Independent University **Bangladesh (IUB)**

IUB Academic Repository

Computer Science and Engineering | Undergraduate Thesis

2025-01

U flow-net: integration of flow estimation and kernel-based method in a modified u-net architecture for video frame interpolation

Tabib, Fatin Israq

IUB

http://ar.iub.edu.bd/handle/11348/987

Downloaded from IUB Academic Repository



UFLOW-NET: INTEGRATION OF FLOW ESTIMATION AND KERNEL-BASED METHOD IN A MODIFIED U-NET ARCHITECTURE FOR VIDEO FRAME INTERPOLATION

Fatin Israq Tabib (2022280)

Halima Khatun (2022625)

Shuvro Sourav Sarker (2021251)

Dissertation submitted in partial fulfillment for the Degree of Bachelor of Science in Computer Science and Engineering

Department of Computer Science & Engineering

Independent University, Bangladesh

January 2025

Attestation

We understand the nature of plagiarism, and we are aware of the university's strict policy on the matter. We certify that this is an original work by us. However, others' written work or software used in any part of this project is properly cited following internationally accepted academic guidelines.

Signature:

I. Fatin Israq Tabib: _____

II. Halima Khatun: _____

III. Shuvro Sourav Sarker: _____

Evaluation Committee

Supervision Panel

……	……
Supervisor	Co-supervisor

External Examiners

……	……
External Examiner 1	External Examiner 2

Office Use

……	……
Program Coordinator	Head of the Department

……

Director Graduate Studies, Research and Industry Relations

……

Library

Acknowledgment

We feel deeply indebted to our supervisor Dr. Saadia Binte Alam. Without her proper guidance and care, it would not be possible for us to complete this research.

We would like to thank Mr. Siam Tahsin Bhuiyan for his technical guidance. Besides, we are grateful for getting the expert supervision of Dr. Md. Rashedur Rahman and Mr. Mahmudul Haque in conducting this study.

Abstract

With the increasing popularity of high-quality high-frame-rate video content, advanced techniques for video frame interpolation have become a necessity. Existing video frame interpolation methods face inherent limitations, particularly flow-based and kernel-based approaches. Flow-based methods struggle with accurately estimating motion in complex or occluded regions, often resulting in visible artifacts or a loss of detail in interpolated frames. Kernel-based methods, on the other hand, tend to blur fine-grained details, especially in regions with intricate textures or rapid motion. These challenges highlight the need for more advanced solutions that can address these weaknesses. To address these limitations, we propose UFlow-Net, a unified deep learning architecture that integrates flow estimation and kernel-based methods exploiting their complementary characteristics in a modified U-Net structure. Existing video frame interpolation methods have inherent drawbacks. Optical flow is sensitive to challenging movements and kernel-based methods commonly lose details. To tackle the challenges, UFlow-Net with a novel flow-enhanced encoder-decoder structure possesses dedicated feature fusion strategies to optimize the integration of spatial and temporal information. The novelty of this architecture relies on the addition of flow estimation layers during the downsampling process, which allows relevant motion details to permeate into frame synthesis. This design, in conjunction with a progressive refinement approach, guarantees high-fidelity output while remaining computationally efficient. Extensive experimental evaluations with Tarzan were performed using the Vimeo90K, UCF101, and Middlebury datasets. Extensive experiments evaluate UFlow-Net's performance against many previous optical flow models in terms of Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM) of the interpolated results, with improved maintaining of structural detail and spatial coherence in UFlow-Net interpolated frames. Though UFlow-Net outperforms current methods by a large margin, it may still struggle in very complex background scenes or images with highly rapid motion. The following are limitations that guide further studies in video frame interpolation technology. This research has significant practical implications. The accuracy and computational efficiency balance of UFlow-Net renders it especially applicable to real-world video enhancement tasks. This work makes a major step forward in the development of high quality video processing systems and looks positive for future work.

List of Publications

F. Israq, S. B. Alam, H. Khatun, S.S. Sarker, S.T. Bhuiyan, M. Haque, R. Rahman and S. Kobashi " UFlow-Net: A Unified Approach for Improved Video Frame Interpolation" in *Proc. 2024 27th International Conference on Computer and Information Technology (ICCIT)*, Cox's Bazar, Bangladesh, Dec. 20-22, 2024. (Accepted)

Table of Contents

CHAPTER 1: Introduction 1
1.1. Background and Context 3
1.2. Scope and Objective 5
1.3. Contributions 7
1.4. Outline of Thesis 7

CHAPTER 2: Literature Review 9

CHAPTER 3: Background 12
3.1. Convolutional Neural Network 12
3.1.1. Convolution Layer 13
3.1.2. Pooling Layer 14
3.1.3. Non-linear Activation Functions 15
3.1.4 Fully Connected Layer 16
3.2. U-Net 16

CHAPTER 4: Materials 18
4.1. Dataset 18
4.1.1. Vimeo90K 18
4.1.2. UCF101 19
4.1.3. Middlebury Optical Flow 20

CHAPTER 5: Proposed Method 22
5.1. Data Preprocessing 22
5.2. Method 23

5.3. UFlow-Net 23
5.3.1. Flow-Enhanced Encoder-Decoder.....25
5.3.2. Refined Frame Synthesis 26
5.4. Model Settings 27
5.5. Evaluation Metrics 27

CHAPTER 6: Results and Discussion.....29
6.1 Training.....29
6.2 Test.....32
6.3 Comparative Analysis.....35

CHAPTER 7: Conclusion.....37
7.1 Summary 37
7.2 Limitation.....38
7.3 Future Work 39

References 40

List of Figures

Figure 3.1: Convolution operation. 14
Figure 3.2: Max pooling operation 14
Figure 3.3: U-Net architecture. 17
Figure 4.1: Samples of the Vimeo90K Dataset. 19
Figure 4.2: Samples of the UCF101 Dataset. 20
Figure 4.3: Samples of the Middlebury Dataset. 21
Figure 5.1: Overview of data preprocessing. 22
Figure 5.2: Overview of the proposed method 23
Figure 5.3: UFlow-Net Architecture. **(a)** Integrated Encoder with Flow Estimation. **(b)** Decoder with Refinement Process for Frame Interpolation**.** 26
Figure 6.1: Training Loss Curve. 30
Figure 6.2: PSNR Curve. 31
Figure 6.3: Model's Performance on UCF101 Dataset. 33
Figure 6.4: Model's Performance on Vimeo90k Dataset. 33
Figure 6.5: Model's Performance on Middlebury Dataset 34

List of Tables

Table 6.1: Comparative Analysis of Methodological Efficacy. 36

CHAPTER 1: Introduction

Video frame interpolation (VFI) is a key technique in digital video processing and computer vision, with applications spanning across various domains. In fields such as entertainment, sports broadcasting, computer vision and scientific visualization VFI is a significant area of research due to its growing demand for high-quality, high-frame-rate video content [1]. VFI plays a crucial role in enhancing video quality, enabling advanced video analysis, and improving the performance of numerous computer vision tasks.

Computer vision is a multidisciplinary field that focuses on enabling machines to interpret and understand visual information from the world, using digital images and videos. It combines techniques from image processing, machine learning, and computer science to perform various tasks. Significant progress has been observed in the past few years across various areas of computer vision [2], including tasks such as image classification, object detection, semantic segmentation, and action recognition [3]. These responsibilities are crucial for numerous contemporary applications, such as self-driving cars, robotics, medical imaging, and augmented reality [4]. Within this context, VFI serves as a critical preprocessing step, enhancing the quality and temporal resolution of video inputs, which in turn improves the performance of downstream computer vision tasks.

The primary goal of VFI is to synthesize intermediate frames between existing frames in a video sequence, thereby increasing the temporal resolution and improving visual quality. This process is essential for various applications, including frame rate up-conversion, slow-motion video generation, and bandwidth-efficient video streaming [5] [6]. Video Frame Interpolation using deep learning was applied to enhance sports video analysis by generating smooth, high-frame-rate sequences, enabling detailed motion insights in real time [7]. Through real-time intermediate flow estimation, VFI accurately predicts frames between existing ones, allowing for more precise analysis of rapid movements and actions in sports footage [8]. In the field of autonomous systems, such as self-driving cars and drones, VFI can help in processing and analyzing video streams more effectively, contributing to improved real-time decision-making in dynamic environments [9][10]. VFI techniques also find applications in video compression, where they can be leveraged to

develop more efficient algorithms. By intelligently interpolating frames, these methods can reduce bandwidth requirements while maintaining visual quality, a critical consideration in the era of streaming media and limited network resources [11]. In the medical domain, VFI enhances the temporal resolution of diagnostic videos, improving the accuracy of motion-based medical assessments in areas such as endoscopy and echocardiography [12]. However, achieving high-quality frame interpolation poses several challenges. Traditional methods, such as those based on optical flow estimation, often struggle with complex motion patterns, preservation of high-frequency details, and handling of occlusions and disocclusions [13]. These limitations can lead to artifacts, blurriness, and inconsistencies in the interpolated frames, particularly in scenes with rapid movement or intricate backgrounds [14].

To address these challenges, researchers have turned to deep learning techniques, particularly convolutional neural networks (CNNs), to develop more sophisticated VFI methods [15], [16], [17]. These approaches have shown promise in capturing complex spatio-temporal features and producing higher quality interpolated frames. The use of deep learning in VFI aligns with broader trends in computer vision, where neural networks have become the dominant paradigm for tackling complex visual tasks [18]. However, many existing methods still struggle to balance accuracy and computational efficiency, especially when dealing with diverse scene content and large motion magnitudes [19]. In computer vision applications, maintaining this balance is critical, as real-time speed is just as essential as precision. Thus, the aim of this research is to create an innovative deep learning method for video frame interpolation that can surpass these restrictions and generate high quality interpolated frames for various types of videos. To this end, we propose UFlow-Net, a unified approach that integrates flow estimation and kernel-based methods within a modified U-Net framework. This approach aims to leverage the strengths of both flow-based and kernel-based methods while mitigating their individual weaknesses, potentially offering a more robust solution for VFI in diverse computer vision applications.

In this study, we utilize three widely-used datasets for training and evaluation: Vimeo90K [14], UCF101 [20], and Middlebury [21]. These datasets provide a diverse range of video content, allowing us to assess the performance of our proposed method across various challenging scenarios that are representative of real-world computer vision tasks. The model is trained on Vimeo90K dataset, and the trained model is used to predict interpolated frames on video sequences that were

not part of the training set from Vimeo90k dataset along with the UCF101 and Middlebury dataset and then their performance is compared using standard metrics PSNR and SSIM, comparing the quality of interpolated frames across different methods. By analyzing these results, we aim to demonstrate the effectiveness of UFlow-Net in addressing the challenges of video frame interpolation.

1.1. Background and Context

The primary concept of video frame interpolation is synthesis of additional frames between pre-existing frames within a video sequence. This method has achieved considerable interest, particularly as a pre-processing step that enhances the quality of input data for improved performance in video processing tasks. With the increase in demand for high-frame rate video content, frame interpolation has become a critical focus in video processing research. It has been used in various ways, such as frame rate up-conversion [22], video compression [23], and quality enhancement [24]. These developments have rendered frame interpolation a critical method for satisfying the growing demands of contemporary video technology. Globally, the need for frame interpolation techniques is expanding rapidly. According to a study conducted by Wu et al., the application of frame interpolation in video compression resulted in a 20% reduction in bitrate while maintaining visual quality [25]. Similar trends can be observed in studies conducted by Bao et al., Jiang et al., and Niklaus et al., in the fields of frame rate up-conversion, slow-motion generation, and view synthesis, respectively. In the frame rate up-conversion study, researchers found a 30% improvement in motion smoothness using interpolation techniques [26]. Jiang et al. demonstrated that frame interpolation could generate high-quality slow-motion videos from standard frame rate inputs [27]. Niklaus et al. showed that frame interpolation techniques could be extended to novel view synthesis, opening up new possibilities in virtual reality applications [28]. Frame interpolation errors can lead to visual artifacts, motion jerkiness, and decreased overall video quality [29], [30], [31], [32]. In a study conducted by Liu et al., videos with poor frame interpolation were found to cause 25% more viewer discomfort compared to those with high-quality interpolation [5]. The researchers also noted a significant correlation between interpolation quality and viewer engagement. Content creators and streaming platforms are particularly susceptible to the effects of poor frame interpolation, as it directly impacts user experience and retention [15], [33], [34], [35].

When faced with the task of video frame interpolation, researchers have employed various approaches, including optical flow estimation, kernel-based methods, and more recently, deep learning techniques [36]. To determine the most effective method for a given scenario, several factors are considered, such as computational efficiency, output quality, and robustness to different types of motion. Although optical flow-based methods offer accurate motion estimation [37], kernel-based approaches provide flexibility in handling complex motions [38]. Deep learning models have emerged as a powerful tool for combining the strengths of both approaches [39]. Besides, in cases of real-time applications or resource-constrained environments, lighter models that prioritize speed over absolute quality may be preferred. Apart from that, due to the ability to learn from large datasets and capture complex spatio-temporal relationships, deep learning-based methods have shown superior performance in various benchmarks [40].

However, existing frame interpolation methods still face challenges in handling complex scenes with intricate backgrounds and fast-moving objects [41]. Some techniques struggle with preserving fine details during the interpolation process, while others may introduce artifacts in areas of occlusion or disocclusion [42]. Besides, the trade-off between computational efficiency and output quality remains a significant challenge, particularly for real-time applications [43]. In addition, the generalization of models across different types of content and motion patterns can be limited [44].

Flow-based methods, while theoretically capable of handling arbitrary motion magnitudes, face significant challenges in complex real-world scenarios due to fundamental limitations in optical flow estimation. These methods rely heavily on the accuracy of motion vector fields, which becomes problematic in scenarios involving occlusions, transparency, or complex texture patterns. In occluded regions, the absence of corresponding pixels between frames leads to unreliable flow estimates, resulting in visible artifacts in the interpolated frames. The presence of repetitive patterns or textures introduces ambiguity in pixel correspondence matching, often leading to incorrect flow predictions. Moreover, rapid motion introduces additional complications through motion blur and large displacements, which can cause traditional flow estimation algorithms to fail in maintaining temporal coherence. These limitations are particularly evident in scenes with complex depth relationships, where the assumption of continuous motion fields breaks down at object boundaries and depth discontinuities. Additionally, variations in lighting conditions and

intricate non-linear motions, such as fluttering fabric or rippling water, further complicate the accurate estimation of optical flow. These scenarios challenge the underlying assumptions of motion consistency, often leading to significant errors. To overcome these hurdles, advanced techniques must be developed to improve the resilience of flow-based methods, particularly in handling unpredictable and highly dynamic environments.

Kernel-based methods for video frame interpolation exhibit fundamental limitations that stem from their architectural constraints and operational mechanisms. The primary limitation arises from the finite spatial receptive field defined by the kernel size, which inherently restricts the method's ability to capture large motion displacements. When object movements exceed the predefined kernel dimensions, the algorithm fails to establish accurate pixel correspondences, resulting in degraded interpolation quality. Furthermore, the utilization of shared convolutional weights across spatial positions introduces a homogeneity assumption that proves problematic in real-world scenarios. This assumption fails to account for the heterogeneous nature of motion patterns present in different regions of the frame, particularly in scenes containing multiple objects with varying motion characteristics. The computational complexity of kernel-based methods also increases quadratically with kernel size, creating a practical upper bound on the maximum motion magnitude that can be effectively handled.

Due to these limitations, it is crucial to develop more robust and versatile frame interpolation techniques. If a method could effectively combine the strengths of flow-based and kernel-based approaches within a unified framework, it could potentially address many of these challenges. Also, such a method could serve as a foundation for further advancements in video processing tasks such as super-resolution, deblurring, and compression.

1.2. Scope and Objective

Video frame interpolation presents a multifaceted challenge in the realm of digital video processing. This complex process involves synthesizing intermediate frames between existing ones in a video sequence, aiming to enhance motion smoothness and visual quality. However, certain regions in the interpolated frames can be distorted or blurred, causing the output to imperfectly represent the original motion and content of the video. This is particularly problematic

in scenes with complex backgrounds or fast-moving objects, making the viewing experience challenging for end-users. Moreover, the preservation of fine details and the effective handling of occlusions during the interpolation process further complicate the achievement of high-quality results.

These inherent challenges in video frame interpolation underscore the necessity for an innovative solution that can significantly improve the accuracy and visual quality of synthesized frames. If intermediate frames can be generated clearly and automatically with high fidelity to the original video content, the overall quality and smoothness of the video would be substantially enhanced. Deep learning has emerged as a powerful tool for analyzing and synthesizing visual content across a wide variety of purposes, including video processing tasks. By utilizing deep neural networks, particularly those designed for spatio-temporal learning, frame interpolation can be significantly optimized, leading to more realistic and visually cohesive video outputs. This approach offers a promising path forward in overcoming the limitations of traditional methods.

In light of these considerations, the primary objective of this study is to develop and evaluate a novel method for automated video frame interpolation. Our approach uniquely combines flow estimation and kernel-based methods within a U-Net framework, resulting in the proposed UFlow-Net architecture. The research focuses on developing UFlow-Net, a novel deep learning architecture for video frame interpolation that seamlessly integrates flow-based and kernel-based approaches. This study encompasses the design, implementation, and optimization of UFlow-Net using diverse video datasets to ensure broad applicability. The model's performance is rigorously evaluated using industry-standard metrics like PSNR and SSIM across multiple benchmark datasets. Additionally, a comprehensive comparative analysis is conducted to assess UFlow-Net's effectiveness against current existing methods, aiming to demonstrate its superior performance and versatility in various video interpolation scenarios.

Beyond the technical development of UFlow-Net, our study provides a critical analysis of its strengths and limitations. This comprehensive approach not only validates the model's effectiveness but also lays the groundwork for future advancements in video frame interpolation. By addressing current challenges and identifying areas for improvement, we seek to push the boundaries of what's possible in high-quality video manipulation and enhancement.

1.3. Contributions

In this study, we develop a novel deep learning-based method, UFlow-Net, for automated video frame interpolation. Our approach involves a comprehensive analysis using a modified U-Net architecture that integrates flow estimation and kernel-based methods. The primary contributions of this research are as follows:

We proposed a unified framework that is built upon the U-Net architecture. This approach estimates parameters with diverse motion information, enabling the synthesis of intermediate frames. To address the common issue of information loss during up- and down-sampling in U-Net architectures, we integrated a flow estimation module into the UFlow-Net encoder. This innovation preserves crucial high-frequency information, thereby enhancing the accuracy of feature acquisition during the encoding process. Our method mitigates the problem of complex background. We achieved this by integrating previous flow estimations from the down-sampling process into the up-sampling process through skip connections, significantly improving the preservation of important information and enhancing the accuracy of feature extraction. We integrated a feature fusion technique that concatenates features of the same resolution in the encoder and decoder paths. The performance of UFlow-Net is thoroughly evaluated using three benchmark datasets, including UCF101 [20], Vimeo90K [14], and Middlebury [21]. Our extensive experiments demonstrate the efficacy of the proposed method in comparison to state-of-the-art approaches, particularly in terms of structural similarity preservation. The obtained results of this study demonstrate the feasibility and effectiveness of utilizing our unified deep learning approach for high-quality video frame interpolation. The findings were presented and published in a peer-reviewed conference, contributing to the advancement of video processing research. Our work provides a solid foundation for future developments in frame interpolation techniques and related video enhancement tasks.

1.4. Outline of Thesis

The thesis is divided into seven chapters:

Chapter 1 introduces the context and scope of the research. It highlights the challenges in video frame interpolation. The chapter also outlines the objectives of developing UFlow-Net and its contributions to improving interpolation accuracy.

Chapter 2 presents a literature review, summarizing key advancements in video frame interpolation. The chapter compares flow-based, kernel-based, and hybrid methods. It also identifies gaps in existing methods.

Chapter 3 focuses on the background, providing a detailed overview of the foundational concepts and techniques that underpin your research. This includes explanations of convolutional neural networks (CNNs), their layers and operations (such as convolution, pooling, and activation functions), and the U-Net architecture.

Chapter 4 describes the datasets used in this study. Vimeo90K, UCF101, and Middlebury datasets are introduced, with an explanation of their relevance to video frame interpolation.

Chapter 5 describes the preprocessing techniques, details of the proposed method, and the design of UFlow-Net, a modified U-Net architecture integrated with flow estimation and kernel-based methods. The chapter discusses data preprocessing steps such as normalization, batching, and caching to prepare the datasets for training and evaluation and explains how motion information is captured through downsampling and upsampling processes, leveraging skip connections for better feature preservation.

Chapter 6 covers the results and discussion. It presents the model's performance across multiple metrics, including PSNR and SSIM, comparing UFlow-Net with other existing methods. The chapter concludes by analyzing the strengths and weaknesses of the proposed method, particularly its ability to preserve spatial-temporal details in challenging video sequences.

Chapter 7 summarizes the findings, limitations and discusses future work. This chapter reflects on the contributions of UFlow-Net to the field of video frame interpolation and suggests potential improvements.

CHAPTER 2: Literature Review

Conventional methods for video frame interpolation usually estimate dense optical flow maps through optical flow algorithms and warp the original frames based on these flow maps. A phase-based approach was proposed to represent motion in the phase shifts of discrete pixel elements, combining phase information across multiple scales in a pyramid [45]. Building upon this, Meyer et al. introduced PhaseNet, a framework designed to address complex, non-linear motion patterns by directly computing the phase decomposition of interstitial frames [46]. He et al. designed an innovative video frame interpolation technique predicated on spatiotemporal saliency analysis [47]. This method aims to prioritize perceptually significant regions and temporal transitions in the interpolation process. However, these methods often struggle to effectively handle complex scenes due to their limitations in accurately estimating optical flow or representing high-frequency components. Additionally, the reliance on hand-crafted algorithms often constrains their ability to generalize across diverse datasets, making them susceptible to errors in scenarios involving occlusions or rapid motion transitions.

In recent years, deep learning techniques have revolutionized the field. Recent advancements have demonstrated the efficacy of Convolutional Neural Networks (CNNs) in modeling object motion through convolution operations [48]. Flow-based methods have seen significant improvements, largely due to enhanced accuracy in flow estimation. FlowNet marked a milestone, using DNNs to learn optical flow through supervised learning [49]. Following this concept, Dutta et al. developed a space-time convolution network with a 3D CNN encoder-decoder architecture [50]. A separate neural module called IFNet was proposed, which directly estimates intermediate optical flow using a privileged distillation scheme for supervision [6]. Recently, a novel VFI approach was proposed that accounts for local variations in motion intensity and location [51]. Shen et al. introduced a unified optimization strategy for concurrently addressing frame interpolation and deblurring while considering spatial distortions [52]. Nikolaus et al. proposed a fully separable framework with multiple bidirectional flows, a many-to-many splatting scheme, and a flexible spatial selective refinement component [53]. These advancements demonstrate the growing sophistication of deep learning frameworks, but challenges remain in effectively balancing computational efficiency with the ability to handle intricate motion patterns.

In research on video frame interpolation, kernel-based methods approach the interpolation process as a convolution operation. In this approach, a predicted kernel is applied to two adjacent input frames to synthesize the intermediate frame. Niklaus et al. introduced the first kernel-based approach that unifies motion estimation and pixel synthesis into a single, cohesive step [17]. This approach involves a DNN that estimates spatially adaptive convolution kernels. A flow-free method, FLAVR is a novel approach that tackles the challenges of traditional optical flow-based video frame interpolation methods [54]. Instead of relying on bidirectional optical flows, FLAVR used three-dimensional spatio-temporal kernels to directly capture motion characteristics from unlabeled videos in an end-to-end trainable manner. Cheng et al. proposed a deformable separable convolution method to derive the kernels, masks, and offsets for interpolation, relying on a significantly reduced set of relevant pixels [55], [56]. However, these approaches often employ shared weights across all positions, which may not be optimal for video interpolation due to the varying motion patterns present in different regions of a frame. Moreover, the inherent limitations of kernel-based techniques in capturing larger motion displacements necessitate hybrid methods that can leverage the strengths of both flow-based and kernel-based paradigms.

Hybrid flow and kernel-based approaches operate by determining optical flow to estimate the movement between input frames and extracting pixel data from regions surrounding the reference points. These methods combine the strengths of flow-based motion estimation with kernel-based interpolation to achieve improved accuracy and robustness. The optical flow component provides precise motion information, while kernel-based operations extract and blend pixel intensities within local neighborhoods to generate the intermediate frames. This combination helps mitigate errors in motion estimation by leveraging spatial context, enabling the model to handle challenging scenarios such as occlusions or large displacements. Another study proposed an adaptive warping layer, which was used to synthesize pixels, making the process fully differentiable [17]. In their subsequent study, the researchers introduced a depth-aware flow projection layer to synthesize intermediate flows, enabling the warping of input frames, depth maps, and contextual features [5]. Meyer et al. introduced a method for correcting phase shifts by combining phase data across various scales within a pyramid structure [48]. In their subsequent study, the researchers explored the integration of the phase-driven and data-driven methods for VFI [49].

Despite the advancements in deep learning for video frame interpolation, challenges still remain when the models deal with intricate backgrounds and rapidly moving objects within the original video frames. In such cases, flow-based approaches often struggle to accurately estimate motion trajectories, and kernel-based methods are limited to capturing movements beyond the size of the kernel. To address these issues, we propose a hybrid architecture that integrates flow estimation and kernel-based methods within a U-Net framework with advanced feature fusion and progressive refinement. The U-Net architecture was first implemented in studies on medical image segmentation to capture features at multiple levels [57]. In recent times, U-Net architecture has gained widespread popularity in enhancement tasks. Its application has also extended to video frame interpolation since it has the ability to capture features across the layers of the network through upsampling and downsampling. By combining high-level and low-level features via skip connections, U-Net effectively preserves the spatio-temporal characteristics of the input frames. The contribution of this paper is that the method uses the U-Net architecture to effectively segment targets from intricate backgrounds and to mitigate the issue of texture loss and spatio-temporal features while upsampling and downsampling. The integration of previous flow estimations during the downsampling process through skip connection to the upsampling process helps to preserve significant information, thereby enhancing the accuracy of feature extraction. Furthermore, this hybrid approach ensures efficient resource utilization and maintains robustness across diverse datasets, offering a scalable solution for real-world video processing tasks.

CHAPTER 3: Background

This chapter provides an overview of the foundational concepts used in video frame interpolation, focusing on convolutional neural networks (CNN) and the U-Net architecture. It explains the components of CNNs, their role in feature extraction, and the U-Net's encoder-decoder structure for preserving spatial and contextual information. Additionally, it outlines modifications to the U-Net, integrating flow estimation layers for enhanced motion capture and interpolation accuracy.

3.1. Convolutional Neural Network

A convolutional neural network (CNN) is a specialized neural network extensively used in current computer vision applications like segmentation, object detection, and image classification. The convolution operation is what sets apart a regular neural network from a CNN. CNNs utilize convolutional layers to capture detailed patterns and spatial data within images. A typical CNN consists of an input layer, activation functions, convolution layer, pooling layer, fully connected layer and output layer. This layered structure allows CNNs to process and interpret image data with remarkable accuracy.

Input layer: This layer consists of the pixel values of the image which are to be fed into the convolution layers. The input layer in a CNN typically consists of the raw pixel values of an image. Here are some key points about the input layer: Dimensions, Normalization, Color channels, Batch processing, Data augmentation.

- Dimensions: The input layer's dimensions are equal to the dimensions of the image. For instance, a 224×224 RGB image would have an input layer with dimensions of 224×224×3 (height × width × color channels).
- Normalization: Normalization is frequently done by scaling pixel values to a range of 0-1 before inputting them into the network. This contributes to the training's speed and stability.
- Color channels: Color images typically have three channels: Red, Green, and Blue. Images in grayscale would only contain one channel.
- Batch processing: CNNs in real-life scenarios frequently handle several images at once, resulting in the input layer having four dimensions (batch size × height × width × channels).

- ❖ Data augmentation: Data augmentation involves applying different transformations, such as rotations or flips, to input images in order to enhance the variety of the training data.

3.1.1. Convolution Layer

A feature map is created when a square kernel moves across the input image in this layer. The value of the overlapping kernel is multiplied with each input pixel, and the resulting products are added together to generate the corresponding value in the feature map. While moving across the input image, the kernel searches for specific characteristics within it. The existence or non-existence of characteristics is indicated in the correct locations on the resulting feature map. Every weight in the kernel can be learned and every bias in the feature map can be learned. Additionally, the convolution operation allows for the adjustment of two other hyper parameters: stride length and padding. The distance of the stride directly impacts how many pixels the kernel moves across the image. Increasing the length of the stride can decrease the size of the feature map and improve computational efficiency, but may lead to losing some image details. Padding is used to decide if extra pixels will be included at the edges of the image. This is carried out in order to guarantee that the size of the output feature map aligns with the size of the input and to preserve the details at the edges of the image. Additionally, if the input contains multiple channels and the convolution includes multiple kernels, each kernel is individually applied to every input channel and the outcomes are combined to create the respective output channel. The output will have the same number of channels as the kernels. The formula for calculating the number of trainable parameters in a convolutional layer is as follows:

$$No.of\ trainable\ parameters = (H \times W \times C + 1) \times F \qquad (1)$$

Where H is kernel height, W is kernel width, C is no. of input channels and F is no. of kernels. The height and width of the output feature map can be determined through the following formulae:

$$Output\ height = \frac{H_i - H_k + 2 \times P}{S} + 1 \qquad (2)$$

$$Output\ width = \frac{W_i - W_k + 2 \times P}{S} + 1 \qquad (3)$$

Where H_i denotes input height, H_k denotes kernel height, W_i denotes input width, W_k denotes kernel width, P denotes padding and S denotes stride length.

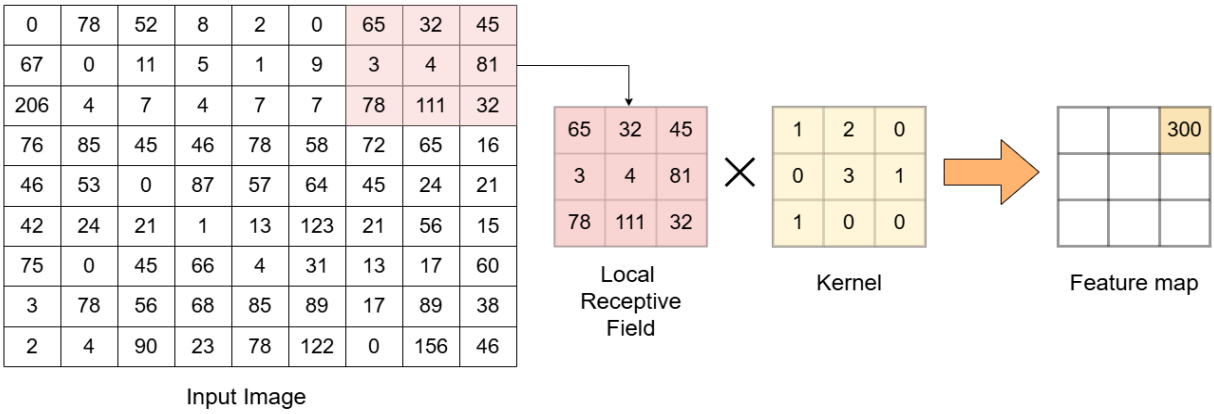


Figure 3.1: Convolution operation.

3.1.2. Pooling Layer

Pooling operations follow convolution operations to decrease the size of the feature maps while retaining key information. There are three kinds of pooling: max pool, average pool, and L2. During max pooling, a square zone of the image is examined, with the highest pixel value being selected as the output neuron's value. The square moves right and repeats until the entire image is covered. In average pooling, the mean of pixel values within a square region is calculated rather than the maximum value, while in L2 pooling, the square root of the sum of the squares of pixel values is calculated. The main purpose of all pooling operations is to simplify the information in feature maps to speed up computations by reducing the output dimensions.

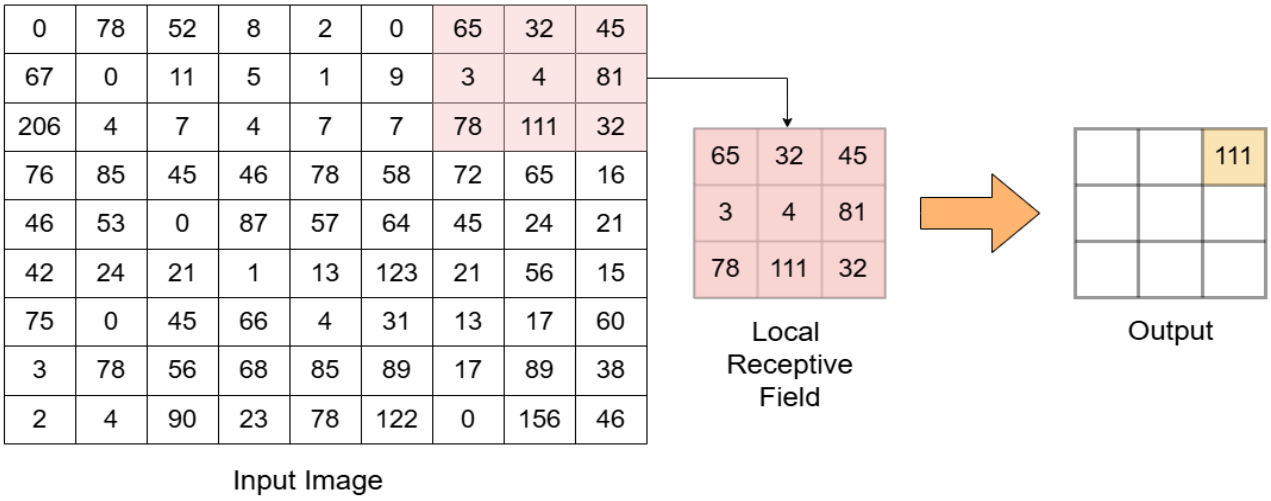


Figure 3.2: Max pooling operation.

3.1.3. Non-linear Activation Functions

Linear mathematical operations in the CNN necessitate the use of non-linear activation functions to introduce non-linearity and allow the CNN to understand intricate details in data. Rectified Linear Unit (ReLU), hyperbolic tangent (tanh), sigmoid and softmax are among the commonly used nonlinear functions in CNNs. These activation functions enable the network to learn complex patterns by introducing flexibility in the mapping of inputs to outputs. Additionally, they help address vanishing gradient problems, improving the convergence of deep networks during training.

1) Rectified Linear Unit (ReLU): ReLU takes an input value from a neuron and returns the same value as output if the value is greater than 0, if it is equal to or less than 0 it returns 0 as output. ReLU also alleviates the vanishing gradient problem in CNNs. The ReLU function is shown below :

$$f(x) = \max(0, x) \qquad (4)$$

2) Hyperbolic tangent (tanh): Tanh converts the input into a value ranging from -1 to 1. The function is as follows :

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \qquad (5)$$

3) Sigmoid: The sigmoid function squashes the input value into a number ranging from 0 to 1. It is useful in binary classification tasks. The sigmoid function is shown below :

$$\sigma(x) = \frac{1}{1 + e^{-x}} \qquad (6)$$

4) Softmax: The softmax function transforms a list of numbers into a list of probabilities that sum to 1. It is useful when you have to select between multiple classes like which class an image falls into.

$$\text{softmax}\,(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}} \qquad (7)$$

3.1.4 Fully Connected Layer

It is a common practice to utilize a fully connected layer at the end of a CNN in image classification tasks. The results from the initial CNN layers are transformed into a one-dimensional array and inputted into a fully connected neural network where every neuron is linked to all others. The activity of each neuron is controlled by an affine function and a subsequent nonlinear activation function. The linear function is written as:

$$f(x) = WX + b \qquad (8)$$

In this context, W represents a tensor containing the weights linked to the output neuron, X represents a tensor containing the neuron activations in the previous layer connected to the output neuron, and b is the bias term.

Output layer: The final output layer contains the desired classification or probability distribution of the classes. The fully connected layer's result is passed through a sigmoid function for binary classification or a softmax function for multiclass classification.

3.2. U-Net

U-Net is a unique convolutional neural network that excels at semantic image segmentation by assigning a class label to every pixel in the image. The structure is capable of producing very accurate segmentation outcomes using only a small set of images. It's the most advanced model currently used for image processing tasks. The structure of U-Net is balanced with an encoder path and a decoder path. The encoder path obtains contextual information from the image and decreases the image's spatial dimensions using a 2x2 max pooling operation. The consecutive 3x3 convolutional layers assist the model in understanding intricate abstract patterns in the data. At the end of the encoder path, there is a large quantity of feature maps that have small dimensions, resulting in the model acquiring features and context but losing location details as it progresses through the encoder path. In the decoder pathway, the size of the image is increased using a 2x2 transposed convolution operation to restore the original dimensions. In reversed convolution, zeros pixels are inserted after each pixel in the input image and extra zeros are padded around the edges of the image; this is then followed by a traditional convolution process producing an output with increased dimensions compared to the input. Feature maps are combined in the decoder path by adding them to the upsampled feature maps from the encoder path to preserve learned features and

localization information. Cropping the feature maps is necessary as the pixels on the edges are lost during convolution.

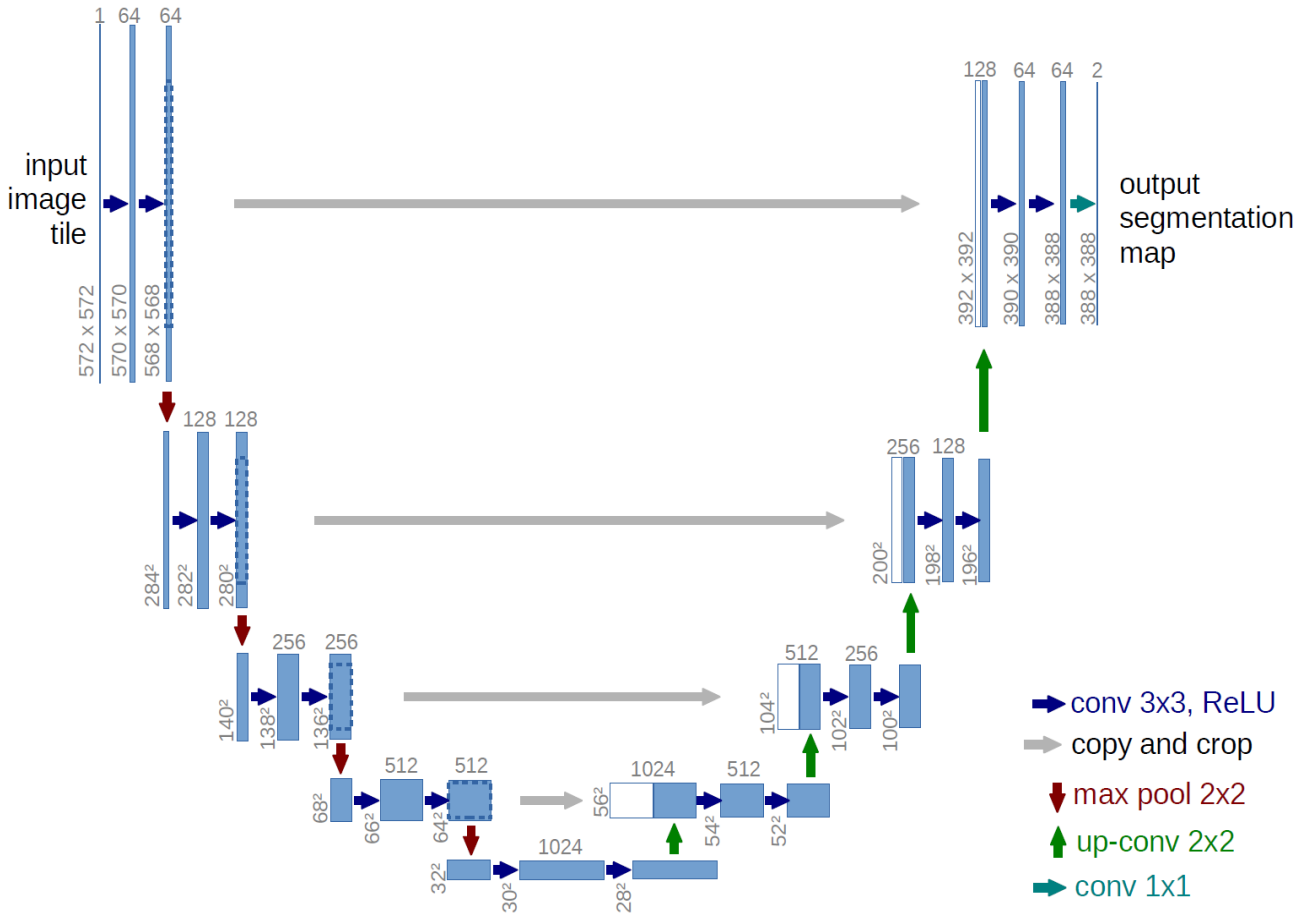


Figure 3.3: U-Net architecture.

Figure 3.2 illustrates the basic U-Net architecture that was proposed by O. Ronneberger [57]. In this study, the U-Net is modified with down-sampling operations coupled with flow estimation layers to capture motion information between the input frames. The motion information is subsequently integrated during the upsampling process and through additional convolutional layers that progressively enhance the features for frame interpolation.

CHAPTER 4: Materials

This chapter provides a detailed description of the datasets employed in this study. The primary dataset used for training and validation is the Vimeo90K dataset, specifically the Triplet Dataset, which is utilized for video frame interpolation tasks. Additionally, the UCF101 dataset is used for evaluation, while the Middlebury Optical Flow Evaluation dataset serves to evaluate optical flow and interpolation performance.

4.1. Dataset

4.1.1. Vimeo90K

The Vimeo90K dataset provides a robust collection of short videos designed to support research in video processing applications, including temporal frame interpolation, video denoising, deblocking, and super-resolution [14]. This dataset comprises 89,800 high-quality short videos sourced from original full-length videos, with options for batch downloading. To prepare these clips, they utilized specific utilities from the AoT Dataset repository for scene detection and camera stabilization, ensuring standardized clip quality across the dataset. The Vimeo90K dataset is organized into two primary subsets: (i). Triplet Dataset specifically designed for temporal frame interpolation, includes 73,171 sequences of 3 frames each. Extracted from 15,000 selected clips within Vimeo90K, each sequence is formatted at a consistent resolution of 448 × 256 pixels. The Triplet Dataset is available for download, with options for a testing-only set (17GB) and a combined training and test set (33GB). (ii). The Septuplet Dataset is tailored for tasks such as video denoising, deblocking, and super-resolution, this subset contains 91,701 sequences of 7 frames each. The septuplet frames are sourced from 39,000 selected clips and maintain a fixed resolution of 448 × 256 pixels. A recent update corrected a bug in the denoising test data generation, with updated quantitative results in the latest version of the associated paper. The Septuplet Dataset includes test sets for video denoising (16GB), deblocking (11GB), and super-resolution (6GB), as well as the original, unprocessed test set (15GB). For comprehensive training and testing, the full original dataset (82GB) is also available.

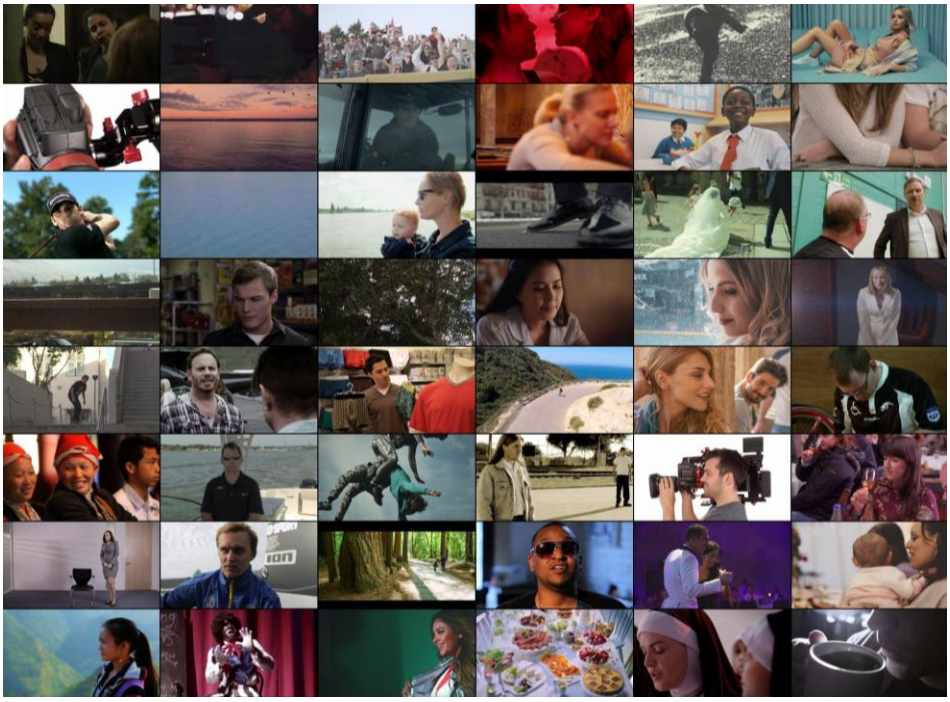


Figure 4.1: Samples of the Vimeo90K Dataset.

4.1.2. UCF101

The UCF101 Action Recognition Dataset, introduced in November 2012, is a comprehensive video dataset widely used as a benchmark for action recognition in computer vision [20]. It comprises 13,320 short videos collected from YouTube, covering 101 distinct human actions, such as "playing guitar," "surfing," and "knitting," with each clip annotated to reflect a single action. This dataset introduces a high level of complexity due to its range of viewpoints, scales, camera motion, and varying lighting conditions, making it suitable for testing the robustness of frame interpolation models. The actions are divided into five categories: Human-Object Interaction, Body-Motion Only, Human-Human Interaction, Playing Musical Instruments, and Sports. In this paper [65] the UCF 101 action dataset was preprocessed to use for testing in video frame interpolation. The dataset contains short videos of human actions which are used for classification purposes. The preprocessing is done by first choosing videos with more movements. These videos are then converted into frames and then three consecutive frames are taken to create triplets for interpolation purposes. The middle frame is used as ground truth and the other two neighboring frames are used as input frames.

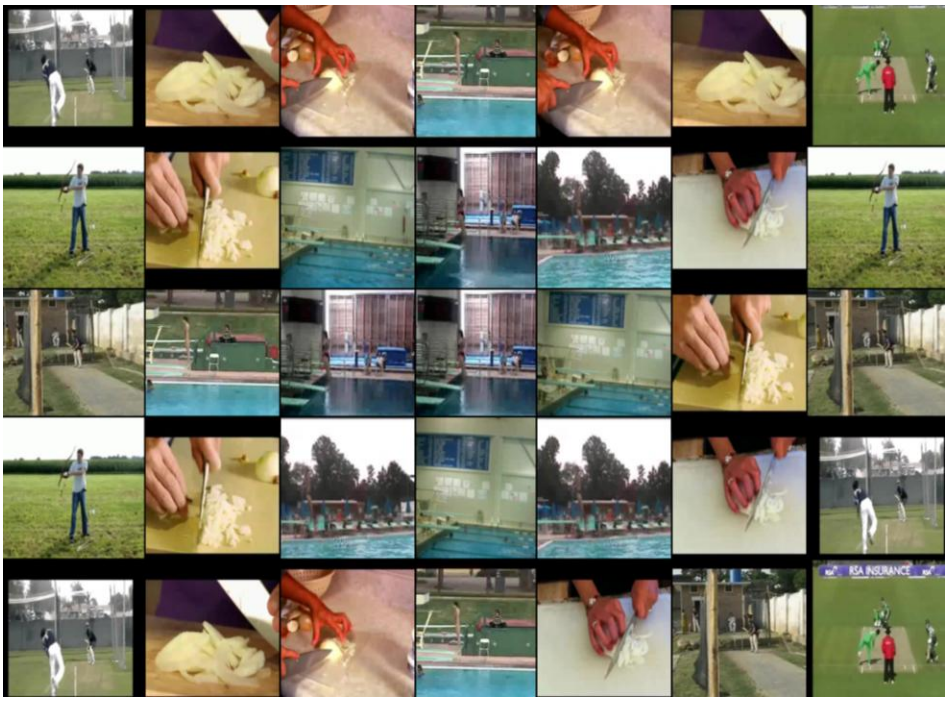


Figure 4.2: Samples of the UCF101 Dataset.

4.1.3. Middlebury Optical Flow

The Middlebury Optical Flow Evaluation Dataset is a comprehensive resource for assessing optical flow and interpolation algorithms [21]. This dataset is categorized into two main sections: “Evaluation Datasets” with Hidden Ground-Truth and “Other Datasets” with Public Ground-Truth, each containing varied scenes with unique characteristics to challenge different optical flow models. The Evaluation Datasets section consists of 12 sequences, including "Army," "Mequon," "Yosemite," and "Backyard," among others. These datasets are designed for rigorous flow and interpolation evaluations, with eight frames per sequence for most scenes and ground-truth flow hidden to encourage unbiased testing. Users can download these sequences in both color and grayscale formats, with options for full or two-frame subsets. In contrast, the Other Datasets section provides publicly available ground-truth flow, facilitating model training. The Middlebury Optical Flow dataset contains 9 consecutive frames from each of 12 short videos. We preprocessed this dataset to triplets using the same method that was used to preprocess the UCF 101 dataset [65] for testing in our video frame interpolation purpose. This comprehensive setup assists in training and testing optical flow algorithms and interpolation on diverse scenes under controlled conditions.



Figure 4.3: Samples of the Middlebury Dataset.

In this study, we utilize the Vimeo90K dataset as our primary source for both training and validation, specifically employing the Triplet Dataset. We further evaluate our model using the Vimeo interpolation test, preprocessed UCF101 and Middlebury Optical Flow Dataset. This combination of datasets provides a comprehensive foundation aligned with the primary objectives of our research, allowing for a detailed and consistent evaluation of UFlow-Net across diverse and challenging scenarios. The Vimeo90K dataset, known for its high-quality frames and motion diversity, provides a solid basis for training the model, while UCF101 adds complexity through its diverse action-based video content. The Middlebury dataset introduces further challenges by including varying motion patterns and occlusions, which are crucial for assessing the model's ability to generalize to different real-world scenarios. This multi-dataset validation approach ensures that the model is evaluated on a wide range of motion complexities, depth variations, and scene dynamics, allowing for a comprehensive understanding of its performance and robustness. Moreover, the inclusion of publicly available ground-truth flow data from Middlebury allows for objective and reliable performance comparison. Through this rigorous testing across these datasets, we aim to showcase UFlow-Net's ability to handle a variety of challenging interpolation tasks.

CHAPTER 5: Proposed Method

This chapter provides a detailed description of the preprocessing methods, proposed method and architecture employed in this study. The chapter further outlines the preprocessing pipeline, including dataset splitting, image processing, and other essential techniques, all of which are crucial for preparing the data and ensuring efficient model training and generalization.

5.1. Data Preprocessing



Figure 5.1: Overview of data preprocessing.

The datasets undergo a comprehensive preprocessing pipeline to prepare them for the video frame interpolation task. The process starts by loading the dataset directory. The folder names are extracted and converted into an array using the NumPy library version 1.21.5 [58], facilitating efficient manipulation. The dataset is then split into training, validation, and testing sets, with the split ratio 90% for training and 10% for validation from vimeo triplet dataset and 100% for testing from vimeo interpolation test, preprocessed UCF101 and Middlebury Optical Flow dataset, determined using a fixed random seed to ensure reproducibility. For each subset, all samples from the subfolders are randomly selected, and the data is stored in separate data structures (dictionaries) for the train, validation, and test sets. These dictionaries are subsequently converted to NumPy arrays for efficient processing.

Next, image preprocessing is carried out to load and process the frames. The first and third frames from each sequence are concatenated to create six-channel inputs for the model. The middle frame in the triplet sequence is used as the target frame for interpolation. To enhance efficiency, the image loading and processing steps are parallelized, allowing for faster image processing.

After processing the images, we use the TensorFlow library version 2.6.0 [59] to convert them into TensorFlow datasets, defining output signatures for the input images (6 channels) and target

images (3 channels) to facilitate efficient data management. To further optimize performance, the datasets are cached and perfected ensuring faster access during training and improving the overall performance of the interpolation model. The dataset is then batched to ensure efficient loading during the training process. Finally, the pixel values of the images are normalized to a range between 0 and 1, which helps in stabilizing and improving the model's training performance by ensuring that all input features are on a consistent scale.

5.2. Method

The proposed methodology for frame interpolation employs a comprehensive deep-learning approach centered on a modified U-Net architecture, named UFlow-Net. The process begins with data preprocessing, where pairs of adjacent frames I_0 and I_2 and their corresponding ground truth intermediate frame, I_t are prepared. The UFlow-Net model is then trained on this data, utilizing a unique architecture that combines down-sampling operations with flow estimation layers to effectively capture motion information between input frames. This motion data is subsequently integrated during the up-sampling process and further refined through additional convolutional layers, enabling the direct synthesis of interpolated frames. The model's performance is optimized using carefully designed loss functions that minimize the discrepancy between the synthesized frame $\hat{I}_t$ and the ground truth I_t. Following training, the model undergoes rigorous evaluation to assess its interpolation capabilities. Finally, the UFlow-Net's performance is benchmarked against other existing frame interpolation methods to validate its effectiveness and contributions to the field.

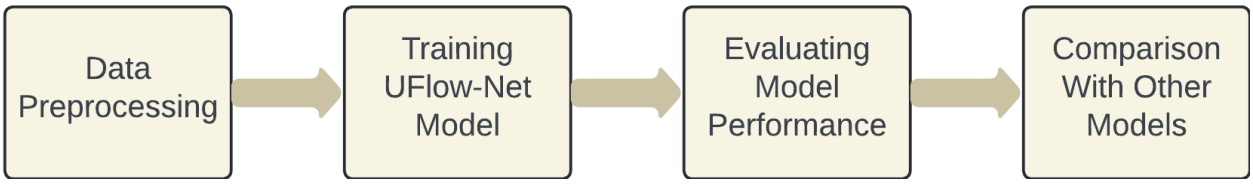


Figure 5.2: Overview of the proposed method.

5.3. UFlow-Net

UFlow-Net represents an advanced deep learning architecture designed specifically for video frame interpolation. Building upon the established U-Net structure, this innovative model incorporates several key enhancements to effectively capture and utilize spatio-temporal

information. At its core, UFlow-Net features a modified U-Net encoder-decoder framework, integrating flow estimation directly into the encoding process. This unique approach allows the network to simultaneously learn motion estimation and frame synthesis tasks. The architecture leverages skip connections to preserve fine-grained details across different scales, while also incorporating flow information from the encoder into the decoder path. UFlow-Net culminates in a refined frame synthesis stage, employing a series of carefully designed operations including masking mechanisms and additional convolutional layers. This comprehensive design enables UFlow-Net to generate high-fidelity interpolated frames that accurately reflect both spatial and temporal variations in the input video data, addressing limitations commonly associated with standard U-Net architectures in preserving crucial spatio-temporal information during the downsampling and upsampling processes. For a comprehensive understanding of the UFlow-Net architecture, we have described it in three key components: the encoder-decoder structure, the flow estimation module, and the frame synthesis stage. Each of these components plays a pivotal role in enhancing the model's ability to handle complex video interpolation tasks. Additionally, the integration of flow-based and kernel-based methods within the model ensures robust motion estimation and effective handling of dynamic video content. The overall design enables UFlow-Net to effectively capture intricate motion details while maintaining high visual fidelity in interpolated frames.

The U-Net structure consists of an encoder for feature downsampling and a decoder for feature upsampling to a predetermined scale [20]. A distinctive aspect of U-Net is its concatenation of same-dimension features between encoder and decoder paths. The encoder extracts low-level features rich in texture and context, while the decoder processes high-level features with reduced spatial dimensions and merges these via skip connections. The proposed network architecture takes two input frames and processes them through a convolutional block, which is a sequence of three convolutional layers. The first two layers have a kernel size of 3x3, followed by a ReLU activation function and the third layer has a 3x3 kernel size without any ReLU activation function and then the output goes through a max pooling layer which is downsampling the convolutional block. However, relying solely on the U-Net can result in the loss of significant spatial-temporal information due to the downsampling and upsampling process. To address this limitation, our approach coupled the downsampling layer with flow estimation to learn both the motion estimation and frame synthesis tasks jointly.

5.3.1. Flow-Enhanced Encoder-Decoder

Numerous studies have demonstrated that flow estimation is beneficial for training kernel-based deep networks in Video Frame Interpolation (VFI) [18]. Building on this insight, we design a U-Net-based encoder with modifications to function as a deep flow encoder, combining the benefits of flow estimation learning with the U-Net architecture to capture detailed spatio-temporal features. This design helps to avoid the loss of crucial contextual information. The deep flow estimation layer is composed of two convolutional layers, each followed by a ReLU activation function. We illustrate this with a simple activation function as follows:

$$\sigma = \max(0, F) \tag{9}$$

The first convolutional layer uses 64 filters, while the second uses 2 filters, both with a kernel size of 3x3. The learnable kernel weights and biases are represented as ω_c and β_c, respectively. The ReLU activation function, denoted by σ, is applied after the convolution operation. The output of the convolutional layer can be described as:

$$F_c = \sigma(\omega_c \times I_{in} + \beta_c) \tag{10}$$

This block is designed to estimate optical flow, producing a flow field with two channels that correspond to the horizontal and vertical components of the flow. The model then concatenates the output from two consecutive downsampled convolutional layers to merge different feature representations, thereby preserving spatial and temporal details. Skip connections are used to transfer information from earlier layers directly to deeper layers, allowing the model to combine high-level features with fine-grained details. The design concludes with a max-pooling operation that reduces dimensions before transitioning to the next stage in the U-Net architecture.

The decoder architecture begins with the bottleneck, which carries the most compressed feature maps. Before entering this bottleneck, the outputs from the downsampling layers for the first and second frames are concatenated, enriching the feature representation with multi-frame information. These are refined through 2x2 upsampling convolutions, which gradually increase the spatial resolution. After each upsampling, the refined features are combined with corresponding high-resolution features from the encoder via skip connections. This combination occurs through concatenation operations, preserving fine-grained details throughout the decoding process. Notably, our architecture integrates flow estimation information from the encoder path

into the decoder through additional skip connections. Each 2x2 upsampling operation systematically reconstructs the input's spatial structure while containing motion information, potentially enhancing the model's ability to process dynamic scenes or video inputs. The upsampling and skip connection concatenated feature undergoes further refinment through the convolutional block. The last convolutional block of the decoder undergoes further processing through two additional 3x3 convolutional layers with ReLU activation. The resulting features are then concatenated with the output from the previous convolutional block, forming a more comprehensive feature representation.

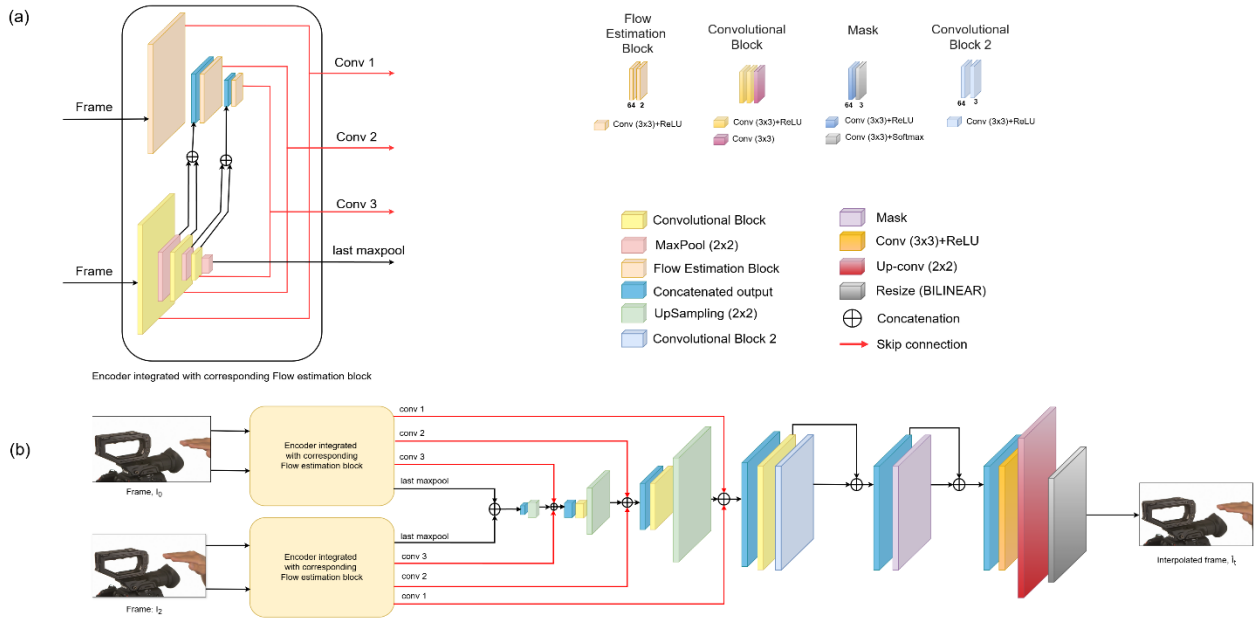


Figure 5.3: UFlow-Net Architecture. (a) Integrated Encoder with Flow Estimation. (b) Decoder with Refinement Process for Frame Interpolation.

5.3.2. Refined Frame Synthesis

We implement a series of operations to refine and upscale the feature maps. The process begins with the concatenated output from the last convolutional block of the decoder and convolutional block 2, which forming a more comprehensive feature representation. A masking mechanism is introduced, which consists of one 3x3 convolutional layer with ReLU activation and one 3x3 convolutional layer with softmax activation function. The softmax mask produced from the convolutional features is represented as:

$$M = \text{Softmax}\ (\omega_m \times F_c + \beta_m) \quad (11)$$

The output of this masking process is then concatenated with the previous feature map, allowing the model to integrate the segmentation information with the refined features. The combined output is further processed through a 3x3 convolutional layer with ReLU activation, which aids in feature refinement and non-linear transformation. Following this, a 2x2 deconvolutional layer is employed to increase the spatial dimensions of the feature map. Finally, a bilinear resize operation is applied to ensure the output dimensions precisely match the input image. This carefully designed sequence of operations enables our model to generate high-fidelity interpolated frames that accurately reflect the spatial and temporal variations of the input data. Figure 5.3 illustrates the overall architecture of UFlow-Net.

5.4. Model Settings

The Adam optimizer, with an initial learning rate set at 1.00E-04 with a decay factor applied for 5 epochs when the validation loss for the model is continuously the same, is employed to guide the model toward optimal convergence. The learning rate gradually decreased from 1.00E-04 to 1.00E-07 and the model stopped by early stopping when 10 continuous epochs had the same validation loss to prevent the overfitting. An SSIM loss function is utilized to improve the preservation of structural details in the interpolated frames. Model effectiveness is evaluated using two key metrics: Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure – (SSIM), both of which provide a quantitative measure of the model's interpolation accuracy. The training process spanned 75 epochs with a batch size of 4, and the model's best weights were saved based on the lowest observed validation loss during training.

5.5. Evaluation Metrics

The performance of the models is evaluated using Structural Similarity Index Measure (SSIM) and Peak Signal-to-Noise Ratio (PSNR) metrics across the training, validation, and test datasets. These metrics provide a comprehensive assessment of the model's interpolation accuracy.

The Structural Similarity Index (SSIM) is a perceptual metric designed to evaluate the similarity between two images by comparing their visual quality. SSIM considers factors such as brightness,

contrast, and structural details, providing a more holistic measure of image fidelity compared to traditional metrics, and is calculated as follows:

$$SSIM(I_t, \widehat{I_t}) = \frac{(2\mu_{I_t}\mu_{\widehat{I_t}} + C_1)(2\sigma_{I_t\widehat{I_t}} + C_2)}{(\mu_{I_t}^2 + \mu_{\widehat{I_t}}^2 + C_1)(\sigma_{I_t}^2 + \sigma_{\widehat{I_t}}^2 + C_2)} \qquad (12)$$

Where:

- μ_{I_t} And $\mu_{\hat{I}_t}$ are the average intensity values of the output I_t and the ground truth $\hat{I}_t$.
- σ_{I_t} And $\sigma_{\hat{I}_t}$ are the variances or measure of contrast of the I_t and $\hat{I}_t$.
- $\sigma_{I_t\hat{I}_t}$ Is the covariance between I_t and $\hat{I}_t$, which measures how structural information is correlated between the two images.
- C_1 and C_2 are constants added to avoid division by zero and stabilize the equation. These constants are usually derived based on the dynamic range of pixel values.

PSNR is a widely utilized metric for evaluating the quality of reconstructed or generated images relative to their reference counterparts. It provides a quantitative measure of the ratio between the maximum possible pixel intensity (MAX) and the Mean Squared Error (MSE) between the reference and reconstructed images, expressed in decibels (dB). A higher PSNR value indicates that the reconstructed image is closer in quality to the reference image. The PSNR is calculated using the following formula:

$$PSNR(I_t, \widehat{I_t}) = 10 \cdot \log_{10}\left(\frac{I_{(2^B-1)}^2}{\frac{1}{M\times N}\sum_{i=1}^{M}\sum_{j=1}^{N}(I_t(i,j) - \widehat{I_t}(i,j))^2}\right) \qquad (13)$$

Where:

- I_t = Original Image.
- $\widehat{I_t}$ = Reconstructed image.
- $(2^B - 1)$ = Maximum possible pixel intensity (MAX) value for a given bit depth.
- M, N = Dimension of the image.

This matrices is particularly important in fields such as video frame interpolation, image compression, and image reconstruction.

CHAPTER 6: Results and Discussion

This chapter presents an in-depth analysis of the results obtained from training, validating, and testing the UFlow-Net model for video frame interpolation. It discusses the model's training performance, focusing on loss convergence, PSNR progression, and the use of SSIM as a loss function to enhance perceptual quality. The testing phase evaluates the model's generalization capability across multiple datasets, including UCF101, Vimeo90k, and Middlebury, highlighting its strengths and limitations in handling diverse scenarios. A comparative analysis with existing methods is also provided, showcasing the proposed method's competitive performance while identifying areas for potential improvement. Additionally, the results are analyzed in terms of the model's ability to preserve structural fidelity and perceptual quality, providing a comprehensive understanding of its practical applicability. Finally, recommendations for future improvements and potential research directions are presented to further enhance the model's performance in more complex or real-world scenarios.

6.1 Training

During the training phase of the network model, we provide the network with the two input frames I_0 (the first frame), I_2 (the third frame), and the ground truth I_t (the actual middle frame) to generate high-quality interpolated frames. To optimize the performance of the model in accurately synthesizing the intermediate frame, we employ the Structural Similarity Index (SSIM) as the primary loss function. SSIM is widely used in image processing tasks due to its ability to capture perceptual quality by comparing luminance, contrast, and structure between two images.

To transform SSIM into a loss function, the similarity index is subtracted from one, effectively converting the similarity measure into a minimization problem. The goal of this approach is to reduce the dissimilarity between the synthesized intermediate frame $\hat{I}t$ (generated by the model) and the ground truth I_t. The SSIM-based loss encourages the model to generate frames with improved perceptual quality, as it minimizes differences in luminance, contrast, and structure, rather than focusing solely on pixel-wise differences.

The objective of the proposed method is to optimize the model by minimizing the variation between the reconstructed middle frame $\hat{I}_t$ and the ground truth frame I_t during training. The SSIM loss function can be mathematically expressed as:

$$\text{Loss}_{SSIM} = 1 - \text{SSIM}(I_t, \hat{I}_t) \quad (14)$$

Where SSIM $I_t, \hat{I}_t$ represents the SSIM index between the generated middle frame $\hat{I}_t$ and the ground truth frame I_t.

By minimizing this SSIM loss, the model learns to generate intermediate frames that closely resemble the actual ground truth frames in terms of perceptual quality, leading to more visually accurate interpolations. This optimization process ensures that the synthesized frames not only achieve high pixel accuracy but also maintain structural and perceptual fidelity, contributing to the overall performance of the video frame interpolation task.

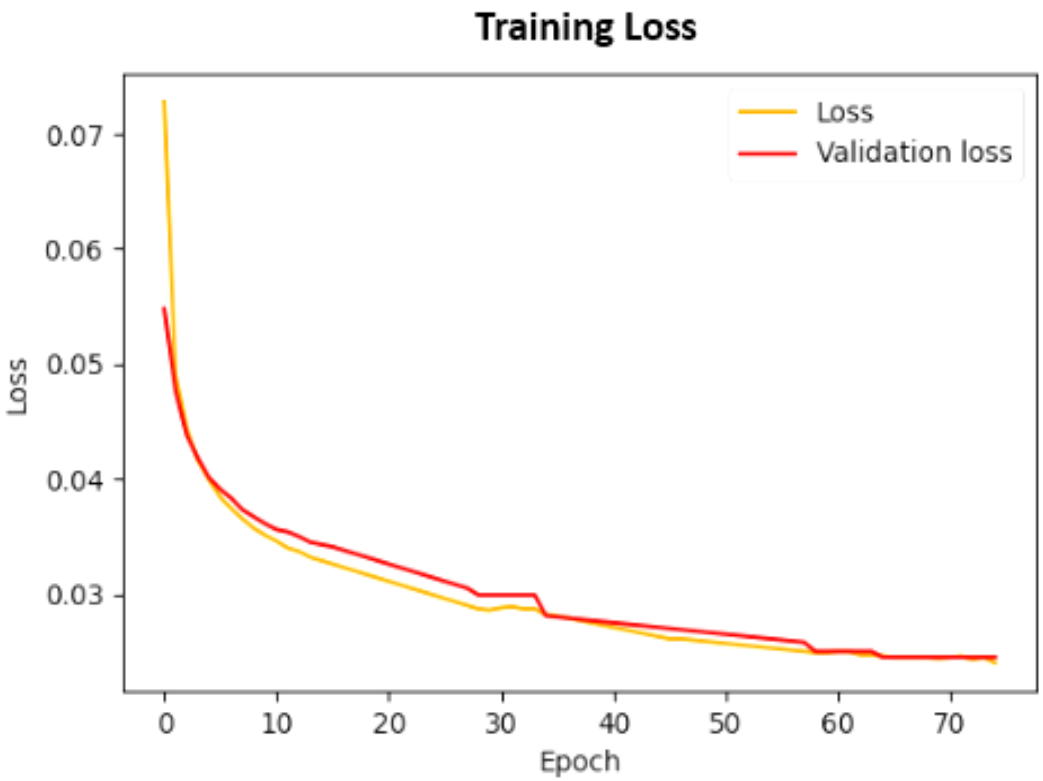


Figure 6.1: Training Loss Curve.

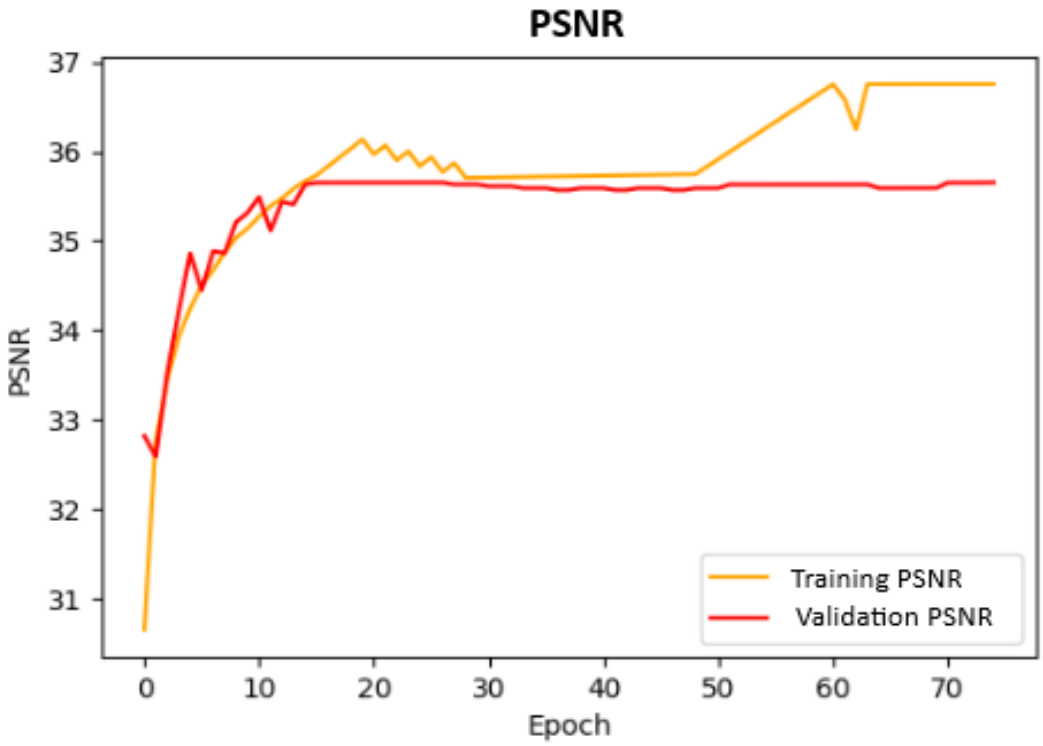


Figure 6.2: PSNR Curve.

The model is trained on a set of 65,854 triplets from the dataset. To assess the model's performance and generalization ability, it is validated using 7317 triplets from the same dataset. Figures 6.1 and 6.2 show the training and validation loss convergence and PSNR throughout the training period respectively for the model across the 75 epochs of training.

The training progression of the frame interpolation model is distinctly captured in the Training Loss graph, illustrating the model's learning dynamics over 75 epochs. The plot demonstrates an initial sharp descent in both training and validation losses from a starting point of approximately 0.07. The most significant improvement occurs within the first 10 epochs, where both curves exhibit rapid convergence. Following this initial phase, the losses continue to decrease more gradually, eventually stabilizing around 0.024-0.027. The close proximity between training and validation loss curves throughout the training period is particularly noteworthy, as it indicates robust generalization capabilities without significant overfitting. This convergence pattern suggests that the model has effectively learned to minimize the reconstruction error while maintaining good generalization to video snippets that were not part of the training set.

The PSNR graph provides crucial insights into the model's image quality preservation capabilities. The plot reveals a substantial improvement in PSNR values during the initial training phase, starting from approximately 30-31 dB. The training curve (shown in yellow) demonstrates consistent improvement, ultimately reaching approximately 36.80 dB, while the validation curve (in red) stabilizes around 35.6 dB. This slight divergence between training and validation PSNR values indicates a minor degree of overfitting, though not severe enough to compromise the model's performance significantly. The overall high PSNR values achieved suggest that the model successfully maintains excellent image quality in its frame interpolation outputs, with the training progression plateauing after about 50 epochs, indicating optimal convergence.

These comprehensive training results, encompassing loss convergence and PSNR improvements, collectively demonstrate the robust performance and effectiveness of our frame interpolation model. The balanced improvement across all metrics suggests that the model has successfully learned to generate high-quality intermediate frames while maintaining both pixel-level accuracy and structural fidelity to the ground truth. Furthermore, the close alignment between the training and validation curves highlights the model’s strong generalization capabilities, mitigating overfitting. The high PSNR and consistent performance across epochs further affirm the model’s ability to preserve fine details and deliver superior results across different video frames and datasets.

6.2 Test

During training, the weights and the model are saved when the lowest validation loss is achieved. This check point ensures that the best-performing version of the model is preserved for further evaluation. The training process focuses on minimizing overfitting while optimizing for generalization to unseen scenarios. Comprehensive testing on separate datasets also helps identify potential weaknesses in the model's predictions. Additionally, such evaluations provide insights into how the model handles varying motion patterns and lighting conditions. The optimum model was then evaluated on video sequences that were not part of the training set, or we can say on the test datasets. This evaluation helps assess the model’s generalization ability and ensures that the frame interpolation performance is not confined to the training data. By testing on diverse video sequences, we can verify the model’s robustness in generating accurate interpolated frames.

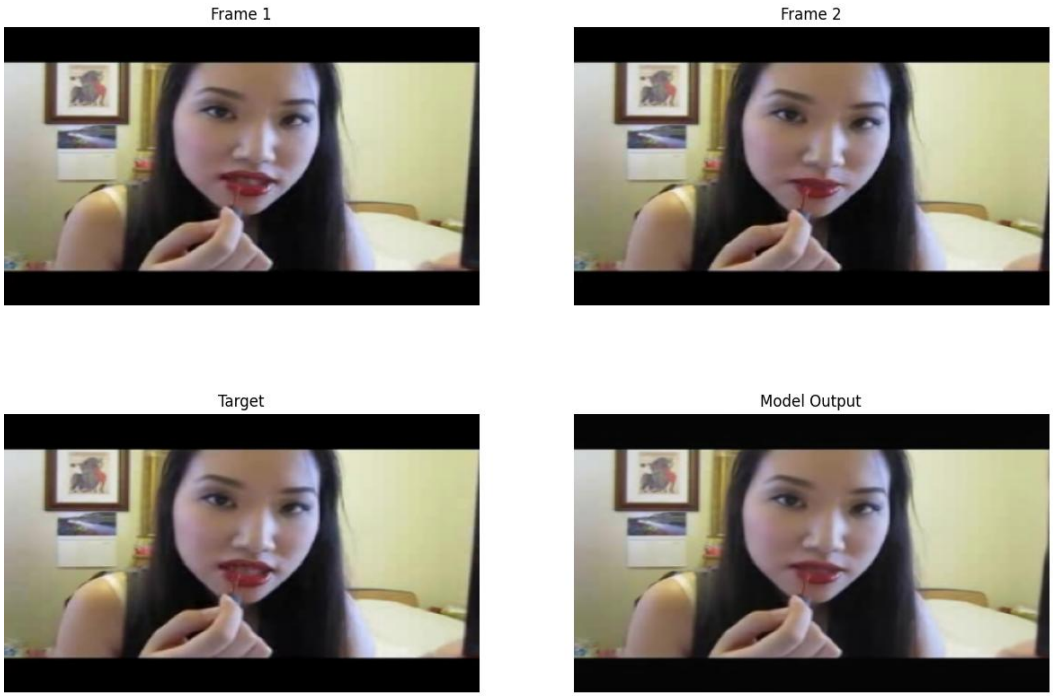


Figure 6.3: Model's Performance on UCF101 Dataset.



Figure 6.4: Model's Performance on Vimeo90k Dataset.

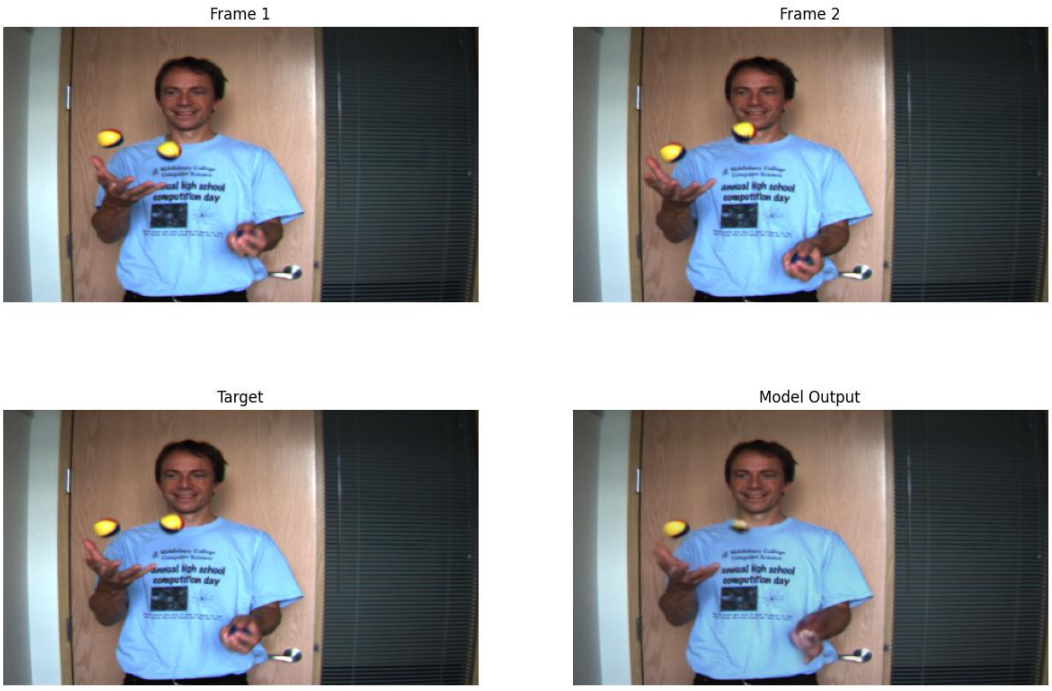


Figure 6.5: Model's Performance on Middlebury Dataset.

The performance of the model on the test dataset resembles their performance in training. The presented Figures 6.3, 6.4, and 6.5 illustrate the efficacy of a video frame interpolation model across three distinct datasets, each representing different challenges in the field of video processing.

In Figure 6.3 derived from the test on UCF101 dataset, the model demonstrates its capability to handle human subjects and subtle facial movements. The interpolation between Frame 1 and Frame 2 results in a Target frame that the Model Output closely approximates. This accuracy in capturing nuanced changes in facial expression and hand positioning suggests the model's proficiency in processing footage featuring human subjects engaged in everyday activities.

Figure 6.4 compares video frames from a cinematic scene, focusing on a woman in a dimly lit setting. Frame 1 and Frame 2 show her with an animated expression, suggesting she's speaking or reacting. She has long, dark hair and wears a light-colored, textured top. The Target frame captures a slightly different moment in her expression. The Model Output demonstrates an attempt to interpolate between the input frames, closely matching the Target. The model shows proficiency

in maintaining consistent lighting, preserving facial details, and accurately representing subtle changes in expression and positioning. It successfully captures the nuanced shift in the woman's facial features and head position, indicating its capability to handle complex human movements in low-light conditions.

The Figure 6.5 is representative of the testing on Middlebury dataset, and introduces additional complexities associated with rapid motion and potential blur. While the model generates a plausible interpolation, subtle discrepancies are observable between the Target and Model Output. The interpolated motion of the juggled objects exhibits slightly less definition, and there is a marginal reduction in the sharpness of facial features and the ball. These observations align with the reported lower SSIM and PSNR metrics for the Middlebury dataset, suggesting that scenarios involving fast movement or motion blur present greater challenges for the current model. This particular case highlights areas for potential refinement in the model's ability to handle high-speed actions and blurred content, providing direction for future improvements in video frame interpolation techniques.

6.3 Comparative Analysis

This section provides a comparison between our proposed architecture, UFlow-Net, and several existing frame interpolation architectures. In our evaluation framework, we included the following methods for comparison: EDSC [56], CtxSyn [60], ToFlow [14], SepConv [17], AdaCoF [61], UNET+RES [62], CDFI [63], and IFRNet [64]. CtxSyn, ToFlow, and IFRNet are flow-based approaches, while SepConv, AdaCoF, and EDSC fall under kernel-based methods, and CDFI and UNET+RES are hybrid-based methods. We employed two widely recognized metrics for objective quality assessment: Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM). These metrics are crucial in evaluating both the perceptual quality and structural accuracy of interpolated frames.

Table 6.1 presents the objective quality evaluation of all methods on three test datasets, with the best score for each metric highlighted in bold. The datasets used for evaluation include UCF101, Vimeo90K, and Middlebury, covering a wide range of scenarios such as human motion, intricate textures, and varying lighting conditions.

Methods	UCF101		Vimeo 90k		Middlebury	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
CtxSyn [60]	31.96	0.92	30.71	0.93	35.58	0.96
ToFLow [14]	28.49	0.92	33.19	0.95	31.06	0.93
IFRNet [64]	32.40	0.93	**35.65**	**0.97**	30.43	0.82
SepConv [17]	29.59	0.90	25.64	0.77	28.15	0.82
AdaCof [61]	32.44	0.92	34.34	0.96	35.56	0.96
EDSC [56]	31.94	0.92	34.49	0.96	36.45	0.96
UNET+RES [62]	32.49	0.93	34.67	0.96	36.23	0.96
CDFI [63]	35.21	**0.95**	35.17	0.96	**37.14**	**0.97**
Ours	**35.23**	**0.95**	**35.65**	**0.97**	33.31	0.95

Table 6.1: Comparative Analysis of Methodological Efficacy.

On the UCF101 dataset, our method achieved a PSNR of 35.23, which ranked best among the compared methods, indicating a strong capability in preserving spatial details. More impressively, our SSIM score of 0.95 was the highest among all methods, showcasing superior performance in maintaining structural integrity and visual quality. For the Vimeo90k dataset, our approach achieved a PSNR of 35.65, placing the highest rank like the flow-based method IFRNet. However, our method truly excelled in terms of SSIM, achieving the highest score of 0.97. This highlighted our method's ability to handle diverse video content, surpassing methods like IFRNet and AdaCoF. On the Middlebury dataset, our method yielded a PSNR of 33.31 and an SSIM score of 0.95, which was respectable given the complexity of this dataset. The Middlebury dataset's fast-moving scenes and rapid frame-to-frame changes presented significant challenges for our frame interpolation technique. These substantial differences between consecutive frames pushed the limits of our motion estimation and frame synthesis algorithms. The task of accurately bridging larger visual gaps and handling complex motion patterns in this dataset proved more demanding than in datasets with more moderate temporal changes, highlighting an area for improvement in our method. Despite this, we outperformed methods like CDFI, which achieved the highest PSNR, and SSIM. These results suggest that while our approach is strong overall, there is room for improvement in handling the specific challenges posed by the Middlebury dataset.

CHAPTER 7: Conclusion

This chapter provides a comprehensive summary of the research, highlighting the significance of video frame interpolation and the challenges addressed by the proposed UFlow-Net architecture. It discusses the model's innovative integration of flow estimation within a modified U-Net framework, emphasizing its effectiveness across diverse datasets while identifying limitations, such as handling rapid motion and computational efficiency. The chapter also outlines opportunities for future research, including advancements in motion estimation, architectural innovation, dataset expansion, and practical applications. In conclusion, UFlow-Net marks a significant advancement in video frame interpolation, with room for further refinement to tackle complex scenarios and enhance its versatility in real-world applications.

7.1 Summary

Video frame interpolation has emerged as a critical technology in the digital era, driven by the increasing demand for high-quality, high-frame-rate video content across various domains. As the entertainment industry, sports broadcasting, and computer vision applications continue to evolve, the need for efficient and accurate frame interpolation techniques has become more pressing than ever. Historically, conventional methods such as optical flow and kernel-based approaches have been employed to address this challenge. However, these techniques often struggle with complex motion patterns and the preservation of fine details, particularly in scenes with intricate backgrounds and rapidly moving objects.To address these limitations, researchers have turned to deep learning techniques, leveraging the power of convolutional neural networks (CNNs) to develop more sophisticated frame interpolation methods. Despite these advancements, many existing approaches still face difficulties in balancing accuracy and computational efficiency, especially when dealing with diverse scene content and large motion magnitudes. This study aimed to overcome these challenges by introducing UFlow-Net, a novel deep learning architecture for video frame interpolation that seamlessly integrates flow estimation and kernel-based methods within a modified U-Net framework. For this research, we utilize the Vimeo90K dataset, a large-scale video dataset specifically designed for video processing tasks. The dataset consists of 89,800 short videos, encompassing a wide range of scenes and actions. We employ the triplet subset,

which includes 73,171 sequences, each comprising three consecutive frames with a fixed resolution of 448 × 256 pixels. These sequences are extracted from 15,000 short videos, providing a diverse set of visual contexts. The dataset is divided into training and testing sets, ensuring a robust evaluation framework. Additionally, we incorporated two other benchmark datasets: UCF101 and Middlebury, to provide a comprehensive evaluation across varying video conditions. The proposed UFlow-Net architecture demonstrates significant advancements in generating high-quality interpolated frames across diverse video content. By incorporating a flow-enhanced encoder-decoder structure with advanced feature fusion and progressive refinement strategies, UFlow-Net effectively addresses the common issue of information loss during up- and down-sampling in U-Net architectures. The integration of flow estimation directly into the encoding process and the utilization of skip connections to preserve fine-grained details across different scales enable UFlow-Net to capture and utilize spatio-temporal information effectively. Experimental results show promising convergence patterns across loss, PSNR, and SSIM metrics, indicating robust generalization capabilities without significant overfitting. The comparative analysis against existing methods across three benchmark datasets demonstrates UFlow-Net's competitive performance, particularly in preserving spatial details, maintaining structural integrity and solving complex background problems. While the model performed exceptionally well on UCF101 and Vimeo90K datasets, the results from the Middlebury dataset highlighted areas for potential improvement, particularly in handling rapid frame-to-frame changes and complex motion patterns.

7.2 Limitation

Despite the promising advances, the research encountered several significant limitations that merit critical examination. The most prominent challenge emerged from the model's variable performance across different datasets. While UFlow-Net demonstrated exceptional results on Vimeo90K and UCF101, the Middlebury dataset revealed inherent difficulties in handling extremely complex motion scenarios and rapid frame-to-frame transitions. Computational efficiency presented another critical limitation. Although the proposed architecture sought to balance accuracy and processing requirements, the model still encountered challenges in achieving optimal performance across all video content types. The inherent complexity of capturing intricate spatio-temporal information necessitates continued refinement of the neural network architecture

and feature extraction mechanisms. The research also highlighted the nuanced challenges of generalizability in deep learning-based frame interpolation. No single model can perfectly address the infinite variety of visual scenarios, and UFlow-Net is no exception. The variations in motion complexity, scene dynamics, and visual characteristics across different datasets underscore the need for more adaptive and flexible interpolation approaches.

7.3 Future Work

The limitations identified in this study highlight promising avenues for advancing video frame interpolation techniques. A primary focus for future research lies in developing sophisticated motion estimation algorithms capable of accurately capturing complex motion patterns, potentially leveraging advanced attention mechanisms and transformer-based architectures. Architectural innovations are equally critical, with opportunities to explore dynamic scaling strategies that adapt to varying resolutions and frame rates, as well as hybrid approaches combining deep learning and traditional signal processing methods to address current challenges. Expanding the scope and diversity of video datasets is another vital direction, as comprehensive datasets enable more robust validation and performance assessment. This effort should be complemented by the creation of nuanced evaluation metrics that capture the subtleties of frame interpolation beyond traditional measures. Practical application development presents an exciting frontier, with collaboration between researchers and industry stakeholders offering insights into real-world requirements across domains like multimedia production, sports broadcasting, medical imaging, and advanced computer vision applications. Additionally, incorporating models for tasks such as denoising and super-resolution could broaden the applicability of interpolation techniques. These advancements, coupled with continuous refinement of existing architectures, aim to create adaptable, efficient, and accurate frame interpolation models, ultimately setting new benchmarks in video quality enhancement.

References

[1] Liu, Y., Y Liao, Yen-Yu Lin and Yung-Yu Chuang. “Deep Video Frame Interpolation Using Cyclic Frame Generation”, in AAAI Conference on Artificial Intelligence, 2019. vol. 33, pp. 8183-8190.

[2] Szeliski, Richard. “Computer Vision - Algorithms and Applications, Second Edition”, texts in Computer Science, 2022.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, 2015, vol. 521, no. 7553, pp. 436-444, 2015.

[4] Liu, Li, Wanli Ouyang, Xiaogang Wang, Paul W. Fieguth, Jie Chen, Xinwang Liu and Matti Pietikäinen. “Deep Learning for Generic Object Detection: A Survey”, in International Journal of Computer Vision, 2018 vol. 128, no. 2, pp. 261-318.

[5] S. Niklaus and F. Liu, "Context-aware synthesis for video frame interpolation," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 1701-1710.

[6] W. Bao, W.S. Lai, C. Ma, ”Depth-aware video frame interpolation,” in Proceedings of IEEE/CVF Conference on Computer Vision Pattern Recognition, Long Beach, USA, 2019, pp. 3703-3712.

[7] T. Soomro, "Deep learning for sports video analysis: A review," in Artificial Intelligence Review, 2022, vol. 55, no. 4, pp. 3109-3170.

[8] Z. Huang, T. Zhang, W. Heng, B. Shi, and S. Zhou, ”RIFE: Real-time intermediate flow estimation for video frame interpolation,” in European Conference on Computer Vision, Cham: Springer Nature, Switzerland, 2020, pp. 624-642.

[9] Wang, J., Sun, K., Cheng, T., Jiang, B., Deng, C., Zhao, Y., Liu, D., Mu, Y., Tan, M., Wang, X., Liu, W., & Xiao, B. "Deep high-resolution representation learning for visual recognition," in IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021, vol. 43, no. 10, pp. 3349-3364.

[10] Grigorescu, Sorin Mihai, Bogdan Trasnea, Tiberiu T. Cocias and Gigel Macesanu. "A survey of deep learning techniques for autonomous driving," in Journal of Field Robotics, 2020, vol. 37, no. 3, pp. 362-386.

[11] Zhang, Xinjie, Jiawei Shao and Jun Zhang. “Low-complexity Deep Video Compression with A Distributed Coding Architecture,” in IEEE International Conference on Multimedia and Expo (ICME), 2023, pp. 2537-2542.

[12] Habermann, Marc, Weipeng Xu, Michael Zollhoefer, Gerard Pons-Moll and Christian Theobalt. “DeepCap: Monocular Human Performance Capture Using Weak Supervision.” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020 pp. 5051-5062.

[13] Y.-L. Liu, Y.-T. Liao, Y.-Y. Lin, and Y.-Y. Chuang, “Deep Video Frame Interpolation Using Cyclic Frame Generation,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 8794–8802.

[14]T. Xue, B. Chen, J. Wu, ”Video enhancement with task-oriented flow,” in International Journal of Computer Vision, 2019, vol. 127, no. 8, pp. 1106-1125.

[15] Jiang, Huaizu, Deqing Sun, V. Jampani, Ming-Hsuan Yang, Erik G. Learned-Miller and Jan Kautz. “Super SloMo: High Quality Estimation of Multiple Intermediate Frames for Video Interpolation,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 9000-9008.

[16] W. Bao, W.S. Lai, X. Zhang, ”Memc-net: Motion estimation and motion compensation driven neural network for video interpolation and enhancement,” IEEE Transactions on Pattern Analysis and Machine Intelligence, 2019, vol. 43, no. 3, pp. 933-948.

[17] S. Niklaus, L. Mai, and F. Liu, ”Video frame interpolation via adaptive separable convolution,” in Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 261-270.

[18] Voulodimos, Athanasios, Nikolaos D. Doulamis, Anastasios D. Doulamis and Eftychios E. Protopapadakis. "Deep learning for computer vision: A brief review," in Computational Intelligence and Neuroscience, vol. 2018, 2018.

[19] Jiang, Junjun, Chengcheng Hao, Zhenyu Li, Xianming Liu and Zhongyuan Wang. “Deep learning based video-related super-resolution technique: a survey.” in Journal of Image and Graphics, 2023.

[20] K. Soomro, A. Roshan Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. In CRCV-TR-12-01, 2012.

[21] S. Baker, D. Scharstein, J.P. Lewis, ”A database and evaluation methodology for optical flow,” in International Journal of Computer Vision, 2011, vol. 92, no. 1, pp. 1-31.

[22] W. Bao, X. Zhang, L. Chen, L. Ding and Z. Gao, ”High-Order Model and Dynamic Filtering for Frame Rate Up-Conversion,” in IEEE Transactions on Image Processing, Aug. 2018, vol. 27, no. 8, pp. 3813-3826.

[23] C.Y. Wu, N. Singhal, and P. Krahenbuhl, ”Video compression through image interpolation,” in Proceedings of European Conference on Computer Vision, Munich, Germany, 2018, pp. 416–431.

[24] W. Shen, W. Bao, G. Zhai, C. Li, M. Xiongkuo, and Gao, ”Blurry video frame interpolation,” In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 5114-5123.

[25] W. Wu, Y. Liu, and J. Li, "Video Frame Interpolation via Adaptive Separable Convolution," in Proceedings of the International Conference on Computer Vision (ICCV), 2017.

[26] Y. Bao, X. Wang, and J. Liu, "Frame Interpolation with Multi-Scale Deep Loss Functions and Generative Adversarial Networks," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[27] H. Jiang, Y. Wang, and Z. Zhang, "High-Quality Slow-Motion Video Generation via Frame Interpolation," IEEE Transactions on Image Processing, vol. 27, no. 8, pp. 3980-3991, 2018.

[28] J. Niklaus, H. Yang, and F. Liu, "Video Frame Synthesis Using Deep Voxel Flow," in Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017.

[29] S. Liu , "Understanding Viewer Discomfort in Frame Interpolation," in IEEE Transactions on Multimedia,, 2017, vol. 19, no. 4, pp. 823-835.

[30] A. Ranjan and M. A. Black, "Optical Flow Estimation Using a Spatial Pyramid Network," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[31] A. S. Parihar, D. Varshney, K. Pandya, and A. Aggarwal, “A comprehensive survey on video frame interpolation techniques,” in The Visual Computer, 2021.

[32]J. Dong, K. Ota, and M. Dong, “Video Frame Interpolation: A Comprehensive Survey,” in ACM Transactions on Multimedia Computing, Communications, and Applications, 2023.

[33] F. Reda, Janne Kontkanen, E. Tabellion, D. Sun, C. Pantofaru, and B. Curless, “FILM: Frame Interpolation for Large Motion,” Lecture notes in computer science, 2022, pp. 250–266.

[34] Niklaus, Simon, Long Mai and Feng Liu. “Video Frame Interpolation via Adaptive Convolution.” 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 2270-2279.

[35] [1]S. S. Sohn et al., “Harnessing Fourier Isovists and Geodesic Interaction for Long-Term Crowd Flow Prediction,” in Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, 2022, pp. 1328–1334.

[36] J.-Y. Chen, "Multi-Frame Motion Deblurring with Optical Flow Estimation," in IEEE Transactions on Image Processing, 2020, vol. 29, pp. 1234-1245..

[37] Y.-S. Chen, "Learning Optical Flow with Deep Learning Techniques," in IEEE Transactions on Multimedia, 2019, vol. 21, no. 6, pp. 1432-1445.

[38] H.-Y. Yang, "Deep Learning-Based Video Frame Interpolation: A Survey," Journal of Visual Communication and Image Representation, 2019, vol. 61, pp. 1-11.

[39] X.-Wang., "A Unified Approach for Video Frame Interpolation Based on Deep Learning," in Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2019.

[40] M. Park, Hak Gu Kim, S. Lee, and Yong Man Ro, “Robust Video Frame Interpolation With Exceptional Motion Map,” IEEE Transactions on Circuits and Systems for Video Technology,2020, vol. 31, no. 2, pp. 754–764.

[41] P.Y.Wang, "Real-Time High-Quality Video Frame Interpolation via Deep Learning Techniques," in IEEE Access, 2020, vol. 8, pp. 123456-123478.

[42] L.H.Liu, "A Comprehensive Review of Motion Estimation Techniques for Video Frame Interpolation," Journal of Visual Communication and Image Representation, 2020, vol .65 ,pp .102 -120.

[43] Tran, Quang Nhat and Shih-Hsuan Yang. “Efficient Video Frame Interpolation Using Generative Adversarial Networks.” Applied Sciences, 2020, vol. 10, pp. 6245.

[44] .H .Zhao, “Deep Learning-Based Methods for Video Frame Interpolation: A Survey,”Journal of Visual Communication and Image Representation ,vol .65 ,pp .102 -120 ,2020 .

[45] S. Meyer, O. Wang, H. Zimmer, M. Grosse, and A. Sorkine-Hornung, ”Phase-based frame interpolation for video,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1410-1418.

[46] S. Meyer, A. Djelouah, B. McWilliams, A. Sorkine-Hornung, M. Gross, and C. Schroers, ”PhaseNet for video frame interpolation,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 468-507.

[47] J. He, G. Yang, X. Liu, and X. Ding, ”Spatio-temporal saliency-based motion vector refinement for frame rate up-conversion,” ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 2020, vol. 16, no. 2, pp. 1-18.

[48] J. Dong, K. Ota, and M. Dong, ”Video Frame Interpolation: A Comprehensive Survey,” ACM Transactions on Multimedia Computing, Communications and Applications, 2022, pp. 2001-2031.

[49] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, ”Flownet 2.0: Evolution of optical flow estimation with deep networks,”in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 2462-2470.

[50] S. Dutta, A. Subramaniam, and A. Mittal, ”Non-linear motion estimation for video frame interpolation using space-time convolutions,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 1726-1731.

[51] M. Park, H.G. Kim, S. Lee, and Y.M. Ro, ”Robust video frame interpolation with exceptional motion map,” IEEE Transactions on Circuits and Systems for Video Technology, 2020, vol. 31, no. 2, pp. 754-764.

[52] W. Shen, W. Bao, G. Zhai, L. Chen, X. Min, and Z. Gao, ”Video frame interpolation and enhancement via pyramid recurrent framework,” IEEE Transactions on Image Processing, 2020, vol. 30, pp. 277-292.

[53] P. Hu, S. Niklaus, S. Sclarof, and K. Saenko, ”Many-to-many splatting for efficient video frame interpolation,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 3553-3562.

[54] T. Kalluri, D. Pathak, M. Chandraker, and D. Tran, ”Flavr: Flow-agnostic video representations for fast frame interpolation,” in Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2023, pp. 2071-2082.

[55] X. Cheng and Z. Chen, ”Video frame interpolation via deformable separable convolution,” in Proceedings of the AAAI Conference on Artificial Intelligence, 2020, vol. 34, pp. 10607-10614.

[56] X. Cheng and Z. Chen, ”Multiple video frame interpolation via enhanced deformable separable convolution,” IEEE Transactions on Pattern Analysis and Machine Intelligence, 2021, no. 10, pp.7029-7025.

[57] O. Ronneberger, P. Fischer, and T. Brox, ”U-Net: Convolutional Networks for Biomedical Image Segmentation,” Lecture Notes in Computer Science, 2015, vol. 9351, pp. 234-241.

[58] C. R. Harris, “Array Programming with NumPy,” Nature, vol. 585, no. 7825, Sep. 2020, pp. 357–362.

[59] M. Abadi, “TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems,” 2015.

[60] S. Niklaus and F. Liu, ”Context-aware synthesis for video frame interpolation,” in Proceedings of IEEE/CVF Conference on Computer Vision Pattern Recognition, Salt Lake City, USA, 2018, pp. 1701-1710.

[61] H. Lee, T. Kim, T.Y. Chung, ”Adacof: Adaptive collaboration of flows for video frame interpolation,” in Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, USA, 2020, pp. 5316-5325.

[62] X. Yang, H. Zhang, Z. Qu, Z. Feng, and J. Tian, ”Video frame interpolation via residual blocks and feature pyramid networks,” IET Image Processing, 2023, no. 4, pp. 1060-1070.

[63] T. Ding, L. Liang, Z. Zhu, and I. Zharkov, ”CDFI: Compression Driven Network Design for Frame Interpolation.” in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2024, pp. 8001-8011.

[64] L. Kong, B. Jiang, D. Luo, ”IFRNet: Intermediate feature refine network for efficient frame interpolation,” in Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, USA, 2022, pp. 1969-1978.

[65] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, and A. Agarwala, “Video Frame Synthesis using Deep Voxel Flow,” arXiv (Cornell University), Feb. 2017.