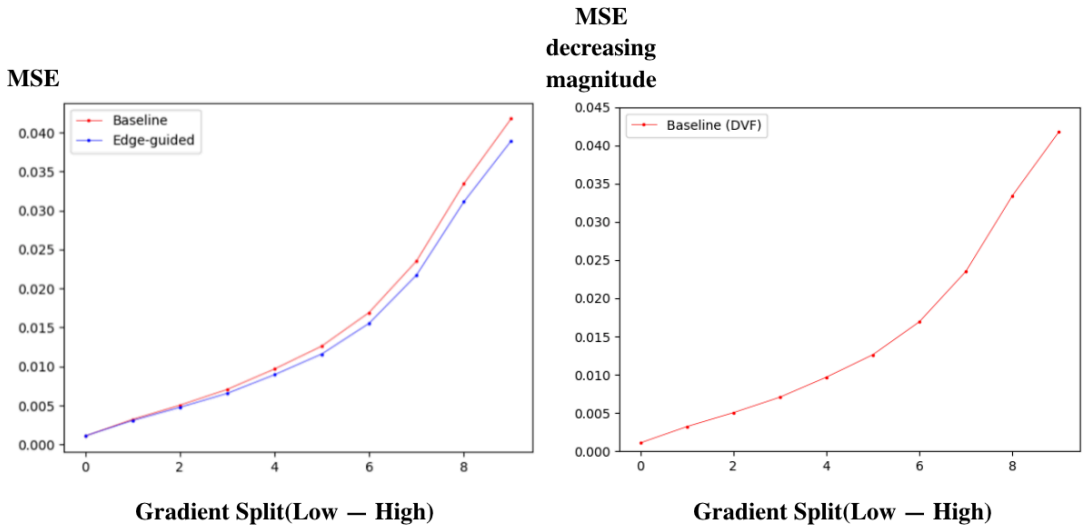
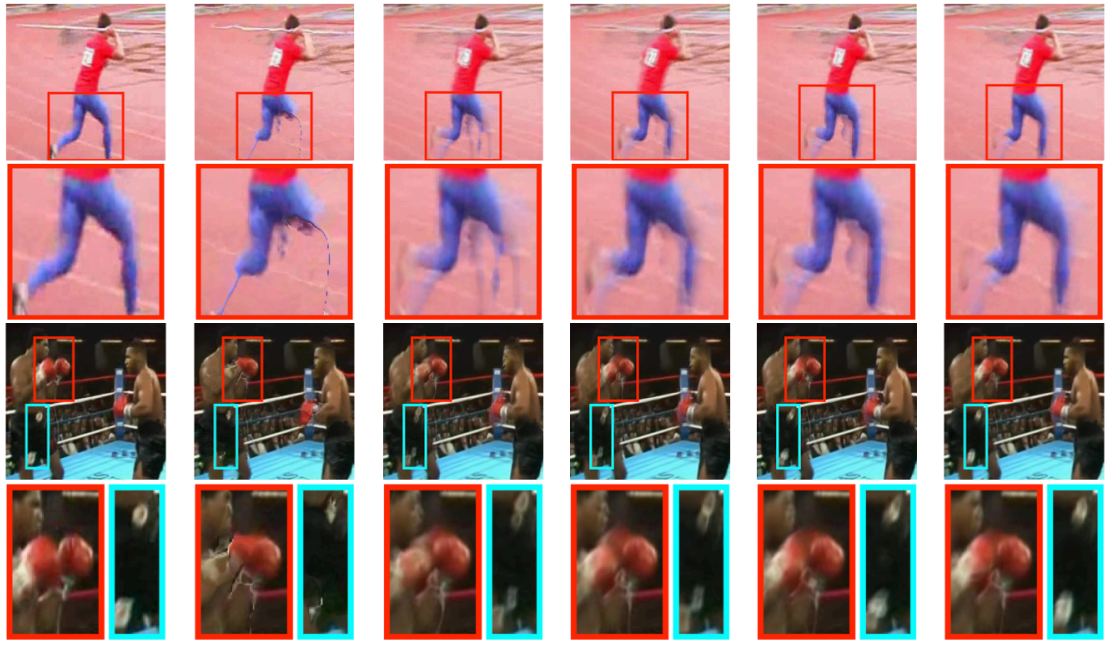


Figure 5.7: *Left*: The comparison of performance for baseline and edge-guided training. The statistics of mean square error (the lower the better) between interpolated frames and ground truth in areas with different gradient magnitude, with the greater value to the right. *Right*: The improvement with the edge-guided training. (The decreasing magnitude in MSE.)

doi:10.6342/NTU201802624



(a) Ground truth (b) DVF (c) Cycle (d) Cycle+Flow (e) Cycle+Edge (f) Full Model

Figure 5.8: Qualitative evaluation on UCF101.

	UCF101	
	PSNR	SSIM
Baseline (DVF [1])	35.89	0.945
+ Cycle	36.71 (+0.82)	0.950 (+0.005)
+ Cycle + Flow	36.85 (+0.96)	0.950 (+0.005)
+ Cycle + Edge	36.86 (+0.97)	0.952 (+0.007)
full model	**37.00** (+1.11)	**0.953** (+0.008)

Table 5.2: Evaluation of the effect of each component in our model. (*Cycle*: Cycle Consistency Loss; *Flow*: Motion Linearity Loss; *Edge*: Edge-guided Training.)

5.4 Comparison to state-of-the-art methods

In this section, we compare our approach mainly with state-of-the-art methods including separable adaptive convolution (SepConv) [21] and deep voxel flow (DVF) [15].

UCF101 dataset and Video: "See You Again". On UCF101, we compute all metrics, both PSNR and SSIM, using the motion masks provided by [15]. On Video: "See You Again", we evaluate our model with 2,841 samples from a high-resolution (960x540) video. The quantitative results are shown in Table5.4. Our model consistently outperforms DVF [15] and SepConv [21] in these two datasets. Some sample visual results could be found interpolation results on a sample from UCF101 can be found at Figure5.10.

doi:10.6342/NTU201802624

	UCF101		Video:"See You Again" 2,841 samples at 960 × 540	
	PSNR	SSIM	PSNR	SSIM
DVF [15]	35.89	0.945	40.15	0.958
SepConv [21]	36.49	0.950	41.01	**0.969**
ours	**37.00**	**0.953**	**41.42**	0.964

Table 5.3: Results on the *UCF101* dataset and high-quality video: "See You Again".

Middlebury dataset. The interpolation error (IE) scores on each sequence form the Middlebury dataset [2] are shown in Table5.4. We compare our model with the four top-performing models on the Middlebury dataset, including models directly generating the interpolated middle frames, which are Context-aware Synthesis (CtxSyn) [19], and Super SloMo (SuperSlomo) [11], and the model, MDP-Flow2 [35], where the interpolation algorithm [2] is coupled with its optical flow method. Our model achieves the best performance on 3 out of all 8 sequences. Particularly, the Urban sequence are generated synthetically and the Teddy sequence contains actually two stereo pairs. The performance of our model shows that we get the best results in the interpolation of real scene sequences, validating the generalization ability of our approach. Some sample visual results could be found interpolation results on a sample from Middlebury can be found at Figure5.11.

	AVERAGE			Mequon			Schefflera			Urban			Teddy			Backyard			Basketball			Dumptruck			Evergreen		
	all	disc.	unt.	all	disc.	unt.	all	disc.	unt.	all	disc.	unt.	all	disc.	unt.	all	disc.	unt.	all	disc.	unt.	all	disc.	unt.	all	disc.	unt.
Ours	5.42	8.23	2.30	2.56	4.25	1.36	3.53	4.85	1.88	4.68	5.45	3.36	5.99	7.19	3.44	9.69	12.1	**3.29**	**4.73**	**8.82**	**2.18**	**6.13**	**13.8**	**1.41**	**6.05**	**9.41**	**1.50**
CtxSyn	**5.28**	**8.00**	2.19	**2.24**	**3.72**	**1.04**	**2.96**	**4.16**	1.35	4.32	**3.42**	3.18	**4.21**	**5.46**	**3.00**	9.59	**11.9**	3.46	5.22	9.76	2.22	7.02	15.4	1.58	6.66	10.2	1.69
SuperSlomo	5.31	8.39	2.12	2.51	4.32	1.25	3.66	5.06	1.93	**2.91**	4.00	1.41	5.05	6.27	3.66	**9.56**	**11.9**	3.30	5.37	10.2	2.24	6.69	15.0	1.53	6.73	10.4	1.66
SepConv	5.61	8.74	2.33	2.52	4.83	1.11	3.56	5.04	1.90	4.17	4.15	2.86	5.41	6.81	3.88	10.2	12.8	3.37	5.47	10.4	2.21	6.88	15.6	1.72	6.63	10.3	1.62
MDP-Flow2	5.83	9.69	2.15	2.89	5.38	1.19	3.47	5.07	**1.26**	3.66	6.10	2.48	5.20	7.48	3.14	10.2	12.8	3.61	6.13	11.8	2.31	7.36	16.8	1.49	7.75	12.1	1.69
DeepFlow	5.97	9.79	**2.05**	2.98	5.67	1.22	3.88	5.78	1.52	3.62	5.93	**1.34**	5.39	7.20	3.17	11.0	13.9	3.63	5.91	11.3	2.29	7.14	16.3	1.49	7.80	12.2	1.70

Table 5.4: Evaluation on the *Middlebury* benchmark. *disc.*: regions with discontinuous motion. *unt.*: textureless regions.

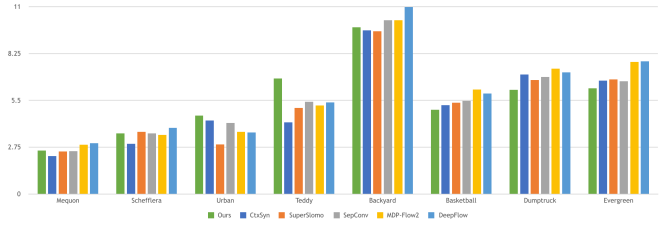
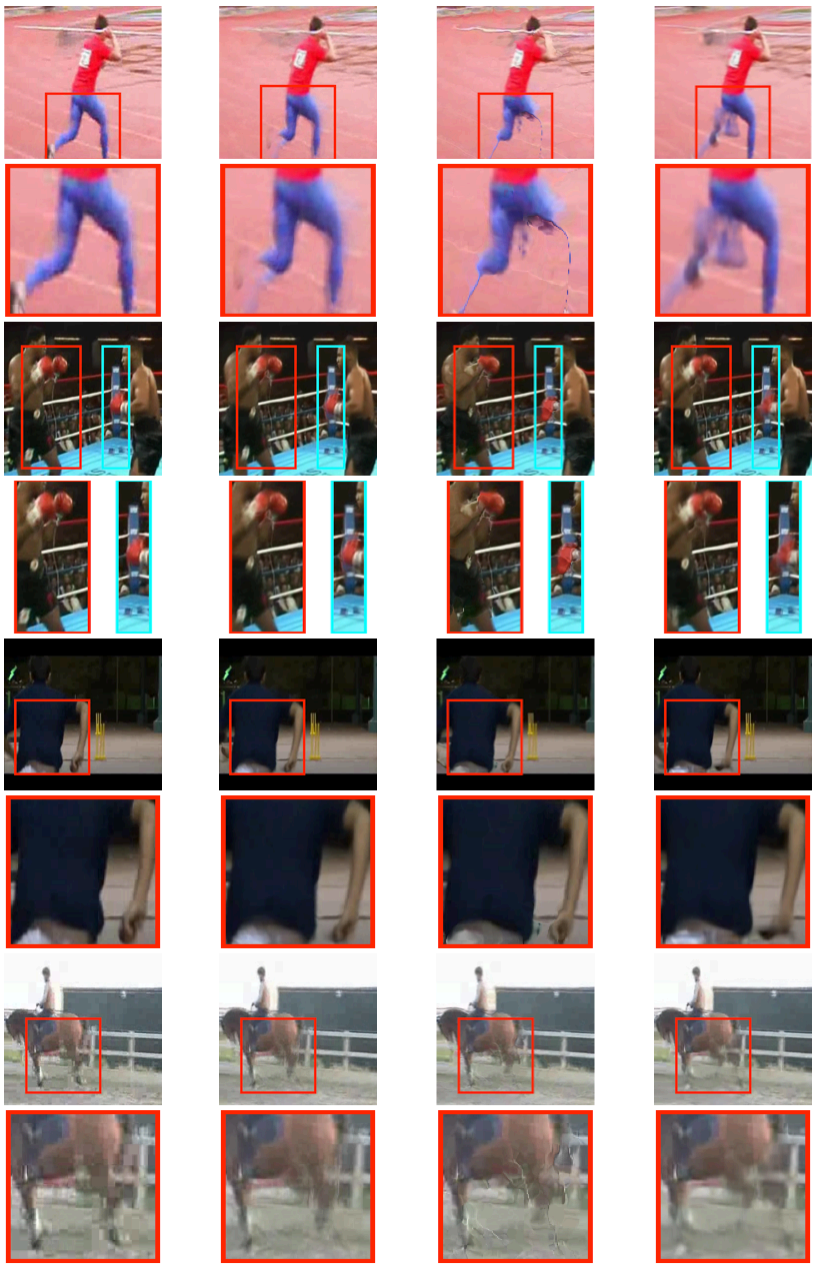


Figure 5.9: Quantitative evaluation on Middlebury dataset.

doi:10.6342/NTU201802624

5.5 Visual comparison

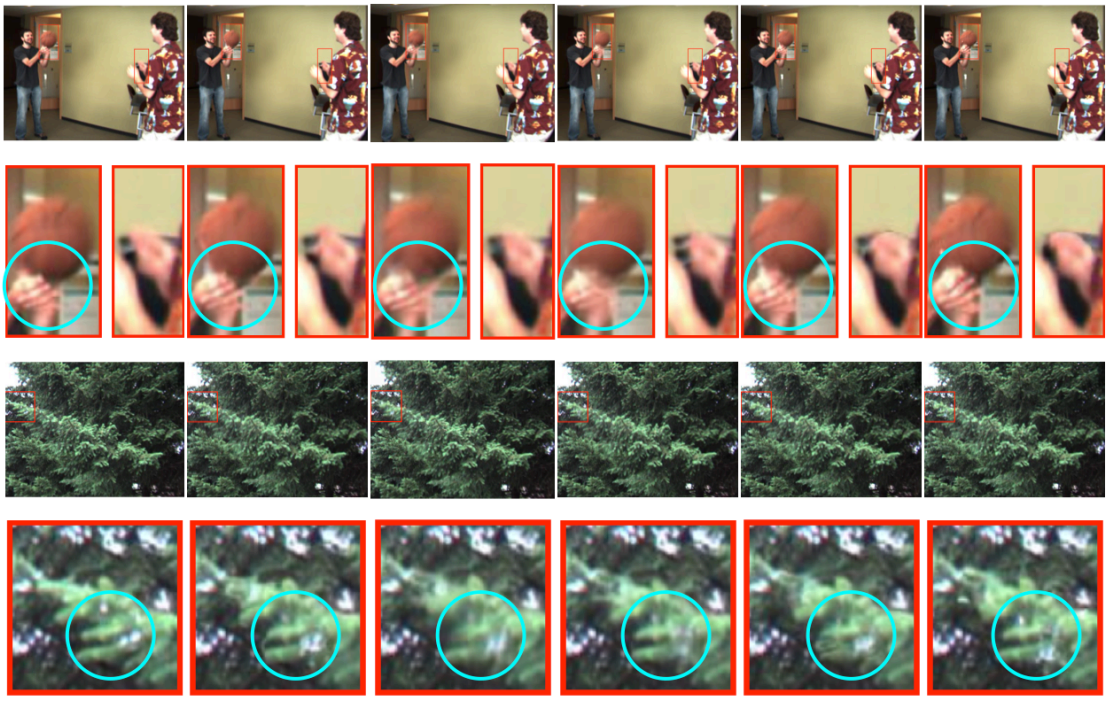
We show a comparison of our proposed approach with state-of-the-art video frame interpolation methods in Figure 5.11 and 5.10.



(a) Ground truth (b) Ours (c) DVF (d) SepConv

Figure 5.10: Qualitative evaluation on UCF101.

doi:10.6342/NTU201802624



(a) Ground truth (b) Ours (c) CtxSyn (d) SepConv (e) SuperSlomo (f) DeepFlow

Figure 5.11: Qualitative evaluation on Middlebury.

doi:10.6342/NTU201802624

Chapter 6

Conclusions

We have presented a new loss term, the cycle consistency loss, on video frame interpolation, to integrate the interpolation with the classical concept *cycle consistency*. The loss term and the architecture behind it enable the fully utilization of the training data, not only to enhance the results of interpolation, but also to maintain the performance with less training data than previous works. Accompanied with the cyclic video frame generation, we perform two other extensions, edge-guided training and motion linearity loss, to deal with the relatively low performance when applying previous methods in great gradient and large motion areas.

We evaluated these three components on the UCF101 dataset, Middlebury dataset, and the high-quality video: "See You Again". The experimental results show that it can reach the state-of-the-art performance in both UCF101 and high-quality video: "See You Again". For Middlebury dataset, even though we are not state-of-the-art on the synthetic data, we have better performance on the real scene data than any other previous work, which is more important for the generalization of the frame interpolation method. In the future, we plan to generalize this work for videos with transitions among and to make the model fully flexible to perform the variable-length multi-frame video interpolation.

doi:10.6342/NTU201802624

Bibliography

[1] C. Bailer, B. Taetz, and D. Stricker. Flow fields: Dense correspondence fields for highly accurate large displacement optical flow estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 4015–4023, 2015.

[2] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1):1–31, 2011.

[3] R. W. Brislin. Back-translation for cross-cultural research. *Journal of cross-cultural psychology*, 1(3):185–216, 1970.

[4] J. Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.

[5] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2758–2766, 2015.

[6] D. Gadot and L. Wolf. Patchbatch: a batch augmented loss for optical flow. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4236–4245, 2016.

[7] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, volume 2, page 7, 2017.

doi:10.6342/NTU201802624

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[9] F. Güney and A. Geiger. Deep discrete flow. In *Asian Conference on Computer Vision*, pages 207–224. Springer, 2016.

[10] Q.-X. Huang and L. Guibas. Consistent shape maps via semidefinite programming. In *Proceedings of the Eleventh Eurographics/ACMSIGGRAPH Symposium on Geometry Processing*, pages 177–186. Eurographics Association, 2013.

[11] H. Jiang, D. Sun, V. Jampani, M.-H. Yang, E. Learned-Miller, and J. Kautz. Super slomo: High quality estimation of multiple intermediate frames for video interpolation. *arXiv preprint arXiv:1712.00080*, 2017.

[12] N. Kanopoulos, N. Vasanthavada, and R. L. Baker. Design of an image edge detection filter using the sobel operator. *IEEE Journal of solid-state circuits*, 23(2):358–367, 1988.

[13] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[14] Y. Li and D. P. Huttenlocher. Learning for optical flow using stochastic optimization. In *European Conference on Computer Vision*, pages 379–391. Springer, 2008.

[15] Z. Liu, R. Yeh, X. Tang, Y. Liu, and A. Agarwala. Video frame synthesis using deep voxel flow. In *International Conference on Computer Vision (ICCV)*, volume 2, 2017.

[16] G. Long, L. Kneip, J. M. Alvarez, H. Li, X. Zhang, and Q. Yu. Learning image matching by simply watching video. In *European Conference on Computer Vision*, pages 434–450. Springer, 2016.

[17] D. Marr and E. Hildreth. Theory of edge detection. *Proc. R. Soc. Lond. B*, 207(1167):187–217, 1980.

doi:10.6342/NTU201802624

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

[19] S. Niklaus and F. Liu. Context-aware synthesis for video frame interpolation. *arXiv preprint arXiv:1803.10967*, 2018.

[20] S. Niklaus, L. Mai, and F. Liu. Video frame interpolation via adaptive convolution. In *CVPR*, volume 2, page 6, 2017.

[21] S. Niklaus, L. Mai, and F. Liu. Video frame interpolation via adaptive separable convolution. *arXiv preprint arXiv:1708.01692*, 2017.

[22] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[23] S. Roth and M. J. Black. On the spatial statistics of optical flow. *International Journal of Computer Vision*, 74(1):33–50, 2007.

[24] K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[25] D. Sun, S. Roth, J. P. Lewis, and M. J. Black. Learning optical flow. In D. Forsyth, P. Torr, and A. Zisserman, editors, *Computer Vision – ECCV 2008*, pages 83–97, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[26] N. Sundaram, T. Brox, and K. Keutzer. Dense point trajectories by gpu-accelerated large displacement optical flow. In *European conference on computer vision*, pages 438–451. Springer, 2010.

[27] D. Teney and M. Hebert. Learning to extract motion from videos in convolutional neural networks. In *Asian Conference on Computer Vision*, pages 412–428. Springer, 2016.

doi:10.6342/NTU201802624

[28] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Deep end2end voxel2voxel prediction. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2016 IEEE Conference on*, pages 402–409. IEEE, 2016.

[29] C. Vondrick, H. Pirsiavash, and A. Torralba. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*, pages 613–621, 2016.

[30] N. Wadhwa, M. Rubinstein, F. Durand, and W. T. Freeman. Phase-based video motion processing. *ACM Transactions on Graphics (TOG)*, 32(4):80, 2013.

[31] F. Wang, Q. Huang, and L. J. Guibas. Image co-segmentation via consistent functional maps. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 849–856, 2013.

[32] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. Deepflow: Large displacement optical flow with deep matching. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1385–1392. IEEE, 2013.

[33] M. Werlberger, T. Pock, M. Unger, and H. Bischof. Optical flow guided tv-l1 video interpolation and restoration. In *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 273–286. Springer, 2011.

[34] S. Xie and Z. Tu. Holistically-nested edge detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1395–1403, 2015.

[35] L. Xu, J. Jia, and Y. Matsushita. Motion detail preserving optical flow estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1744–1757, 2012.

[36] T. Xue, J. Wu, K. Bouman, and B. Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2016.

doi:10.6342/NTU201802624

[37] Z. Yu, H. Li, Z. Wang, Z. Hu, and C. W. Chen. Multi-level video frame interpolation: Exploiting the interaction among different levels. *IEEE Transactions on Circuits and Systems for Video Technology*, 23(7):1235–1248, 2013.

[38] C. Zach, M. Klopschitz, and M. Pollefeys. Disambiguating visual relations using loop constraints. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1426–1433. IEEE, 2010.

[39] T. Zhou, Y. Jae Lee, S. X. Yu, and A. A. Efros. Flowweb: Joint image set alignment by weaving consistent, pixel-wise correspondences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1191–1200, 2015.

[40] T. Zhou, P. Krahenbuhl, M. Aubry, Q. Huang, and A. A. Efros. Learning dense correspondence via 3d-guided cycle consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 117–126, 2016.

[41] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*, 2017.

doi:10.6342/NTU201802624